ІНФОКОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ ТА ЕЛЕКТРОННА ІНЖЕНЕРІЯ
INFORMATION AND COMMUNICATION TECHNOLOGIES, ELECTRONIC ENGINEERING

№ 4 (1), 2024

DEVELOPMENT OF NETWORK SIMULATION MODEL FOR EVALUATING THE EFFICIENCY OF DISTRIBUTED CONSENSUS TAKING INTO ACCOUNT THE INSTABILITY OF NETWORK CONNECTIONS

S. Zhuravel [ORCID: 0009-0006-8818-4255]

Lviv Polytechnic National University, S. Bandery Str., 12, 79013, Lviv, Ukraine

Corresponding author: Stanislav Zhuravel (e-mail: stanislav.s.zhuravel@lpnu.ua).

(Received 1 March 2024)

The dynamic and unpredictable nature of network environments poses a significant challenge for distributed systems, particularly those relying on consensus algorithms for state management and fault tolerance. To address this challenge, this article introduces a novel simulation model designed to study the impact of unstable network connections on clusters running consensus algorithms. The model is engineered to mimic varying degrees of network instability, including latency fluctuations and connection disruptions, which are characteristic of real-world distributed systems. Our proposed model represents a significant advancement in the simulation of distributed networks. It employs a sophisticated network emulation layer capable of generating a wide spectrum of unstable network conditions. The core of the model is a highly configurable consensus mechanism simulator that allows for the adjustment of key parameters such as heartbeat intervals, election timeouts, and message loss rates. This level of configurability enables a comprehensive analysis of consensus behaviors under different network scenarios. The article focuses on the methodology behind the development of the model, detailing the theoretical underpinnings and the implementation strategies used to ensure a realistic representation of network instability. We also discuss the potential applications of the model, which extend beyond academic research into practical domains where distributed ledger technologies and distributed databases are prevalent. Through the deployment of this model, researchers and system architects can gain deeper insights into the resilience and adaptability of consensus algorithms. The model serves as a tool for preemptively identifying and addressing potential issues in distributed systems, facilitating the development of more robust and reliable technologies. In summary, the article showcases the design and capabilities of a new model that enables an in-depth understanding of the delicate interplay between network instability and consensus efficiency. By focusing on the model itself, the article aims to lay a foundation for future studies and improvements in the field of distributed systems.

Keywords: *consensus algorithms, network instability, fault tolerance, simulation model, distributed systems*

1. Introduction

In the intricate world of distributed computing, the resilience and efficiency of consensus algorithms are of utmost importance. These algorithms form the backbone of a wide array of critical applications, from the underpinnings of blockchain technology to the orchestration of distributed databases. Yet, one of

the most pressing challenges these systems face is the unpredictability of network conditions, which can severely impact their performance and reliability.

While much of the existing research on consensus algorithms focuses on optimizing algorithmic efficiency and enhancing fault tolerance, there's a significant oversight regarding the influence of network instability. This oversight is particularly glaring, given that real-world networks are frequently subject to fluctuations and disruptions [1].

In this article, the aim to bridge this gap by introducing a sophisticated model specifically designed to assess the impact of unstable network connections on the functioning of consensus algorithms within clusters. This model, offers a unique simulation environment that enables the precise control of network stability variables. This controlled setting is ideal for examining how consensus algorithms react to various degrees of network stress.

Here, the focus is on meticulously outlining the design and potential applications of this model, deliberately avoiding the full presentation of empirical results at this stage. The goal is to provide a detailed understanding of the model's architecture and operational capabilities. This foundational work is intended to prepare the ground for future empirical studies that will test consensus algorithms under different network conditions.

The article introduces a new tool to the academic community and invites collaborative exploration [2]. Validation of the model is anticipated to be a collective effort, with researchers applying the model across a wide variety of network scenarios to fully assess its utility. It is hoped that this model will work to significantly advance understanding of consensus algorithms and how they might be most effectively and efficiently optimized to navigate real-world network contingencies.

By following this approach, it is hoped that a more nuanced, deeper understanding of consensus algorithms can be fostered, leading to the creation of distributed systems that are far more robust in the presence of the inherent instability of networks.

2. Background and related work

In the quest for reliable distributed systems, the robustness of consensus algorithms under adverse network conditions has emerged as a paramount concern. These algorithms are the linchpin of distributed computing, enabling nodes in a network to reach agreement on a single value despite faults or discrepancies [3]. This section lays the groundwork for understanding the importance of consensus algorithms, the impact of unstable network connections, and the state of research in modeling these conditions, thus contextualizing our novel model's contribution to the field.

Consensus algorithms, such as Paxos, Raft, and Byzantine Fault Tolerance (BFT) mechanisms, are central to the operation of distributed systems. They ensure data consistency and system reliability by enabling nodes to agree on a single version of the truth, even in the face of failures. While Paxos and Raft are celebrated for their fault tolerance and ease of understanding respectively, BFT algorithms address the challenge of malicious actors, making systems resilient to a broader range of threats. Despite their strengths, these algorithms' effectiveness is significantly compromised by unstable network conditions, which can disrupt the delicate balance of distributed consensus.

Unstable network connections, characterized by high latency, packet loss, and network partitions, pose significant challenges to consensus algorithms. These conditions can lead to increased message delays, lost updates, and even conflicting decisions within a distributed system, undermining its integrity and performance. The literature has extensively explored these challenges, highlighting the need for robust solutions that can withstand such adversities. However, existing research often falls short of providing comprehensive models that accurately reflect the complexities of real-world network instability.

The pursuit of algorithms and systems capable of withstanding the unpredictable nature of network instability has led to various innovations in distributed computing [3-6]. Researchers have explored mechanisms to enhance resilience, such as adapting consensus protocols to tolerate higher degrees of

network variability and incorporating predictive models to mitigate the effects of disruptions [7]. These advancements represent significant strides towards more resilient distributed systems, yet the quest for a solution that can fully navigate the complexities of network instability continues [8-12].

The development of sophisticated simulation tools has been instrumental in understanding and improving the resilience of consensus algorithms. These tools enable researchers to model and evaluate the behavior of distributed systems under a wide range of network conditions, providing valuable insights into their performance and limitations. By simulating complex scenarios that mirror real-world challenges, researchers can identify potential weaknesses and refine algorithms to better resist network disruptions. The evolution of simulation techniques has thus played a critical role in advancing the field of distributed computing.

The reliability of consensus algorithms in the face of network instability is a critical concern for the future of distributed computing. By providing a detailed overview of the challenges and the advancements in resilience and simulation techniques, this section sets the stage for our contribution: a novel model that offers unprecedented insight into the resilience of consensus mechanisms under adverse network conditions [2]. Through this work, the aim to empower researchers and practitioners with a tool that can drive the development of more robust and reliable distributed systems, ensuring their effectiveness even in the most challenging environments.

Model is predicated on a thorough understanding of the network behavior within a distributed cluster, particularly focusing on the consensus algorithm's response to unstable network conditions.

To accurately simulate consensus algorithm behaviour, it's essential to base our models on real-world network dynamics. This foundation is vital for creating simulations that closely mirror actual network conditions, enhancing their reliability and predictive accuracy. Moving forward, the focus will shift to continually refining this model, incorporating new data to ensure it is relevant and effectively capture the complexities of network behavior. This iterative process is key to developing simulations that can guide the optimization of distributed systems with precision.

3. Network delay analysis

For an accurate simulation of consensus algorithm behavior, basing model on real-world network dynamics is crucial. This foundation ensures that simulations closely resemble actual network conditions, thus improving their reliability and predictive accuracy. As an initial step, conducting research on the nature of network delays should be prioritized.

In this study, a comprehensive dataset from an extensive network stability analysis, meticulously documented in a Kaggle notebook by Gary Stafford, was leveraged [13]. This dataset was specifically designed to capture and visualize the timing variability inherent in LAN network connections to the Internet, facilitating a nuanced understanding of delay distributions within such networks. By employing ping response timings collected at 10-second intervals from a variety of IoT collection devices, the dataset offers a dual perspective, encompassing both the 2.4 GHz wireless and 100 Mbps ethernet networks. These timings, recorded on their journey to the local Internet router and onward to the first-hop server on the Internet, serve as a pivotal foundation for modeling network behavior. The presence of gaps within this time-series data provides unique insight into the occurrences of outages, either within the LAN itself or concerning the router's capacity to maintain Internet connectivity. Utilizing the ICMP protocol's ECHO_REQUEST and ECHO_RESPONSE datagrams ensures the accuracy and reliability of these measurements. In this research, the dataset played a crucial role in simulating delays within the model. The data itsel presented on figure 1.

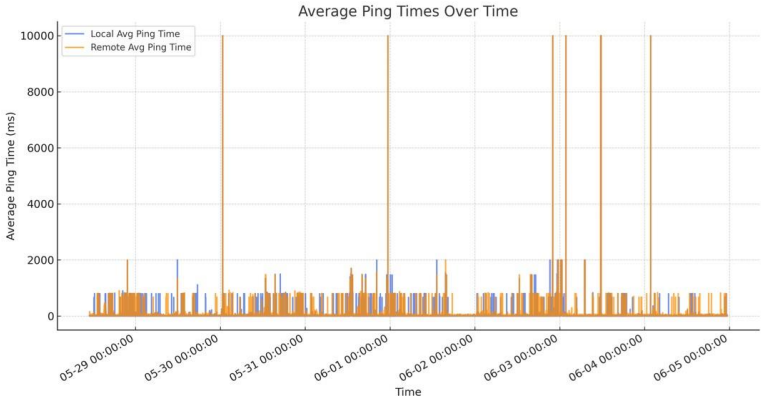

Fig. 1 Filtered average ping times over time

After analyzing the data presented in the figure, one cannot help but say that cleaning is very evidently required over here. So, here comes an updated dataset which is presented in the figure 2, where the outliers are carefully wiped off.

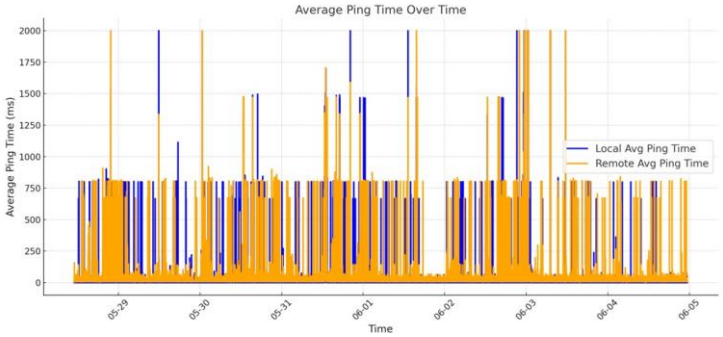

Fig. 2 Filtered average ping times over time

To capture the fundamental characteristics of the data transformation is requred. The figure 3 is a combined illustration using histograms and boxplots, which are used to display distribution of a data set. The first part of this diagram depicts "Local Avg Ping Times Histogram" compared with "Local Avg Ping Times Boxplot"; the second one shows the "Remote Avg Ping Times Histogram" and "Remote Avg Ping Times Boxplot" Histograms pictorially provide the view of frequency data distribution over some interval basis, whereas boxplots give a graphical summary of data quartiles, median, and any kind of outliers.

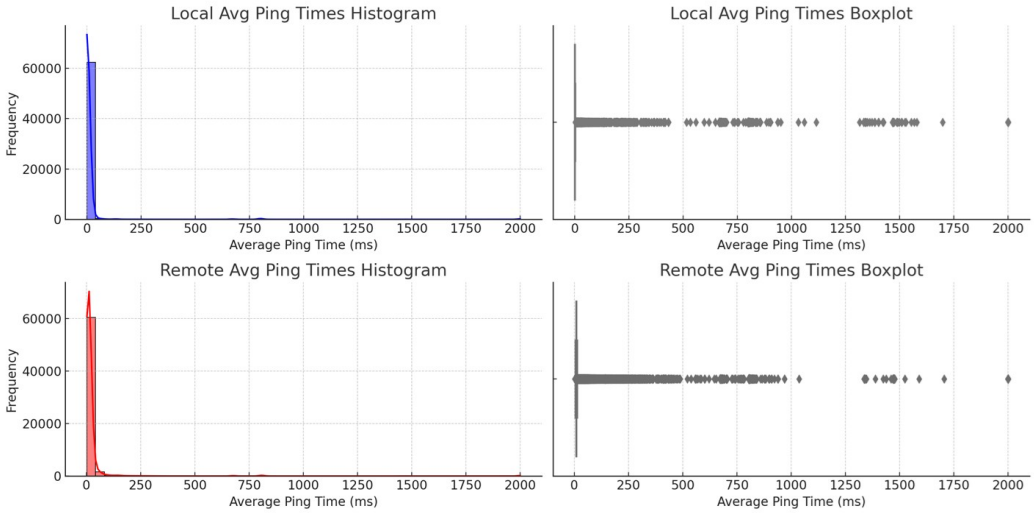

Fig. 3. Comparative analysis of local and remote average ping times: histograms and boxplots

Figure 4 also provides insights, it represents the density estimation of the data's distribution, showing the probability density of the ping times. The width of each violin varies with the frequency of

observations: wider sections of the violin indicate a higher concentration of data points (a higher probability density), while narrower sections indicate fewer observations.

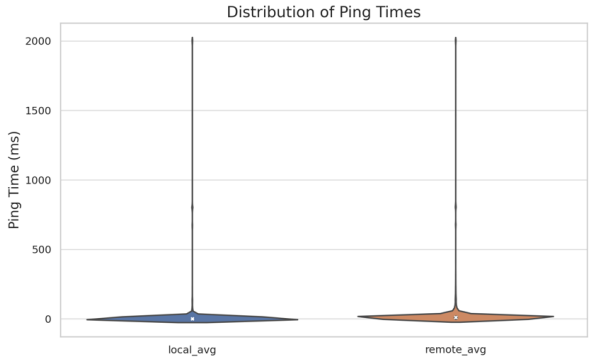

Fig. 4. Violin plots of local vs remote average ping time distribution

After an initial examination, it is clear that the data exhibis a concentrated range of lower ping times for both local and remote connections, with outliers extending to higher values. The density plots suggest a steep peak at lower milliseconds, indicating a rapid drop-off in frequency as ping time increases. Nevertheless, it is still not clear which distribution the data follows; further tests are required.

To model network behavior, we need to understand the distribution that the data follows; thus, conducting a more thorough statistical analysis is required.

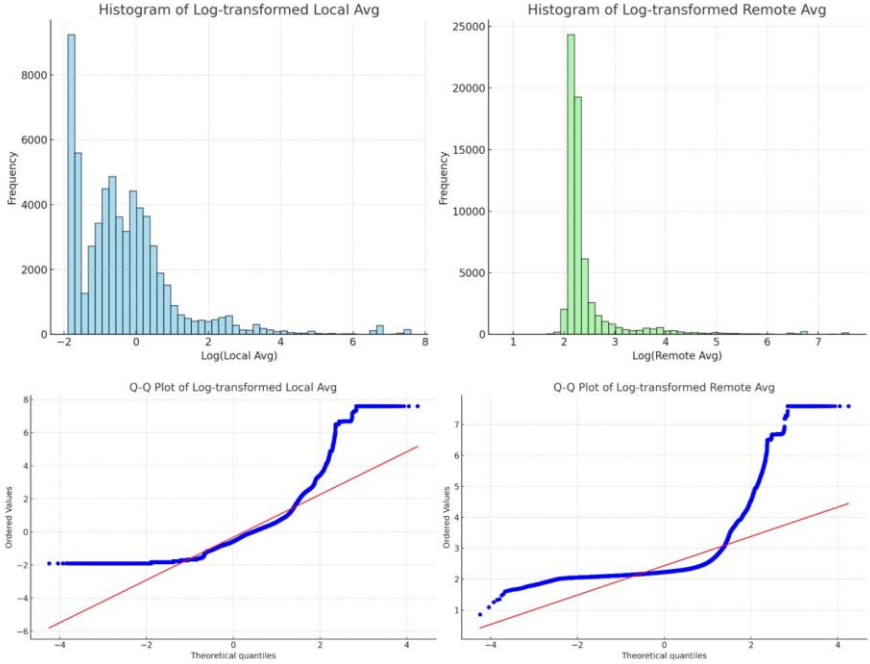

Fig. 5. Comparative statistical analysis of log-transformed local and remote ping times: histograms and quantile-quantile plots

Our analysis commenced with the extraction of network delay data, specifically focusing on the average local (**local_avg**) and remote (**remote_avg**) delays. The dataset comprised measurements from numerous nodes within a distributed network, aiming to capture a comprehensive representation of delay variations. To scrutinize the distribution of these delay metrics, we employed logarithmic transformation (figure 5), a technique that facilitates the comparison of empirical data distributions with the log-normal model.

The transformed data were visualized through histograms and quantile-quantile (Q-Q) plots, with the former illustrating the frequency distribution and the latter assessing the fit against a theoretical normal distribution. Both the **local_avg** and **remote_avg** delay data, upon transformation, exhibited characteristics

reminiscent of a log-normal distribution. Specifically, the histograms revealed a skewness that, once transformed, aligned more closely with the symmetry expected of a normal distribution. Correspondingly, the Q-Q plots further supported this observation, where the data points largely adhered to the linearity indicative of a normal distribution in logarithmic space.

In assessing the normality of data distributions within our analysis, the Shapiro-Wilk test stands as a critical statistical tool. This test evaluates whether a sample comes from a normally distributed population, leveraging the test statistic W, defined by formula 1:

$$W = \frac{\left(\sum_{i=1}^{n} a_i x(i)\right)^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}, \tag{1}$$

where n is the sample size, x_i - denotes the ordered sample values, $\bar{x}$ represents the sample mean, and a_i are coefficients derived from the expected values of the order statistics for a normal distribution.

Parametr W statistic close to 1 suggests that the data are well approximated by a normal distribution, while values significantly lower than 1 indicate deviations from normality.

Upon applying the Shapiro-Wilk test to dataset, specifically examining the avr_local and avr_remote variables, the test yielded W statistics that significantly deviate from the value indicative of a normal distribution. For avr_local, the W statistic suggests a substantial departure from normality, asserting the data's non-normal distribution. Similarly, the avr_remote dataset's W statistic corroborates this finding, further affirming the non-normal character of the distribution.

Despite the statistical tests' stringent rejection of normality in the log-transformed data, the overall analysis suggests a log-normal distribution's viability for approximating network delay in consensus algorithm models. The empirical evidence, particularly the visual analyses, underscores the potential of utilizing log-normal parameters to simulate network delay. This approximation can significantly enhance model accuracy, offering a more realistic representation of network dynamics in distributed systems.

Incorporating a log-normal approximation of network delay into simulation models can provide researchers and practitioners with a robust tool for predicting and analyzing consensus algorithm performance under varied network conditions. Future research may explore refining these approximations and investigating their implications on consensus efficiency and system resilience.

This investigation paves the way for a nuanced understanding of network delay dynamics, advocating for a statistically informed approach to modeling in the realm of distributed consensus algorithms.

4. Simulation model description for evaluating distributed consensus under network fluctuations

To examine the problem of constant leader reelection, the simulation model was built [2]. It consists of 5 nodes linked in cluster where each node has a connection to other nodes (figure 6).

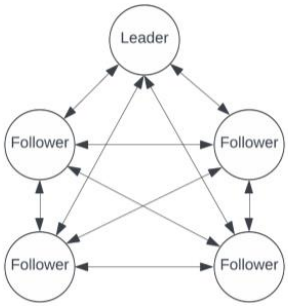

Fig. 6. Proposed model cluster configuration

In the leader-based consensus algorithms (i.e Raft), the heartbeat step consists of signals sent by the leader to all followers at regular intervals [9]. This maintains the leader's authority and prevents new

elections. Heartbeats demonstrate that the leader is operational and aid in synchronizing the cluster without data transfer, this step represented on figure 7. Should a follower not receive a heartbeat within a specified timeframe, it may initiate an election (cluster reconfiguration) to select a new leader. This mechanism is crucial for the cluster's stability and reliability.

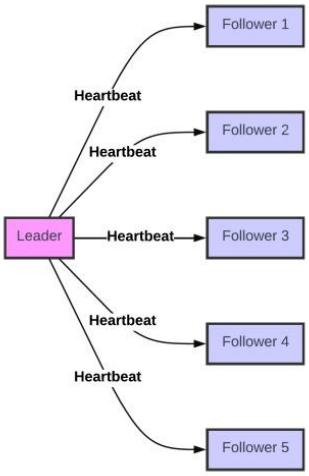

Fig. 7. Leader-node heartbeat communication diagram

The idea is to use a log-normal distribution generator to simulate delays among nodes, launch the heartbeat step of the algorithm, and assess whether reconfiguration is performed, using the law of large numbers (LLN) to determine whether the reconfiguration is triggered, where law of large numbers can be represented by formula 2:

$$X_n = \frac{1}{n}\left(X_1 + ... + X_n\right); X_n \to \mu \text{ as } n \to \infty, \tag{2}$$

where: X_n - average of $X_1 ... X_n$ values; n – number of experiments; μ – actual probability.

In this case considered a log-normal distribution to simulate delay among nodes. This is a realistic choice because network delays often exhibit a log-normal distribution (discussed abow), as they are typically non-negative and can have a long tail, where most of the delays are short but there are occasional long delays. By simulating delays in this way, it is possible to generate a wide range of scenarios to test the robustness of distributed system.

When the LLN applied to this situation, binary outcome is observed, whether reconfiguration is triggered by the consensus algorithm in response to these delays. The LLN will allow to understand the probability of reconfiguration being triggered over a large number of simulated heartbeat steps.

By employing the LLN, over many iterations, the proportion of times that reconfiguration is triggered will approach the expected probability of triggering. This method will give more accurate picture of the algorithm's behavior under 'normal' conditions, with many nodes and interactions. It also helps in assessing the stability of the system; if reconfiguration is consistently triggered under certain simulated conditions, it suggests that the system is sensitive to those conditions.

Additionally, using the LLN in this way can help in the fine-tuning of the algorithm. For instance, if it is found that reconfiguration is triggered too often or not as frequently as required, one could decide to adjust the parameters of the algorithm.

In summary, the law of large numbers is a good choice for assessing binary outcomes such as 'reconfiguration triggered/not triggered' in distributed systems. It provides a solid theoretical foundation for anticipating how the system will behave over time, which is crucial for ensuring the reliability and efficiency of the reconfiguration process.

The final model’s flow depicted on figure 8:

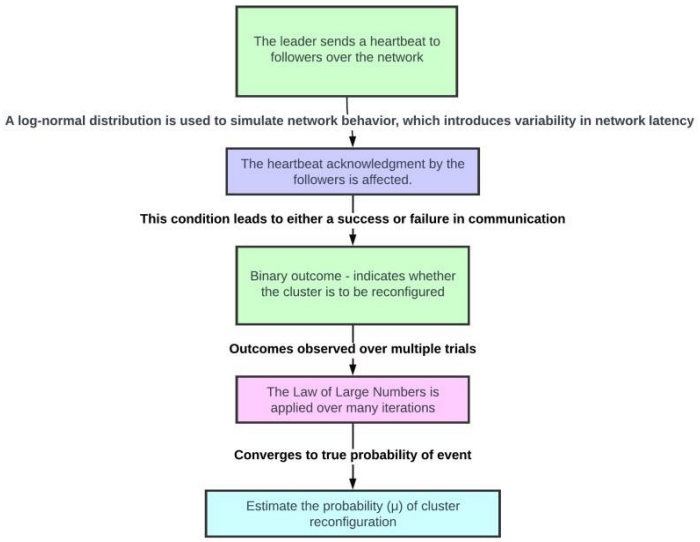

Fig. 8. Flowchart for estimating the probability of cluster reconfiguration due to network latency

The calibration of model parameters plays a pivotal role in the simulation of distributed systems. This discussion begins by emphasizing the importance of selecting appropriate inputs, particularly election timeout and network stability, to accurately assess system performance.

Choosing the election timeout value in distributed systems, particularly in the context of network delays, requires a practical approach that accounts for the variability and unpredictability of network performance. The goal is to set a timeout that is long enough to avoid unnecessary leader elections due to transient network issues but short enough to ensure timely failover in the event of actual leader failures, so this task is specific to the system and network state at the moment and requires constant system administrator attention.

A common recommended approach is to set the election timeout significantly higher than the maximum observed network delay, including a buffer to accommodate variability and unexpected spikes. This buffer ensures that transient network issues do not lead to leader elections, which can destabilize the cluster and impact performance. Having this buffer still not prevents the system from reconfiguration since network stability can change overtime, so for our simulation it is crucial to understand how system works on the edge of its capabilities, to examine this we will set the election timeout to the constant value in our case (3475 milliseconds, it is assumed that cluster is geographically dispersed) and set reconfiguration timeout to 0.9 of network delay max upper limit. In this case min and max limit of delays for network simulation equal to 700ms and 3475ms respectively, and simulation is performed 3000 times to assert model is stable, the result is depicted on picture 9.

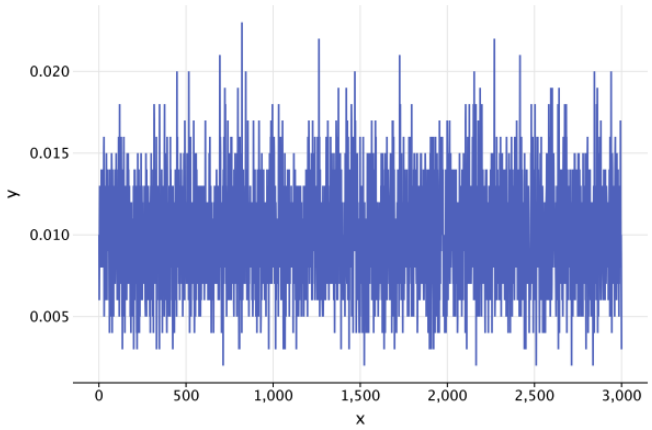

Fig. 9. Assessment of cluster reconfiguration possibility over 3000 experimental trials

As we see no dynamics exist and value converges to 0.01 meaning that there is 1% chance that cluster will reconfigure itself with each heartbeat sent to nodes, so now it is possible to predict how cluster will behave in specific network conditions and cluster parameters.

Conclusion

This study introduced a novel simulation model to assess the impact of network instability on consensus algorithms within distributed clusters. Our investigation highlights the model's ability to emulate various degrees of network instability, including latency fluctuations and connection disruptions, providing valuable insights into the resilience and adaptability of consensus mechanisms under adverse conditions. The findings underscore the importance of incorporating realistic network behavior simulations to enhance the robustness and reliability of distributed systems. Future research should focus on expanding the model's applicability to more complex network scenarios and exploring the integration of additional consensus algorithms. By advancing our understanding of consensus mechanisms' performance in unstable networks, we can contribute to the development of more fault-tolerant distributed systems, ensuring their effectiveness in real-world applications.

Moreover, the study suggests a promising avenue for future work in the development of hybrid consensus mechanisms. These mechanisms could dynamically adjust their operational parameters in response to real-time network conditions, potentially offering a balanced approach to achieving both high performance and reliability in distributed systems. By exploring these adaptive strategies, we can lay the groundwork for creating more robust and efficient distributed systems capable of operating reliably within the increasingly complex and unpredictable landscapes of modern networks.

Additionally, there is a pressing need for comprehensive investigations into how variations in network behavior influence the performance and reliability of consensus algorithms. This study's simulation model presents a pioneering tool for such inquiries, offering the capability to meticulously replicate diverse network conditions and their impacts on consensus processes. Through the utilization of this model, researchers can dissect the nuanced dynamics between network instability and consensus algorithm behavior, uncovering insights vital for the enhancement of distributed systems' resilience. Embracing this avenue of research holds the promise of significantly advancing our understanding and optimization of consensus mechanisms in the face of network variability.

References

[1] *Stanislav Zhuravel, Mykhailo Klymash, Olha Shpur and Orest Lavriv, "Achieving Consistency and Consensus of Distributed Infocommunication Systems", 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), p. 386-389, February 22-26, 2022.*

[2] *Stanislav Zhuravel, "Network Instability Consensus Simulator (NICS): A Tool for Assessing Distributed Systems' Resilience" [Software], GitHub, https://github.com/ZLStas/simulation*

[3] *Stanislav Zhuravel, Olha Shpur and Yulia Pyrih, "Method of achieving consensus in distributed service", vol.2, p. 58-66, November 10, 2022*

[4] *Nazar Peleh, Stanislav Zhuravel, Olha Shpur and Olha Rybytska, "Structured and Unstructured Log Analysis as a Methods to Detect DDoS Attacks in SDN networks", Internet of Things (IoT) and Engineering Applications, Vol 6, Issue 1, 2021*

[5] *S. Zhuravel, S. Dumych and O. Shpur, "Research of data collection and processing methods in distributed information systems", Information and communication technologies, electronic engineering, Vol 1, p. 20-38, November 1, 2021*

[6] *M. Kleppmann, Designing Data-Intensive Applications, O'Reilly UK Ltd., 2017.*

[7] *Muñoz Palacios, Filiberto & Espinoza Quesada, Eduardo Steed & La, Hung & Salazar, Sergio & Commuri, Sesh & Garcia Carrillo, Luis Rodolfo, "Adaptive consensus algorithms for real-time operation of multi-agent systems affected by switching network events". International Journal of Robust and Nonlinear Control. October 20, 2016*

[8] Liu, S., Zhang, R., Liu, C. et al. An improved PBFT consensus algorithm based on grouping and credit grading, 2023, https://doi.org/10.1038/s41598-023-28856-x

[9] Lin Chen, Jing Liao, Naixue Xiong, "Byzantine Fault-Tolerant Consensus Algorithms: A Survey" Electronics, 2023, https://doi.org/10.3390/electronics12183801

[10] .Z. Hussein, M.A. Salama and S.A. El-Rahman, "Evolution of blockchain consensus algorithms: a review on the latest milestones of blockchain consensus algorithms", Cybersecurity, Vol 6, p. 30, 2023, https://doi.org/10.1186/s42400-023-00163-y

[11] K. Venkatesan and S.B Rahayu, "Blockchain security enhancement: an approach towards hybrid consensus algorithms and machine learning techniques", Sci Rep, Vol 14, p. 1149, 2024, https://doi.org/10.1038/s41598-024-51578-7

[12] Faisal Nawab, Mohammad Sadoghi, "Consensus in Data Management: From Distributed Commit to Blockchain", Foundations and Trends in Databases: Vol. 12: No. 4, pp 221-364, 2023, http://dx.doi.org/10.1561/1900000075

[13] Gary Stafford, "LAN network stability: measure response time of a wireless vs. ethernet-based LAN", Kaggle, 2021, https://www.kaggle.com/code/garystafford/network-stability-notebook/input

РОЗРОБКА ІМІТАЦІЙНОЇ МОДЕЛІ МЕРЕЖІ ДЛЯ ОЦІНКИ ЕФЕКТИВНОСТІ РОЗПОДІЛЕНОГО КОНСЕНСУСУ З УРАХУВАННЯМ НЕСТАБІЛЬНОСТІ МЕРЕЖЕВИХ З'ЄДНАНЬ

Станіслав Журавель

Національний університет «Львівська політехніка», вул. С. Бандери, 12, 79013, Львів, Україна

Динамічний і непередбачуваний характер мережевих середовищ створює серйозну проблему для розподілених систем, особливо для тих, які покладаються на консенсусні алгоритми для управління станом і відмовостійкості. Щоб вирішити цю проблему, у статті представляється нова імітаційна модель, призначена для вивчення впливу нестабільних мережевих з'єднань на кластери, які виконують консенсусні алгоритми. Модель створена для імітації різного ступеня нестабільності мережі, включаючи флуктуації затримки та порушення з'єднання, характерні для розподілених систем реального часу. Запропонована нами модель є значним прогресом у моделюванні розподілених мереж. Він використовує складний рівень емуляції мережі, здатний генерувати широкий спектр нестабільних умов мережі. Ядром моделі є симулятор механізму консенсусу з широкими можливостями налаштування, який дозволяє регулювати такі ключові параметри, як інтервали між передаваннями, тайм-аути виборів і частоту втрат повідомлень. Цей рівень конфігурації дає змогу здійснювати комплексний аналіз консенсусної поведінки за різними мережевими сценаріями. У статті зосереджено увагу на методології розробки моделі, деталізовано теоретичні основи та стратегії реалізації, які використовуються для забезпечення реалістичного представлення нестабільності мережі. Завдяки розгортанню цієї моделі дослідники та системні архітектори можуть отримати глибше розуміння стійкості та адаптивності консенсусних алгоритмів. Модель служить інструментом для завчасного виявлення та вирішення потенційних проблем у розподілених системах, сприяючи розробці більш стійких і надійних технологій.

Ключові слова: *алгоритми консенсусу, нестабільність мережі, відмовостійкість, імітаційна модель, розподілені системи*