ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2020. No. 2 (12)

UDC 004.942 DOI: https://doi.org/10.30837/2522-9818.2020.12.040

G. ZHOLTKEVYCH

METRICS FOR EVALUATING CONSISTENCY IN DISTRIBUTED DATASTORES

The **subjec**t of the paper is metrics for evaluating consistency of distributed datastore as one of main CAP-guarantees, more precisely, criteria for reliable distributed datastore. The **goal** of the research is investigation of the ability to develop such a program on the earlier stage of building distributed network and build some components of decision-making algorithm, which purpose is to build optimal network topology. This decision-making algorithm should be suitable for any business model and its requirements. To be more detailed, for that purpose the following **tasks** had been done: mathematical model for stochastic metric for consistency in distributed datastore is built; the conditions of consistency convergence time are investigated in initial perfect datastore environment. **Methods** used are: theory of number partitions, basics from graph theory and probability theory, computer modeling and program for running sets of experiments. As a **result**, it is established that in the conditions of data loss absence the consistency convergence after first write request is equal or less than diameter of graph that represents topology of distributed network. Such convergence has the same unit of measure as the link cost of each link in the network; the stochastic model is proposed for metric to evaluate consistency. Making a final **conclusion**, this will give the opportunity to investigate or monitor the current state of the system in the given time interval. This research is the base to form some elements of decision-making algorithm for building topology in a distributed network and the elements of the algorithm for monitoring such a system. Also, based on trends of requests frequency of data modification and reading, the strategy of nodes allocation in the topology is suggested, which can improve the response time and speed of convergence of the distributed storage to the fully consistent or close to that state. The practical role of the components of the decision-making algorithm is that the network architect could apply the algorithm at the stage of building the network for a distributed database, so that CAP characteristics will be optimized in the context of specific business needs. The mathematical model for the stochastic metric of distributed storage consistency can be applied both at the system design stage, for testing the satisfactory level of consistency, and at the system operation stage, as a component of the network monitoring system.

Keywords: distributed datastore; response time; CAP-theorem; stochastic consistency metric; methods of building distributed network.

Introduction

In the epoch of popular usage of IoT, Big Data, Cloud Computing, the data become more and more important thing and require larger, more reliable storage [1]. This leads to increasing size of distributed storages. They become bigger and require the huge network across all the distributed nodes. But there are several unsolved problems using large distributed datastores and some of them tied to the CAP-theorem.

Given that the ACID strategy cannot be supported for systems of this class, mechanisms for delivering data across distributed storage still lack fast eventual consistency convergence, reliability and tolerance to network partitions (basic factors of the reliable datastore that are defined in [2]).

In this paper we concentrate on efficient and widely used consistency model - BASE (Basically Available, Soft State, Eventual Consistency), specifically, on eventual consistency as one of the component of this model.

Problem statement

Supporting replicas of an evolving distributed system up-to-date in the conditions of BASE is important, but hard problem. Thus, we need to provide a set of indicators of main characteristics of the distributed data store to assess the risk of a wrong decision because of the data inconsistency or unavailability.

In this work, we focus our study on the problem of metrics investigation for evaluating data consistency in a distributed data store. We want to demonstrate the useful concept of defining the consistency metric as stochastic value and abilities that it gives so far.

For that we propose stochastic metric for eventual consistency / inconsistency instead of binary one.

Also, we would like to know if it is possible to find the optimal interval between writable operations occurring on distributed systems so that system can eventually be consistent in that interval. To achieve that, we develop consistency model defined in the previous paper (analyzed and referenced in the next section) and extend by the formula of inconsistency metric for a distributed datastore. Then we approximate the time that is needed for consistency convergence – state of the distributed system when all nodes have consistent replicas. Afterwards, we conduct experiments that prove the correctness of metric defined in previous section. Finally, we provide diagrams of the implemented application that we use for carrying out experiments and make some assumptions for decision-making algorithm for building optimal network topology for a distributed datastore that is the goal of the current research.

Analysis of related research

During the latest decade the problem of scaling distributed systems and their effectiveness is investigated quite deeply. The evaluation of scaling such systems and its reasonability is presented in [1]. But needs for storing more data are growing fast and new research works are needed.

In [3] mainly used models for such systems and its comparative characteristics was considered (ACID and BASE). But here appears some problem. Historically, before final proof of CAP-theorem Eric Brewer had been investigating the opportunity of strong consistent, highly available and partition-tolerant distributed datastore and made a CAP-hypothesis [4]. The formal proof of the

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2020. № 2 (12) ISSN 2524-2296 (online)

theorem had been presented in 2002 [5], some clarifications appeared in 2012 [6].

This theorem became one of important blockers on the step of distributed datastores enhancement and the theorem and BASE and ACID models has been deeply investigated during 15 years [3]. Since lots of systems have chosen BASE model, sacrificing strong consistency and durability, eventual consistency was deeply analyzed in [7]. The problem of that it is impossible to fulfill consistency, partition tolerance and availability, is also investigated in [8]. There it is considered if there is an ability to reach a trade-off between AP (availability, partition tolerance), CP (consistency, partition tolerance) and CA. This work is mainly devoted to the analysis of theorem, but it does not declare the formal model for methods overcoming the problem. Microsoft came closer and had developed algorithms for maintaining strong consistency in [9]. But the problem is investigated in the conditions of about 15 nodes in a system and scaling in this paper seems to be for the future work. Also, in [10] the problem of replicas conflicts is investigated in conditions of parallel write requests to many nodes. Methods of evaluating of consistency metrics have been defined in [14] and the methods of improvement of consistency state value have been proposed in [15].

Materials and methods

In the previous paper [11] we considered the metrics for all three elements of CAP-theorem. Now our paper is mainly devoted to consistency question and how fast it can converge. From that paper we are taking the developed mathematical model and expand it with elements, needed for the current research. We specify this model as

$$(N, L, \partial, D, r, N_d, l(N_d), n_c), \quad (1)$$

where N – finite set of nodes in a datastore; L – finite set of links in a datastore; $\partial : L \to 2^N$ – mapping where each link is associated with two adjacent nodes; D – finite set of stored data units; $r : D \to 2^N$ – mapping that associates each of data unit to a set of nodes that store the replica of this data unit; N_d – finite set of nodes that store the given data unit d; l(N_d) – the number of nodes that store data unit d; n_c – the number of nodes in a subset N_d, where all the nodes have the same version of replica.

So, we expanded the model now with three last assumptions above.

Our mathematical model for inconsistency I will be able to define the inconsistency state of distributed datastore (see (1)), and afterwards will focus on time which distributed datastore require to become fully consistent. Further we call it consistency convergence time.

For this metric we created the probabilistic event taking two random nodes from the set N_d, they can be consistent with some probability. Carrying out this event many times, we can obtain the average probability of that two nodes will be consistent.

Thus, we have the following sample space:

$$\Omega = \{0,0,1,0,1,0,1,1,0,1,1,....\}, \quad (2)$$

where 0 denotes the event when two nodes are consistent and 1, on the contrary, when two nodes are inconsistent. Also let's claim that p is the probability that two nodes are consistent, and $q = 1 - p$ – the probability that two nodes are inconsistent. So, we introduce the stochastic metrics for inconsistency/consistency instead of binary ones. Carrying out experiments, we will "freeze" the simulated distributed datastore and time of that "freezing" we call time t. Thus, we will be able to investigate the current state of the system. Let's denote I as a value calculated by probabilistic formula. We are thinking that having this formula will help us to eventually fully specify the mathematical model for consistency, so let it be one of mathematical model components. So, the probability of that two nodes taken at random are from consistent subset is

$$p_c = \frac{n_c}{l(N_d)} \cdot \frac{n_c - 1}{l(N_d) - 1} = \frac{n_c * (n_c - 1)}{l(N_d) * (l(N_d) - 1)}, \quad (3)$$

where n_c is length of consistent subset in the system at time t.

Obviously, inconsistency probability then is

$$p_i = 1 - p = 1 - \frac{n_c * (n_c - 1)}{l(N_d) * (l(N_d) - 1)}. \quad (4)$$

Taking into account that minimum number of consistent nodes will be equal to 1, and the maximum number of consistent nodes is equal to $l(N_d)$, we have following consequences:

Two nodes are inconsistent if $p_{ic} = 1$.

Two nodes are consistent if $p_{ic} = 0$.

We developed the inconsistency formula for two nodes. Let's now extend it to more general one. We still suppose that a data unit is represented by replicas on N_d servers. Let's denote it as temporary N.

Let we have K classes of mutually consistent replicas. Then we denote by N_k a number of replicas in k^{th} consistency class $(1 \le k < K)$. It is evident that $N_k > 0$ for all $k = 1, \ldots, K$ and $N = N_1 + \ldots + N_K$. Such representations $N = N_1 + \ldots + N_K$ are called integer partitions. Thus, in this case any integer partition describes some inconsistency state. We take integer partitions well-known example from [12] book.

Example. *All integer partitions for* 5 *(5) are*

5	4+1	3+2
3+1+1	2+2+1	2+1+1+1
	1+1+1+1+1	

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2020. No. 2 (12)

Taking into account this consideration we define inconsistency metric for a data unit in the term of the corresponding integer partition. Let's suppose that the studied data unit d is represented by N replicas, which are being described by the integer partition $N = N_1,\ldots,M_K$.

Then the inconsistency metric $I(d)$ is defined by the formula

$$I(d)=1-\sum_{k=1}^{K}\frac{N_k(Y_k-1)}{N(N-1)}. \quad (5)$$

Let's look at one more example:

Example. *Let us suppose that a data unit u are being stored on five servers. Then the corresponding inconsistency states (see Example (5)) have the following values of the inconsistency metric* (6)

$$I(5)=0$$

$$I(4+1)=\frac{2}{5} \qquad I(3+2)=\frac{3}{5}$$

$$I(3+1+1)=\frac{7}{10} \qquad I(2+2+1)=\frac{4}{5}$$

$$I(2+1+1+1)=\frac{9}{10} \qquad I(1+1+1+1+1)=1$$

The meaning of $I(d)$ is the value of probability to establish the fact of inconsistency by the way of comparison two randomly chosen replicas.

Example (7) demonstrates that the proposed metric is equal to 0 for the absolutely consistent data unit (the case with 5 partitions) and is equal to 1 for the absolutely inconsistent data unit (the case 1+1+1+1+1 partitions) in the subset N_d.

Now we are interested to calculate the time that can be taken for consistency convergence. We want to prove the following:

Proposition. *Let we have a distributed datastore where all links are available and reliable (network partitions do not happen in a datastore and nodes are stable and respond in approximately equal time); the interval between writing operations is t_w. If and only if such input conditions are met, then t_w is less than the diameter of network graph ensures eventual consistency of the datastore.* (7)

Proof. Let us denote as T_c time for consistency convergence (time that is needed for the whole system to become consistent). Let we have the trivial network where all links have link cost 1. Thus, as the input we have the connected graph G with set of nodes N and set of edges E. Let us assume now that weight of each edge $e \in E$ satisfies the equality

$$w(e)=1. \quad (8)$$

Let us take the nodes n_1 and n_2 that are at the largest distance each from other. So we can count that the time of delivering replica between n_1 and n_2 is the shortest path from n_1 to n_2. Extrapolating this to all the system and taking into account that in the distributed datastore nodes are broadcasting each to other in parallel, we obtain the upper boundary of T_c is the maximum of shortest paths in the worst case. It is well-known that such maximum is diameter of the graph satisfies the definition of graph diameter. (see [13]). So, then we can conclude:

$$T_c = diameter(G) \quad (9)$$

Let us complicate the system introducing the different link cost for links in distributed datastore that means that now our graph G is weighted and $w(e) \in \mathbb{N}$ for all $e \in E$.

Thus, the diameter is the path $P=[e_1....e_n]$ where e_i has own weight and

$$T_c=\sum_{i=1}^{n} w_i, \quad (10)$$

where w_i is the weight of edge e_i of the path P. T_c is the number of time slots that a datastore requires to become fully consistent. This means that after T_c time points, all replicas become consistent. This also means that a datastore is again available to accept writing requests, so that datastore will be able to store all the replicas passed before, and no accidental updating happen in the meantime.

Case Study

The study in the previous section had been checked by the following experimentation: we implemented a code that allows to test the accuracy of the inconsistency metric. Probability intervals for different partitions are demonstrating that the formula is correct – values obtained are around values obtained theoretically. To be more intuitive we took the same number of nodes and same partitions. It is obviously that we do not have exact matching, because the experiment is based on number of iterations where two nodes are taken from a given set of nodes randomly, but all we need is that value should vary slightly around theoretical value. Look in the figures below:

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2020. № 2 (12) ISSN 2524-2296 (online)

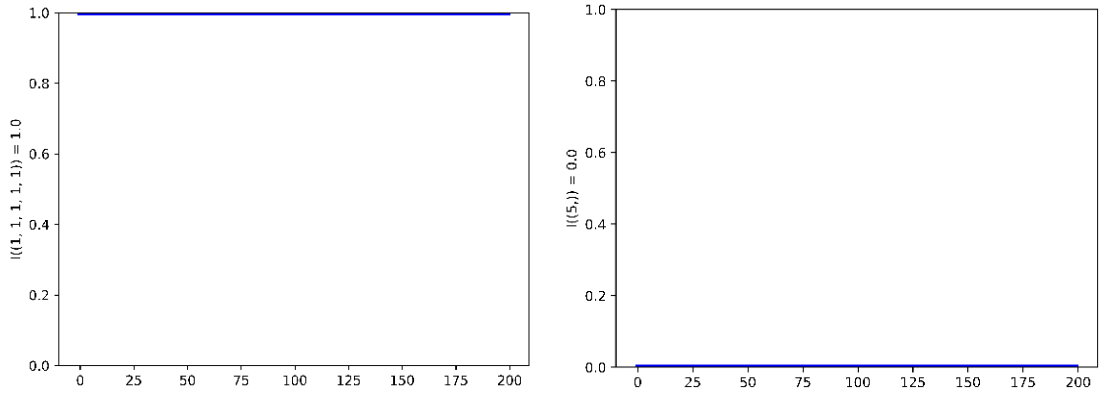

Fig. 1. Datastore is fully inconsistent and datastore is fully consistent

Firstly, it can be seen that for one consistent partition $I(d)$ is equal to 0 and when the system is fully inconsistent $I(d)$ is equal to 1 (see in fig. 1).

Let us compare now the situation when we have two consistent partitions: the set which contains 2 and 3 nodes that consistent in the own subset, and the set with 4 and 1-length consistent subsets (see results in fig. 2).

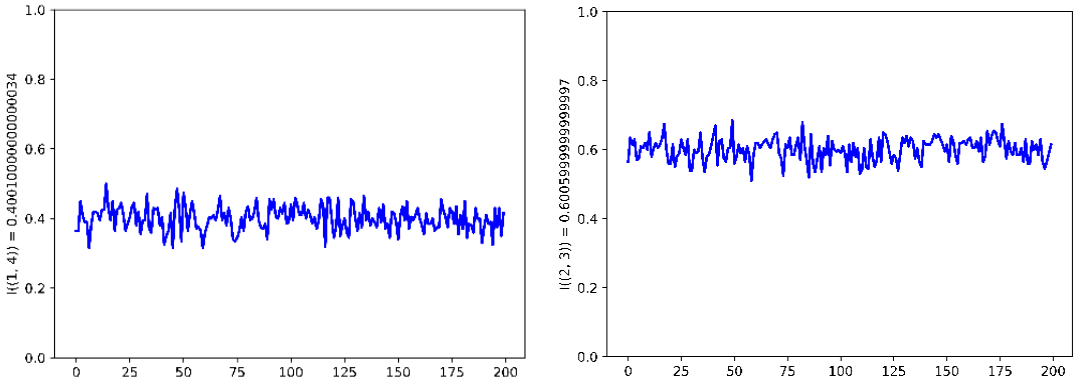

Fig. 2. The datastore has two consistent partitions: (1,4) and (2,3)

Then we can easily see that:

$$I(3,2)=1-\frac{3\cdot 2}{5\cdot 4}-\frac{2\cdot 1}{5\cdot 4}=\frac{3}{5}, \quad (11)$$

and

$$I(4,1)=1-\frac{4\cdot 3}{5\cdot 4}-\frac{1\cdot 0}{5\cdot 4}=\frac{2}{5}. \quad (12)$$

Following this procedure for three consistent partitions (presented in fig. 3) in N_d :

$$I(3,1,1)=1-\frac{3\cdot 2}{5\cdot 4}-\frac{1\cdot 0}{5\cdot 4}-\frac{1\cdot 0}{5\cdot 4}=\frac{7}{10} \quad (13)$$

and

$$I(2,2,1)=1-\frac{2\cdot 1}{5\cdot 4}-\frac{2\cdot 1}{5\cdot 4}-\frac{1\cdot 0}{5\cdot 4}=\frac{4}{5}. \quad (14)$$

And the final one:

$$I(2,1,1,1)=1-\frac{2\cdot 1}{5\cdot 4}-3\cdot\frac{1\cdot 0}{5\cdot 4}=\frac{9}{10}. \quad (15)$$

See results in fig. 4.

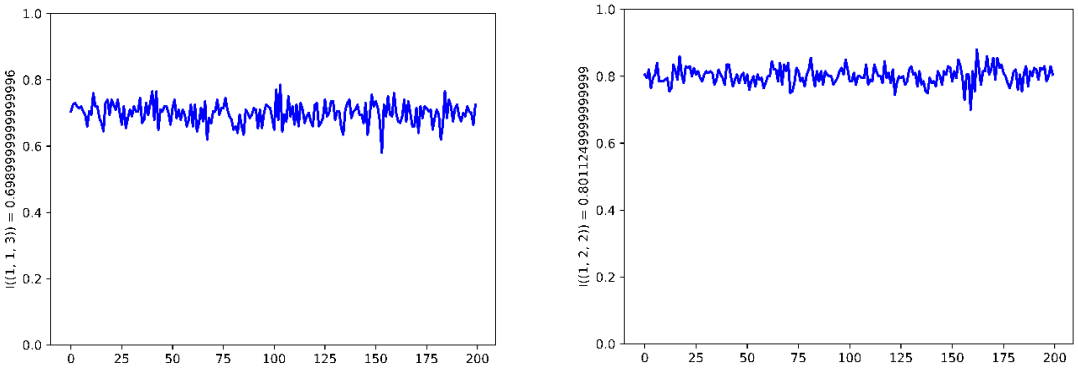

Fig. 3. The datastore has three consistent partitions: (1,1,3) and (1,2,2)

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

So, we had easily shown on a simple subset, that our inconsistency metric value corresponds to theoretical one. Now we present the graphic that demonstrates the verity of that claim about that consistency convergence time T_c for graph G will be no greater than diameter of G.

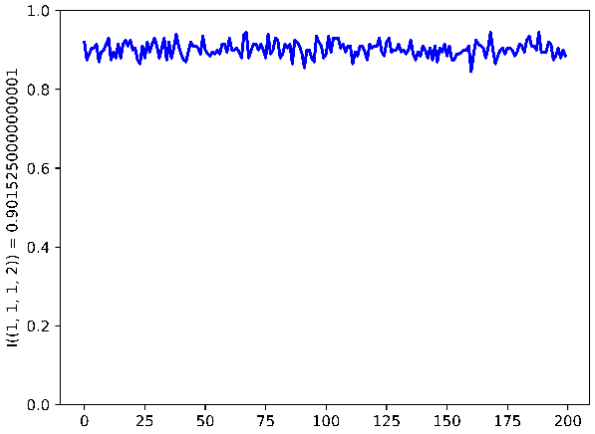

Fig. 4. The datastore has four consistent partitions (1,1,1,4)

To be more intuitive, we carried out the set of experiments on our simulation model: having simulated distributed datastore, we easily could run the imitation of one message broadcasting through the datastore and calculate the number of time slots it is taking in the real time. We have done it for graph with each edge of link cost of 1. Below there are graphics for 100, 200, 1000 experiments respectively (see in fig. 5). We draw graphics in the following manner: obtain the diameter of graph G (ordinates) and calculated by the simulation T_c (axis). We can easily see that either we have the line demonstrating that $T_c = D(G)$, or points that are showing that $T_c < D(G)$.

So, now the case of an unweighted graph is considered. But the general situation requires to consider weighted one. In this case, weight of an edge means the average time for delivering a message via the corresponding link.

However, algorithms to simulate this case are more complex. Therefore, this general case will be considered in the future, expanding experiments for random regular graph. Now we are able to present some results for random regular graph, that substantiate our proposition (1.8) in the general case (see in fig. 6). We can observe that all points have a location in the graphics such that T_c is less or equal to $D(G)$ for weighted random regular graph.

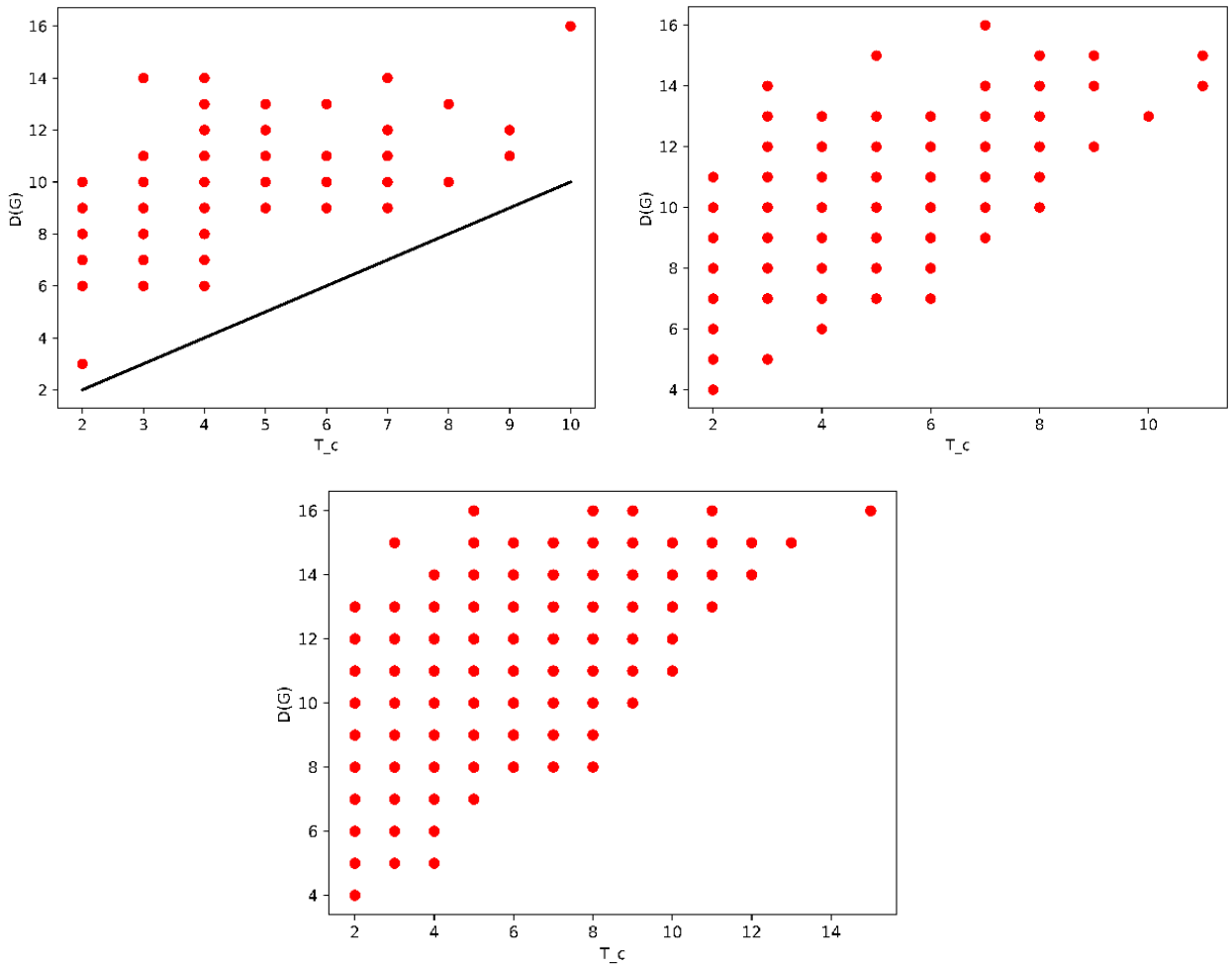

Fig. 5. Graphics for consistency convergence time on non-weighted graph

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2020. № 2 (12) ISSN 2524-2296 (online)

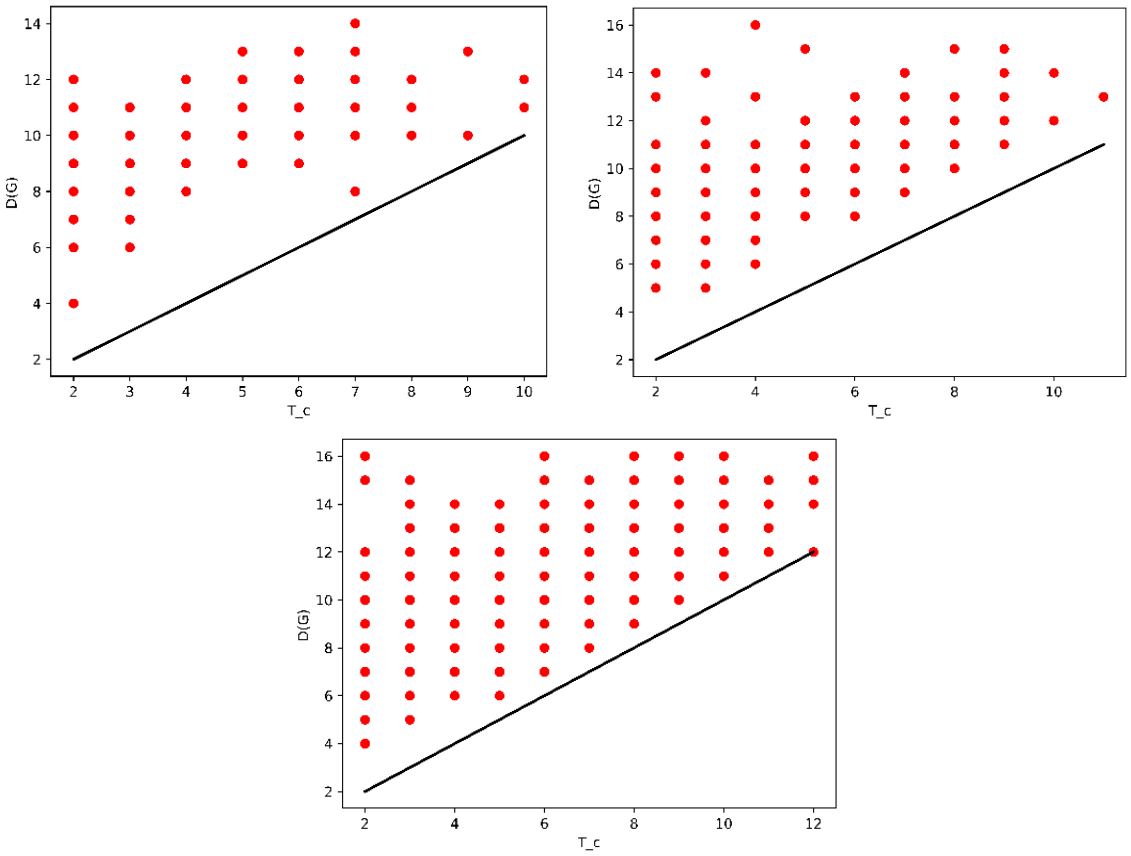

Fig. 6. Graphics for consistency convergence time on weighted graph

Simulation Model to Estimate Inconsistency Ratio of Data. To carry out experiments, we needed to implement the simulation model that will do experiment and calculate all needed values. This model is implemented as a computer program in Python language and available our project page. We assume that it would be useful to present the short class diagram for the simulation model (see fig. 7 below).

Also, we would like to present the state machine diagram for the algorithm that was implemented to carry out experiments of consistency convergence estimation on a distributed datastore (see fig. 8).

For experiment simulation we had chosen random regular graph. During simulation we could see that for degree great than 2, calculating the time-slots taken for consistency convergence for one iteration, we can take a minimum between paths to neighbors and it will be the correct choice.

Because for regular graph if the path to another neighbor is greater, there will be another path with lesser link cost that will take less time to converge.

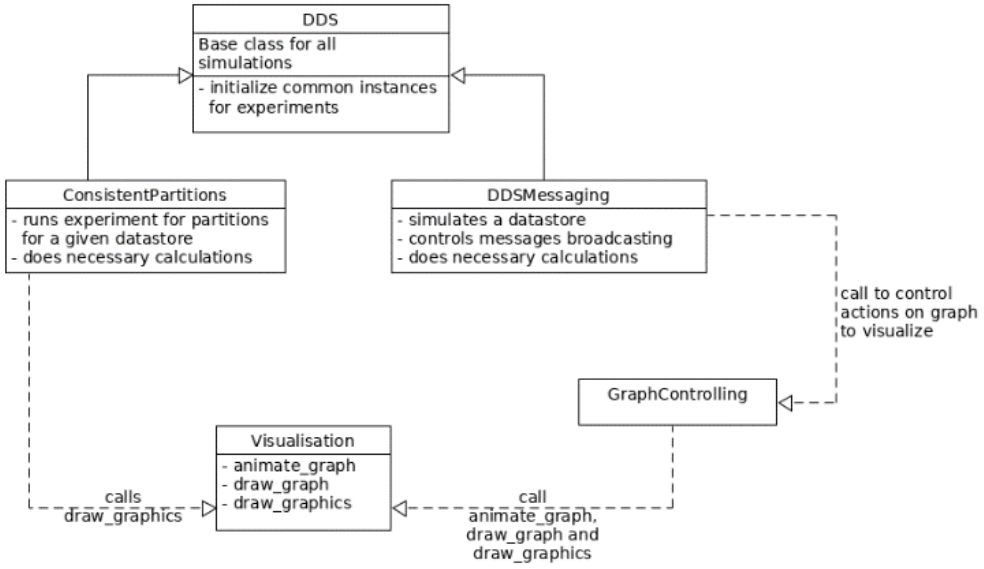

Fig. 7. Class diagram for simulation model of a distributed datastore

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2020. No. 2 (12)

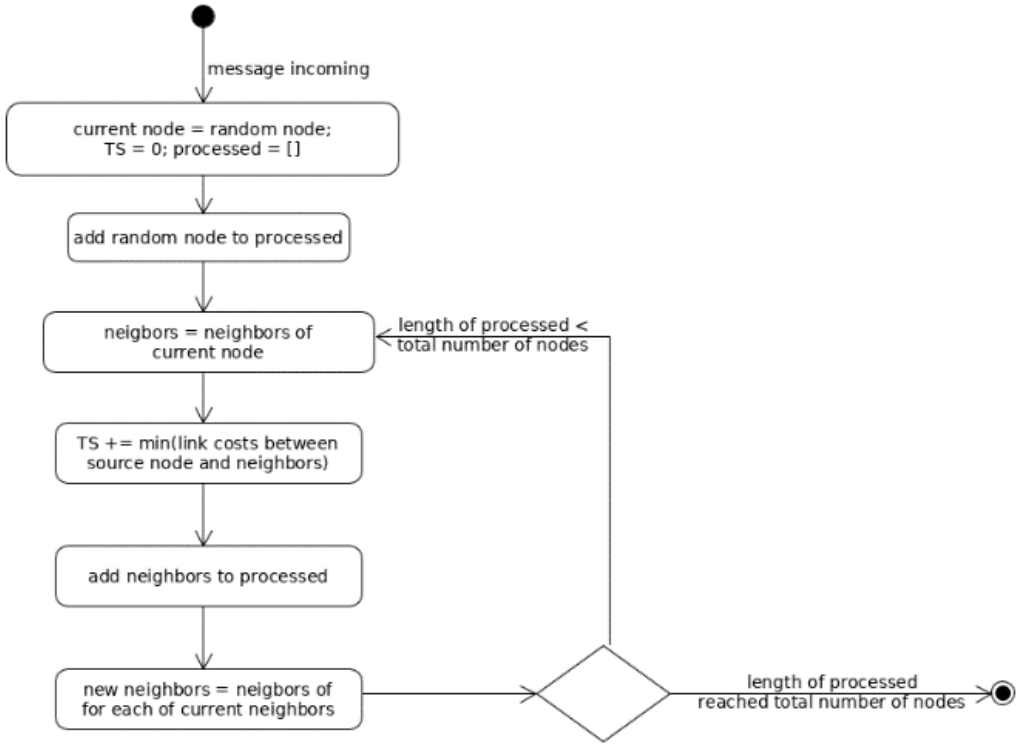

Fig. 8. State machine of simulation of message broadcasting through a datastore

Based on the research of current paper we can make following recommendations for those who build the network topology of the distributed datastore:

- in the case of your datastore is a system with a domination of read operations, it must be sufficient to choose such network topology where frequency of write operations will be no greater than diameter of the graph representing network topology of a distributed datastore;

- if frequency of write operations is greater than the diameter of the graph, it may be useful to evaluate current inconsistency state of the graph;

- if inconsistency state is close to 0 enough for current requirements, it means that there are inconsistent nodes that are far enough to not conflict with new replicas. Thus, developer or administrator of a datastore can choose as source available for writing that node that is in consistent list and far enough for inconsistent ones. But for that developer needs to provide such an algorithm for a datastore so that he will be able to obtain the current list of consistent nodes;

- if still more strict consistency needs to be satisfied and inconsistency state is close enough to 0, a developer or administrator of a datastore can choose as source node for writing the node closest to the source node where previous write operation occurred;

- if replica's history is not important for requirements, it may be possible to solve the problem fixing conflicts. This problem has been investigated in the paper [10].

Also, we would like to notice that the ways of improving routing on some types of networks has already been investigated in [16], [17]. This may be one of approach that will help to build better network for distributed datastores.

Conclusions

This research is devoted to mathematical model for distributed datastore that is evolved with the metric in the form of stochastic formula that allows to measure the consistency at certain time slot in distributed datastore. This work continues the previous research [11], which is the analysis of CAP-guarantees and first version of mathematical model for distributed datastore guarantees. As the result of this paper, this mathematical model is enhanced with new stochastic formula, based on probability theory basics and number partitions theory. Also, we define the necessary condition for the time that it takes for consistency to converge to perfect value in the conditions of data loss absence. This result has a theoretical and practical proof in the form of set of experiments that were run on big set of nodes. Basing on this theory built we succeeded to make some claims and recommendations for building network topology of a datastore. In the future this should result in integral decision-making algorithm that could be applied as a part of algorithm during design distributed datastore stage.

For the future work we consider to extend the formula to calculate mean time of message delivery in distributed system in the conditions of dataloss and network partition.

References

1. Kuhlenkamp, J., Klems, M., Röss, O. (2014), "Benchmarking Scalability and Elasticity of Distributed Database Systems", *PVLDB*, No. 7, P. 1219–1230.

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2020. № 2 (12) ISSN 2524-2296 (online)

2. Tanenbaum, A. S., Steen, M. V. (2007), *Distributed systems - principles and paradigms*, 2nd Edition, Upper Sddle River, Prentice-hall, 686 p.
3. Banothu, N., Bhukya, S. and Sharma, K. (2016), "Big-data: Acid versus base for database transactions", *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, P. 3704–3709. DOI: 10.1109/ICEEOT.2016.7755401.
4. Brewer, E. A. (2000), "Towards robust distributed systems (abstract)", *PODC '00: Proceedings of the nineteenth annual ACM symposium on Principles of distributed computingJuly 2000*, DOI: https://doi.org/10.1145/343477.343502
5. Gilbert, S., Lynch, N. A. (2002), "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services", *SIGACT News*, No. 33, P. 51–59.
6. Brewer, E. A. (2012), "CAP twelve years later: How the "rules" have changed", *Computer*, No. 45, P. 23–29.
7. Bailis, P., Ghodsi, A. (2013), *Eventual Consistency Today: Limitations, Extensions, and Beyond*, QUEUE, Vol. 11, Issue 3, P. 9–13, available at : https://dl.acm.org/doi/pdf/10.1145/2460276.2462076?download=true
8. Gilbert, S., Lynch, N. A. (2012), "Perspectives on the CAP Theorem", *Computer*, No. 45, P. 30–36.
9. Calder, B., Wang, J., Ogus, A., Nilakantan, N., Skjolsvold, A., et. al. (2011), "Windows Azure Storage: a highly available cloud storage service with strong consistency", *SOSP '11*, available at : https://azure.microsoft.com/en-us/blog/sosp-paper-windows-azure-storage-a-highly-available-cloud-storage-service-with-strong-consistency/.
10. Madria, S. K. (1998), "Handling of Mutual Conflicts in Distributed Databases Using Timestamps", *Comput. J.*, No. 41, P. 376–385.
11. Rukkas, K., Zholtkevych, G. (2015), "Distributed Datastores: Towards Probabilistic Approach for Estimation of Dependability", *ICTERI, Computer Science*, available at : https://pdfs.semanticscholar.org/5eb0/01632c6cd6da2e4ec92adbc288939de0f4f9.pdf?_ga=2.235973185.1723289165.1592897490-2045290888.1592897490.
12. Andrews, G. E. (1976), *The theory of partitions*, Cambridge University Press, 255 p.
13. Bondy, J. A., Murty, U. S. (1976), *Graph Theory with Applications*, Elsevier Science Ltd.The Boulevard Langford Lane Kidlington, Oxford OX5 1GBUnited Kingdom, 270 p.
14. Rukkas, K., Zholtkevych, G. (2020), "Probabilistic model for estimation of cap-guarantees for distributed datastore", *Advanced Information Systems*, No. 4, P. 47–50. DPI: 10.20998/2522-9052.2020.2.09
15. Rukkas, K., Zholtkevych, G. (2020), "Load balancing consistency in a distributed datastore", *Control, Navigation and Communication Systems*, No. 2, P. 95–100. DOI: 10.26906/SUNZ.2020.2.095
16. Lemeshko, O., Yevdokymenko, M., Yeremenko, O. (2019), "Model of data traffic qos fast rerouting in infocommunication networks", *Innovative Technologies and Scientific Solutions for Industries*, No. 3 (9), P. 127–134. DOI: https://doi.org/10.30837/2522-9818.2019.9.127
17. Yeremenko, O., Yevdokymenko, M., Sleiman, B. (2020), "Advanced performance-based fast rerouting model with path protection and its bandwidth in software-defined network", *Innovative Technologies and Scientific Solutions for Industries*, No. 1 (11), P. 163–171. DOI: https://doi.org/10.30837/2522-9818.2020.11.163.

Received 22.05.2020

Відомості про авторів / Сведения об авторах / About the Authors

Жолткевич Галина Григоріївна – дослідник, інженер – розробник програмного забезпечення, приватний підприємець, Харків, Україна; email: galynazholtkevych1991@gmail.com; ORCID: https://orcid.org/0000-0002-9772-4691.

Жолткевич Галина Григорьевна – исследователь, инженер – разработчик программного обеспечения, частный предприниматель, Харьков, Украина.

Zholtkevych Galyna – Researcher, Software Engineer, a Private Entrepreneur, Kharkiv, Ukraine.

МЕТРИКИ ДЛЯ ОБЧИСЛЕННЯ УЗГОДЖЕНОСТІ У РОЗПОДІЛЕНИХ СХОВИЩАХ ДАНИХ

Предметом дослідження статті є метрики для обчислення стану узгодженості у розподіленому сховищі даних як одної з найважливіших критеріїв надійного розподіленого сховища даних. **Метою** роботи є дослідження можливості розроблення програми, яка буде працювати на ранніх етапах проектування розподіленої мережі та побудувати компоненти для алгоритму прийняття рішень, метою якого є побудування оптимальної топології мережі. Такий алгоритм має задовольняти будь-яку бізнес-модель та її потреби. Для цього наступні **задачі** були вирішені: побудована математична модель для стохастичної метрики оцінювання стану узгодженості, сформовані умови для збіжності часу узгодженості в початкових умовах ідеального середовища розподіленого сховища. Використані **методи**: теорія числових розділень, базові поняття та формули з теорії графів та теорії ймовірності, комп'ютерне моделювання та програма для проведення експериментів. Як **результат**, встановлено, що в умовах середовища без втрат даних значення збіжності стану узгодженості після першого запиту на запис менше або дорівнює діаметру графу, що відображає топологію мережі. Таке значення має таку ж саму одиницю вимірювання, що і "link cost" кожного зв'язку в мережі. Також, пропонована стохастична модель для метрики оцінювання стану узгодженості. Це дасть можливість моніторингу поточного стану узгодженості системи у заданому часовому інтервалі. Це дослідження є базою для формування елементів алгоритму прийняття рішень для побудови топології в розподіленій мережі та елементів алгоритму моніторингу системи. Також, на основі частоти запитів на запис та читання даних, пропонована стратегія розташування вузлів у мережі, що може зменшити час на відповідь системи, порівняно, якщо не використовувати цю стратегію. Роблячи **висновок**, практична роль компонентів алгоритму прийняття рішень – допомога архітектору великої розподіленої мережі сховища на етапі проектування, і як результат, CAP-характеристики будуть задовільнені оптимально для конкретних бізнес-потреб. Математична модель для стохастичної метрики оцінювання

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2020. No. 2 (12)*

узгодженості розподіленого сховища може бути застосована як і на етапі проектування системи, для тестування задовільного рівня узгодженості, так і на етапі операційної підтримки системи у якості компонента моніторингу.

Ключові слова: розподілені сховища; час на відповідь; CAP-теорема; стохастична метрика узгодженості; методи побудування розподіленої мережі.

МЕТРИКИ ДЛЯ ВЫЧИСЛЕНИЙ СОГЛАСОВАННОСТИ В РАСПРЕДЕЛЕННЫХ ХРАНИЛИЩАХ ДАННЫХ

Предметом исследования являются метрики для вычисления состояния согласованности в распределенном хранилище данных как одни из важнейших критериев надежного распределенного хранилища. **Цель** работы – исследования возможности разработки программы, которая будет работать на ранних этапах проектирования распределенной сети и построить компоненты для алгоритма принятия решения, который строит оптимальную топологию. Такой алгоритм должен удовлетворять требования любой бизнес-модели. Для этого следующие **задачи** были решены: построена математическая модель для стохастической метрики оценки состояния согласованности, сформированы условия для времени сходимости согласованности в начальных условиях идеального окружения распределенного хранилища. Использованы следующие **методы**: теория числовых разделений, базовые понятия теории графов и теории вероятности, компьютерное моделирование и компьютерная программа для проведения экспериментов. Как **результат** установлено, что в условиях окружения без потерь данных значения времени сходимости состояния согласованности после первого запроса на запись меньше или равняется диаметру графа, который отображает топологию сети. Такое значение имеет ту же единицу измерения, что и "link cost" каждой связи в сети. Также, предложена стохастическая модель для метрики оценивания состояния согласованности, что даст возможность мониторинга текущего состояния согласованности в заданном часовом промежутке. Это исследование является базой для формирования элементов алгоритма принятия решений для построения топологии распределенной сети и элементов алгоритма мониторинга системы. Также, на основе частоты запросов на запись и чтение данных, предложена стратегия расположения узлов в сети, что может сократить время на ответ от системы пользователю, в сравнении с ситуацией неиспользования этой стратегии. Делая **выводы**, практическая роль компонентов алгоритма принятия решений – помощь архитектору большой сети распределенной базы данных на этапе проектирования, в следствии чего CAP-характеристики будут оптимально сбалансированы для конкретных бизнес-требований. Математическая модель для стохастической метрики оценивания распределенного хранилища может быть применена как на этапе проектирования системы, а именно, для тестирования удовлетворительного уровня согласованности, так и на этапе операционной поддержки системы в качестве компонента мониторинга.

Ключевые слова: распределенные хранилища; время отклика; CAP-теорема; стохастическая метрика согласованности; методы построения распределенной сети.

Бібліографічні описи / Bibliographic descriptions

Жолткевич Г. Г. Метрики для обчислення узгодженості у розподілених сховищах даних. *Сучасний стан наукових досліджень та технологій в промисловості*. 2020. № 2 (12). С. 40–48. DOI: https://doi.org/10.30837/2522-9818.2020.12.040.

Zholtkevych, G. (2020), "Metrics for evaluating consistency in distributed datastores", *Innovative Technologies and Scientific Solutions for Industries*, No. 2 (12), P. 40–48. DOI: https://doi.org/10.30837/2522-9818.2020.12.040.