UDC 004.75 **doi: 10.32620/reks.2023.2.14**

Jaafar AHMED[1], Andrii KARPENKO[2], Olga TARASYUK[3,4], Anatoliy GORBENKO[1,2], Akbar SHEIKH-AKBARI[1]

[1] ***Leeds Beckett University, Leeds, United Kingdom***
[2] ***National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine***
[3] ***Odesa Technological University STEP, Odesa, Ukraine***
[4] ***Newcastle University, Newcastle, United Kingdom***

CONSISTENCY ISSUE AND RELATED TRADE-OFFS IN DISTRIBUTED REPLICATED SYSTEMS AND DATABASES: A REVIEW

Distributed replicated databases play a crucial role in modern computer systems enabling scalable, fault-tolerant, and high-performance data management. However, achieving these qualities requires resolving a number of trade-offs between various properties during system design and operation. This paper reviews trade-offs in distributed replicated databases and provides a survey of recent research papers studying distributed data storage. The paper first discusses a compromise between consistency and latency that appears in distributed replicated data storages and directly follows from CAP and PACELC theorems. Consistency refers to the guarantee that all clients in a distributed system observe the same data at the same time. To ensure strong consistency, distributed systems typically employ coordination mechanisms and synchronization protocols that involve communication and agreement among distributed replicas. These mechanisms introduce additional overhead and latency and can dramatically increase the time taken to complete operations when replicas are globally distributed across the Internet. In addition, we study trade-offs between other system properties including availability, durability, cost, energy consumption, read and write latency, etc. In this paper we also provide a comprehensive review and classification of recent research works in distributed replicated databases. Reviewed papers showcase several major areas of research, ranging from performance evaluation and comparison of various NoSQL databases to suggest new strategies for data replication and putting forward new consistency models. In particular, we observed a shift towards exploring hybrid consistency models of causal consistency and eventual consistency with causal ordering due to their ability to strike a balance between operations ordering guarantees and high performance. Researchers have also proposed various consistency control algorithms and consensus quorum protocols to coordinate distributed replicas. Insights from this review can empower practitioners to make informed decisions in designing and managing distributed data storage systems as well as help identify existing gaps in the body of knowledge and suggest further research directions.

Keywords: *distributed databases; Big Data, NoSQL; replication; consistency; latency; throughput; availability; trade-offs; CAP; PACELC; review.*

Introduction

The exponential growth of data together with the increasing demand for scalability, fault tolerance, and high-performance computing, has driven the rise of distributed databases as a fundamental component of modern computer systems. Internet-scale distributed computer systems and data storages are now extensively used in social media platforms, science and business applications and critical infrastructures.

The importance of distributed data storages and modern NoSQL databases lies in their ability to address the limitations of traditional relational database management systems, which struggle to handle the scale, velocity, and variety of data as required by today's applications [1]. By sharing and replicating data across multiple nodes, distributed databases provide inherent advantages such as enhanced performance and scalability, improved availability and fault tolerance.

However, achieving these benefits involves making trade-offs in various aspects, including consistency, availability, latency, cost and other system properties. Deep understanding of these trade-offs enables system engineers, developers, and users to make informed decisions and find the right balance to meet their specific requirements and is crucial for resource-efficient system design.

The paper is aimed at analysing trade-offs in modern distributed systems and replicated data storages and reviewing research publications that study the consistency and other properties of such systems. The rest of the paper is organised as follows. In the next section we discuss CAP and PACELC theorems and their implications in distributed systems. Section 2 explores main trade-offs between the various properties of distributed replicated databases. Section 3 provides an overview of recent research works studying distributed data storages, analyses keywords and identifies major areas of interests and research efforts. The last section concludes our survey.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

1. Data Replication. CAP and PACELC Theorems

1.1. Data Replication

Data replication is a database scale-out approach where the same data is replicated across multiple nodes, i.e. replicas. It is a vital technique used to build large-scale distributed applications of high performance.

Data replication primarily improves system availability and performance. If one of the database nodes fails, the system will continue to work because there are other nodes available. The system performance is improved by load balancing (i.e. sharing the workload among available nodes) and distributing replicas close to end users to minimize latency. However, replication increases implementation complexity and rises multiple inter-related trade-offs between different system properties.

1.1. CAP Theorem

The CAP theorem [2], first introduced in 1998-1999, establishes a fundamental trade-off between three properties of distributed replicated systems: Consistency, Availability, and Partition tolerance. According to the theorem, it is impossible to simultaneously guarantee all three properties in such systems, allowing for the preservation of only two out of the three. Gilbert and Lynch in [3] consider the CAP theorem as a specific instance of a broader trade-off between consistency and availability that exists in unreliable distributed systems, where updates propagate over time. When a distributed database encounters a partition (for example, when one of the replicas rejects a user request or fails to respond within a given timeout), it must decide whether to return a possibly stale result to the client, sacrificing consistency or to report service unavailability. Depending on this decision, the CAP theorem specifies the following three types of systems:

– ***CA***: traditional ACID-oriented databases (Oracle, MySQL, PostgreSQL, etc.) which preserve Consistency and Availability, but are not suitable for efficient data distribution and cannot tolerate system partitioning;

– ***AP***: distributed databases such as Riak, Voldemort, Cassandra, Dynamo, CouchDB relaxing Consistency in favour of Availability and Partition tolerance;

– ***CP***: NoSQL databases such as MongoDB, HBase, BigTable, Redis, MemcacheDB which give up availability but preserve Consistency when Partitioned.

1.2. PACELC Theorem

The PACELC theorem [4] is a further development of CAP which suggests that that in the case of *partitioning* (P) of a distributed data storage, one has to choose between *availability* (A) and *consistency* (C), *else* (E) in the absence of partitions the replicated database faces a trade-off between *latency* (L) and *consistency* (C). There could be four types of distributed databases as follows from PACELC (some distributed databases such as Couchbase, DynamoDB can be configured to be either PC/EL or PC/EC):

– ***PC/EC***: ACID databases (MySQL Cluster, PostgreSQL, Megastore, VoltDB/H-Store, etc.) and NoSQL databases (HBase, BigTable);

– ***PC/EL***: PNUTS, Couchbase, DynamoDB, FaunaDB;

– ***PA/EC***: MongoDB, Aerospike;

– ***PA/EL***: Cassandra, Dynamo, Cosmos DB, Riak.

2. Trade-offs in Distributed Replicated Databases

Though CAP and PACELC theorems are helpful in understanding a key trade-off between consistency and availability/latency, they do not consider other important compromises discussed below.

2.1. Availability vs Performance

Replicating data across multiple nodes increases data availability. On the one hand, it also enables load balancing to improve system performance. On the other, increased system complexity and overheads related to the need for replica synchronization and coordination introduce delays and impact system latency. This impact considerably depends on the replication model used and consistency settings. Another important factor balancing availability versus performance is the application timeout [5]. Timeout serves as the major error detection mechanism in distributed systems. If it is less than the typical response time, a system will likely enter a partition mode more often [6]. If the timeout is too long, the system will be too slow to handle exceptions and inefficient in error-recovery.

2.2. Latency vs Consistency

Strong consistency ensures that a read always reflects the most recent write and all nodes store the same version of data at all times. It is always desirable to maintain data consistency. However, the strong consistency cannot be efficiently achieved in distributed (especially of the large scale) replicated systems without affecting system latency. Strong consistency is based on synchronous updates which are controlled by consensus protocols or coordination mechanisms such as Paxos or two-phase commits. They require multiple communication and coordination among replicas resulting in increased latency.

The more nodes involved in the coordination and the larger the distance between them, the higher the latency impact. Moreover, strong consistency often relies on locking mechanisms to prevent concurrent conflicting operations. This can increase latency as transactions should wait for locks to be released.

2.3. Availability and Performance vs Cost and Energy Consumption

A higher replication factor coupled with a relaxed consistency model can considerably increase database performance, fault-tolerance and availability. However, increased redundancy incurs higher storage costs and cost of maintenance and management. Besides, stronger consistency model involves coordination of responses from multiple replicas which brings interaction overheads additional to propagating updates over the network. Consequently, these overheads lead to higher energy consumption by multiple database nodes and networking equipment.

2.4. Consistency vs Availability

If one of the replicas is down or unreachable and does not respond within the specified timeout, the system becomes partitioned. In this case the database can sacrifice availability to guarantee consistency or proceed and return a potentially stale data to a client, as it is described by the CAP theorem. Prioritizing high availability can lead to the necessity of relaxing the consistency level and using a weaker consistency model.

2.5. Read Latency vs Write Latency

In a traditional master–slave replicated system reads become faster because only one (the nearest to a client) replica must be read. At the same time writes become slower due to all (in case of strong consistency) replicas need to return the status of the update before confirming its success to a client. Read and write latencies can be balanced by shifting consistency management from reads to writes. For example, Cassandra NoSQL database maintains a tuneable consistency level for individual read and write operations. This consistency level specifies the number of replicas that should respond for a write (W) or read (R) request to be considered complete. If the sum of R and W is larger than the total number of replicas (i.e. the replication factor N) then the strong consistency is guaranteed: R +W > N. Balancing R and W allows trading-off read and write performance while preserving strong consistency.

2.6. Durability vs Performance

Durability refers to the ability of a database to ensure that once data updates are committed, they will persist even in the event of failures or crashes. Ensuring durability usually involves synchronous writes to durable storage (i.e. the hard disk). However, slow disk I/O operations increase latency and impact system performance, especially for write-intensive workloads. To reduce write latency many NoSQL databases store data in memory, periodically flushing it and transaction/commit logs to disk. This improves overall system performance but creates a so called ‘data loss window’ [7], leading to a trade-off between durability and performance.

3. A Survey of Recent Research Works Studying Distributed Data Storages

Recently, Big Data storages and distributed databases have gained significant interest from both developers and the research community. Tag clouds built using titles (Fig. 1) and keywords (Fig. 2) from the recent research publications depict a visual taxonomy of key topics of researchers’ interests in the field.

Consistency (consistency models, weak and eventual consistency) and *replication* are shown to be the most important and dominant research topics. In turn, data redundancy causes development of *quorum and consensus protocols* and *algorithms for adaptive replications* and *optimization of replica placement*. Other important areas of research include evaluation and enhancement of system latency, availability, and reliability.

Fig. 1. A word cloud built using titles of recent publications in the domain of distributed data storages

Fig. 2. A word cloud built using keywords from papers in the domain of distributed data storages

Fig. 3 further classifies the studied publications in two major groups: those that report results of various types of analysis of existing distributed data storages and those that provide solutions to enhance them.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

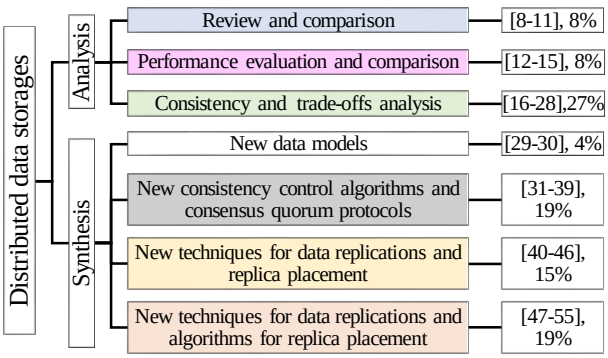

Fig. 3. Classification of research publications studying distributed data storages

The first group (see Table 1) encompasses research works reviewing and comparing different distributed databases [8-11], evaluating their performance [12-15], and studying trade-offs between data consistency and other system properties [16-28] the most popular of which is a trade-off between consistency and latency. It is worth noting that other important trade-offs (e.g. between durability and performance; energy consumption and availability) have received much less attention, which determines the need for further research in this area. The analyzed works suggest that Cassandra and MongoDB are two the most popular NoSQL databases that attract researchers the most, and that YCSB (Yahoo Cloud Serving Benchmark) is used as the default standard for their performance evaluation. Other frequently studied databases are Hbase, Redis, Couchbase, BigTable, DynamoDB, Riak and Cosmos DB. These and other distributed data storages support various consistency models that are difficult to match and to verify in all possible application scenarios. In [28] researchers propose *consistency oracles* which mimic behavior of a distributed system and are used to formally specify the consistency model of that system and predict possible data state under a given consistency model. Unfortunately, many research publication comparing performance of various NoSQL databases do not take into account the impact of the used consistency model, consistency settings and the replication factor on the reported results of performance evaluation, which creates a certain gap in the body of knowledge.

The second large group of recent research activities (see Table 2) puts forward (i) new data models suitable for efficient distributed across multiple nodes [29-30]; (ii) new consistency models [40-46], (iii) consistency control algorithms and consensus quorum protocols [31-39], and (iv) techniques for data replications and strategies for replica placement [47-55].

A considerable number of recent publications focus on developing *adaptive replication techniques* [32, 34] and proposing *optimal replica placement* strategies [52-54] in

Table 1

Research publications on comparison and performance evaluation of NoSQL databases; consistency and trade-offs analysis in distributed systems

Research domain	Reference	Year	Distributed data storage											Database property								Cloud deployment	Evaluation technique
			ACID	BASE																			
			General ACID	General BASE	Cassandra	MongoDB	Hbase	JBossDataGrid	Couchbase	BigTable	DynamoDB	Riak	Cosmos DB	Others	Consistency	Latency	Throughput	Availability	Durability	Fault-tolerance	Storage space	Energy consumption		
Review and comparison	[8]	2016	1												1									review
	[9]	2017			1					1	1				1	1	1	1					1	review
	[10]	2021			1	1																		review
	[11]	2022		1																			1	review
Performance evaluation and comparison	[12]	2019			1	1	1									1	1							benchmarking
	[13]	2020							1	1						1	1						1	benchmarking
	[14]	2021			1	1										1	1						1	YCSB benchmarking
	[15]	2017					1	1				1		1		1	1							experimental evaluation
Consistency and trade-offs analysis	[16]	2021													1									review
	[17]	2019			1										1	1		1						analytical modeling
	[18]	2020		1	1	1							1		1	1		1	1	1		1		theoretical
	[19]	2019		1											1	1		1						experimental evaluation; analytical modelling
	[20]	2012		1											1	1		1		1				theoretical
	[21]	2019		1											1			1		1				theoretical
	[22]	2015		1											1								1	experimental evaluation
	[23]	2017			1										1	1	1						1	theoretical; YCSB benchmarking
	[24]	2020		1											1	1	1			1				theoretical
	[25]	2020			1										1	1		1					1	YCSB benchmarking
	[26]	2014			1										1	1							1	YCSB benchmarking
	[27]	2022			1										1	1	1							YCSB benchmarking
	[28]	2017		1											1									theoretical

Table 2

Research publications on new consistency models, quorum protocols, data replications and replica placement

Research domain	Reference	Year	Distributed data storage									Database property								Cloud deployment	Evaluation technique
			ACID	BASE																	
			PostgreSQL	General BASE	Cassandra	MongoDB	Redis	Apache Accumulo	RO-DCOP	SDN controller	HyFlow DTM	Consistency	Latency	Throughput	Availability	Fault-tolerance	Storage space	Energy consumption	SLA		
New data models	[29]	2016					1						1	1							implementation and testing
	[30]	2017						1					1				1				implementation and testing
New consistency control algorithms and consensus quorum protocols	[31]	2019		1								1	1	1	1	1				1	implementation and testing
	[32]	2018							1			1			1	1					implementation and testing
	[33]	2020		1								1	1	1		1					simulation in Akka
	[34]	2020		1								1	1			1				1	implementation; benchmarking (TPC-C)
	[35]	2015		1								1			1	1					theoretical
	[36]	2020		1								1	1	1		1				1	simulation
	[37]	2017		1								1				1	1				modelling and benchmarking
	[38]	2018		1	1							1	1	1		1				1	implementation; benchmarking (YCSB)
	[39]	2021		1								1	1	1		1					theoretical and implementation
New consistency models	[40]	2018								1		1		1					1		implementation and failure simulation
	[41]	2017								1		1	1						1		implementation and failure simulation
	[42]	2016		1											1	1			1		theoretical
	[43]	2020		1								1	1	1						1	prototyping on Alibaba cloud
	[44]	2020			1							1	1	1						1	implementation; YCSB benchmarking
	[45]	2021				1						1	1			1					implementation and benchmarking
	[46]	2018		1									1								simulation in CPN
New techniques for data replications and replica placement	[47]	2017									1		1			1					simulation with ReDstm
	[48]	2022	1									1		1	1	1					implementation; benchmarking (TPC-C, YCSB, CHB)
	[49]	2021	1									1		1		1					implementation; benchmarking (TPC-C, YCSB)
	[50]	2020		1								1	1	1		1				1	simulation, implementation and testing
	[51]	2020		1								1	1		1	1		1		1	simulation in CloudSim
	[52]	2020		1									1			1				1	simulation using MSR Cambridge Traces and Facebook Friendships Dataset
	[53]	2020		1								1	1			1				1	simulation in OptorSim
	[54]	2019			1							1	1			1					benchmarking (YCSB)
	[55]	2017		1								1				1					formal verification

large-scale geo-spatial distributed systems, cloud environments and edge applications. Because strong consistency in distributed replicated systems causes significant delays and impacts system performance, other researchers are introducing mechanisms to provide certain guarantees (e.g. operations *causality*) for systems with relaxed consistency [37, 38, 43, 44]. *Causal consistency* is a weak consistency model which, nevertheless, preserves the order of read and write/update operations for all clients (which is crucial for many business- and mission-critical application) while accepting some level of data staleness. Furthermore, in [45] authors put forward a concept of *fuzzy consistency model*.

Yet the most promising approach seems to be to balance data consistency and system performance in line with the idea of *adaptive consistency* developed in [35, 40, 41, 56]. In this regard, timeouts settings seem to be an important part of the adaptability mechanism playing also a role of a major error detection mechanism in distributed systems and databases [25, 56].

However, it seems that the problem of optimal timeout settings and its effect on system consistency, availability and performance is less studied.

Conclusions

In recent years, there have been significant advances in development and research interests in the field of distributed data storages. These efforts are driven by the increasing demands for scalable, fault-tolerant, and high-performance data management systems to deal with high volume, velocity, and variety of big data. Consistency and replication as one of the most crucial aspects of distributed databases have received considerable attention from researchers aiming to tackle the challenges associated with data integrity, availability, and performance.

Through our comprehensive review several key insights and trends have been identified. First, there is a growing emphasis on analysing trade-offs and achieving a balance between consistency, performance and other system properties.

Researchers have proposed various approaches such as relaxed consistency models, concurrency control mechanisms, and adaptive consistency protocols to mitigate the latency and coordination overhead associated with the strong consistency.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Second, we observed a shift towards exploring models going beyond the traditional strong and eventual consistency. Causal consistency and eventual consistency with causal ordering have gained attention due to their ability to strike a balance between operations ordering guarantees and high performance. These new models enable applications to maintain causal relationships between read/write requests while relaxing the strict requirements of linearizability or serializability.

Another prominent theme in recent research works is utilizing machine learning (ML) algorithms to optimize data replication strategies and dynamically adapt consistency levels based on workload patterns. This intersection of ML and distributed databases opens up new avenues for intelligent and adaptive data management in distributed environments. Furthermore, we observed an increased focus on consistency and replication in specific domains such as edge computing, IoT, and large-scale geo-spatial applications. These domains present unique challenges in terms of limited resources, intermittent connectivity, and decentralized trust models. Overall, the reviewed publications reflect continuous evolution of distributed databases, consistency models and replication strategies and highlight need for further innovations and research activities.

Contributions of authors: conceptualization – **Anatoliy Gorbenko, Akbar Sheikh-Akbari**; methodology – **Anatoliy Gorbenko**, **Akbar Sheikh-Akbari**; literature review and analysis – **all authors**; visualization – **Jaafar Ahmed**, **Andrii Karpenko**, **Olga Tarasyuk**; writing: original draft preparation – **Jaafar Ahmed**; writing: review and editing – **Andrii Karpenko**, **Olga Tarasyuk, Anatoliy Gorbenko**, **Akbar Sheikh-Akbari**.

All the authors have read and agreed to the published version of the manuscript.

Acknowledgment

This research is partly supported by the British Academy's Researchers at Risk Fellowships Programme (RaR\100289).

References

1. Ormandjieva, O., Omidbakhsh, M. & Trudel, S. Measuring the 3V's of Big Data: A Rigorous Approach. *Int. Workshop on Software Measurement (IWSM) and Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 2020. Available at: www.iwsm-mensura.org/wp-content/uploads/2020/10/paper5.pdf (accessed Jan. 17, 2023).

2. Brewer, E. Towards Robust Distributed Systems. *ACM Symposium on Principles of Distributed Computing*, 2000. DOI: 10.1145/343477.343502.

3. Gilbert, S. & Lynch, N. Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. *ACM SIGACT News*, 2002, vol. 33, no. 2, pp. 51-59. DOI: 10.1145/564585.564601.

4. Abadi, D. Consistency Tradeoffs in Modern Distributed Database System Design. *IEEE Computer*, 2012, vol. 45, no. 2, pp. 37-42. DOI: 10.1109/MC.2012.33.

5. Gorbenko, A. & Romanovsky, A. Time-outing Internet Services. *IEEE Security & Privacy*, 2013, vol. 11, no. 2, pp. 68-71. DOI: 10.1109/MSP.2013.43.

6. Gorbenko, A., Romanovsky, A. & Tarasyuk, O. Fault tolerant internet computing: Benchmarking and modelling trade-offs between availability, latency and consistency. *J. Netw. Comput. Appl.*, 2019, vol. 146, article no. 102412. DOI: 10.1016/j.jnca.2019.102412.

7. Alagappan, R., Ganesan, A., Patel, Y., Pillai, T. S., Arpaci-Dusseau A. C. & Arpaci-Dusseau, R. H. Correlated Crash Vulnerabilities. *USENIX Symposium on Operating Systems Design and Implementation*, 2016. Available at: www.usenix.org/system/files/conference/osdi16/osdi16-alagappan.pdf (accessed Jan. 17, 2023).

8. Abbas, Q., Shafiq, H., Ahmad, I. & Tharanidharan, S. Concurrency control in distributed database system. *Int. Conf. on Computer Communication and Informatics (ICCCI)*, 2016, pp. 1-4. DOI: 10.1109/ICCCI.2016.7479987.

9. S. Kalid, S., Syed, A., Mohammad, A. & Halgamuge M. N. Big-data NoSQL databases: A comparison and analysis of Big-Table, DynamoDB, and Cassandra. *IEEE 2nd Int. Conf. on Big Data Analysis (ICBDA)*, 2017, pp. 89-93. DOI: 10.1109/ICBDA.2017.8078782.

10. Anusha, K., Rajesh, N., Kavitha, M. & Ravinder, N. Comparative Study of MongoDB vs Cassandra in big data analytics. *5th Int. Conf. on Computing Methodologies and Communication (ICCMC)*, 2021, pp. 1831-1835. DOI: 10.1109/ICCMC51019.2021.9418441.

11. Sandhu, A. K., Big Data with Cloud Computing: Discussions and Challenges. *Big Data Mining and Analytics*, 2022, vol. 5, iss. 1, pp. 32-40. DOI: 10.26599/BDMA.2021.9020016.

12. Pramukantoro, E. S., Kartikasari, D. P. & Siregar, R. A. Performance Evaluation of MongoDB, Cassandra, and HBase for Heterogenous IoT Data Storage. *Int. Conf. on Applied Information Technology and Innovation (ICAITI)*, 2019, pp. 203-206. DOI: 10.1109/ICAITI48442.2019.8982159.

13. Iurian, C.-M., Ivanciu, I.-A. & Dobrota, V. Couchbase Server in Microsoft Azure Cloud: A Docker Container Approach. *International Symposium on Electronics and Telecommunications (ISETC)*, 2020, pp. 1-4. DOI: 10.1109/ISETC50328.2020.9301052.

14. Araujo, J. M. A., de Moura, A. C. E., da Silva, S. L. B., Holanda, M., Ribeiro, E. de O. & da Silva, G. L. Comparative Performance Analysis of NoSQL Cassandra and MongoDB Databases. *16th Iberian Conference on Information Systems and Technologies (CISTI)*, 2021, pp. 1-6. DOI: 10.23919/CISTI52073.2021.9476319.

15. Misaki, M., Tsuda, T., Inoue, S., Sato, S., Kayahara, A. & Imai, S.-I. Distributed Database and Application Architecture for Big Data Solutions. *IEEE Trans. Semicond. Manuf.*, 2017, vol. 30, no. 4, pp. 328-332. DOI: 10.1109/TSM.2017.2750183.

16. Naik, N. Comprehending Concurrency and Consistency in Distributed Systems. *IEEE International Symposium on Systems Engineering (ISSE)*, 2021, pp. 1-6. DOI: 10.1109/ISSE51541.2021.9582518.

17. Gomes, B., Borba, E., Tavares, E. & Junior, M. N. de O. Performability Model for Assessing NoSQL DBMS Consistency. *IEEE International Systems Conference (SysCon)*, 2019, pp. 1-6. DOI: 10.1109/SYSCON.2019.8836757.

18. Gorbenko, A., Karpenko, A. & Tarasyuk, O., Analysis of Trade-offs in Fault-Tolerant Distributed Computing and Replicated Databases. *IEEE 11th Int. Conf. on Dependable Systems, Services and Technologies (DESSERT)*, 2020, pp. 1-6. DOI: 10.1109/DESSERT50317.2020.9125078.

19. Ductor S. & Guessoum, Z. A coordination mechanism to replicate large-scale multi-agent systems. *Int. Conf. on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 130-136. DOI: 10.1145/3194133.3194154.

20. Gilbert S. & Lynch, N. Perspectives on the CAP Theorem. *Computer*, 2012, vol. 45, no. 2, pp. 30-36. DOI: 10.1109/MC.2011.389.

21. Muñoz-Escoí, F. D., de Juan-Marín, R., García-Escrivá, J.-R., González de Mendívil, J. R. & Bernabéu-Aubán, J. M. CAP Theorem: Revision of Its Related Consistency Models. *Comput. J.*, 2019, vol. 62, no. 6, pp. 943-960. DOI: 10.1093/comjnl/bxy142.

22. Xhafa, F., Naranjo, V., Barolli, L. & Takizawa, M. On Streaming Consistency of Big Data Stream Processing in Heterogenous Clutsers. *18th Int. Conf. on Network-Based Information Systems*, 2015, pp. 476-482. DOI: 10.1109/NBiS.2015.122.

23. Huang, X., Wang, J., Yu, P. S., Bai, J. & Zhang, J. An experimental study on tuning the consistency of NoSQL systems: An Experimental Study on Tuning the Consistency of NoSQL Systems. *Concurr. Comput. Pract. Exp.*, 2017, vol. 29, no. 12, article no. e4129. DOI: 10.1002/cpe.4129.

24. Tomforde S. & Gruhl, C. Fairness, Performance, and Robustness: Is There a CAP Theorem for Self-adaptive and Self-organising Systems? *IEEE Int. Conf. on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, 2020, pp. 54-59. DOI: 10.1109/ACSOS-C51401.2020.00029.

25. Gorbenko, A. & Tarasyuk, O. Exploring Timeout as a Performance and Availability Factor of Distributed Replicated Database Systems. *Radioelectron. Comput. Syst.*, 2020, no. 4, pp. 98-105. DOI: 10.32620/reks.2020.4.09.

26. Kumar, S. P., Lefebvre, S., Chiky, R. & Soudan, E. G. Evaluating consistency on the fly using YCSB. *International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM)*, 2014, pp. 1-6. DOI: 10.1109/IWCIM.2014.7008801.

27. Vyas, K. & Jat, P. M. Study of Consistency and Performance Trade-Off in Cassandra. *Computer Science & Technology*, 2022, vol. 12, no. 19, pp. 61-77. DOI: 10.5121/csit.2022.121907.

28. Kim, B. H., Oh, S. & Lie, D. Consistency Oracles: Towards an Interactive and Flexible Consistency Model Specification. *16th Workshop on Hot Topics in Operating Systems*, 2017, pp. 82-87. DOI: 10.1145/3102980.3102994.

29. Hutchison, B., Kepner, J., Gadepally, V. & Howe, B. From NoSQL Accumulo to NewSQL Graphulo: Design and utility of graph algorithms inside a BigTable database. *IEEE High Performance Extreme Computing Conference (HPEC)*, 2016, pp. 1-9. DOI: 10.1109/HPEC.2016.7761577.

30. Cao, C., Wang, W., Zhang, Y. & Ma, X. Leveraging Column Family to Improve Multidimensional Query Performance in HBase. *IEEE 10th Int. Conf. on Cloud Computing (CLOUD)*, 2017, pp. 106-113. DOI: 10.1109/CLOUD.2017.22.

31. Sun, B., Zhang, G. & Gao, S. Data Management across Geographically Distributed Autonomous Systems: Architecture, Implementation, and Performance Evaluation. *IEEE Int. Conf. on High Performance Computing and Communications; (HPCC*), 2019, pp. 2284-2292. DOI: 10.1109/HPCC/SmartCity/DSS.2019.00317.

32. Ductor, S. & Guessoum, Z. A coordination mechanism to replicate large-scale multi-agent systems. *13th Int. Conf. on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 130-136. DOI: 10.1145/3194133.3194154.

33. Zhao X. & Haller, P. Replicated data types that unify eventual consistency and observable atomic consistency. *J. Log. Algebr. Methods Program.*, 2020, vol. 114, article no. 100561. DOI: 10.1016/j.jlamp.2020.100561.

34. Fetai, I., Stiemer, A. & Schuldt, H. QuAD: A quorum protocol for adaptive data management in the cloud. *IEEE Int. Conf. on Big Data (Big Data)*, 2017, pp. 405-414. DOI: 10.1109/BigData.2017.8257952.

35. Pankowski, T. Consistency and Availability of Data in Replicated NoSQL Databases. *10th Int. Conf. on Evaluation of Novel Approaches to Software Engineering*, 2015, pp. 102-109. DOI: 10.5220/0005368101020109.

36. Gu, S., Wang, Y., Wang, Y., Zhang, Q. & Qin, X. Grouping-Based Consistency Protocol Design for End-Edge-Cloud Hierarchical Storage System. *IEEE Access*, 2020, vol. 8, pp. 8959-8973. DOI: 10.1109/ACCESS.2020.2964626.

37. Hsu, T.-Y., Kshemkalyani, A. D. & Shen, M. Causal consistency algorithms for partially replicated and fully replicated systems. *Future Gener. Comput. Syst.*, 2018, vol. 86, pp. 1118-1133. DOI: 10.1016/j.future.2017.04.044.

38. Fouto, P., Leitao, J. & Preguica, N. Practical and Fast Causal Consistent Partial Geo-Replication. *IEEE 17th International Symposium on Network Computing and Applications (NCA)*, 2018, pp. 1-10. DOI: 10.1109/NCA.2018.8548067.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

39. Maneas, S., Chondros, N., Diamantopoulos, P., Patsonakis, C. & Roussopoulos, M. On achieving interactive consistency in real-world distributed systems. *J. Parallel Distrib. Comput.*, 2021, vol. 147, pp. 220-235. DOI: 10.1016/j.jpdc.2020.09.010.

40. Bannour, B., Souihi, S. & Mellouk, A. Adaptive State Consistency for Distributed ONOS Controllers. *IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1-7. DOI: 10.1109/GLOCOM.2018.8647168.

41. Sakic, E., Sardis, F., Guck, J. W. & Kellerer, W. Towards adaptive state consistency in distributed SDN control plane. *IEEE Int. Conf. on Communications (ICC)*, 2017, pp. 1-7. DOI: 10.1109/ICC.2017.7997164.

42. de Souza, R. H., Flores, P. A., Dantas, M. A. R. & Siqueira, F. Architectural recovering model for Distributed Databases: A reliability, availability and serviceability approach. *IEEE Symposium on Computers and Communication (ISCC)*, 2016, pp. 575-580. DOI: 10.1109/ISCC.2016.7543799.

43. Tian, I. & Pang, Y. Adjoin: A causal consistency model based on the adjacency list in a distributed system. *Concurr. Comput. Pract. Exp.*, 2020, vol. 32, no. 22. DOI: 10.1002/cpe.5835.

44. Nejati Sharif Aldin, B., Deldari, H., Moattar, M. H. & Razavi Ghods, M. Strict Timed Causal Consistency as a Hybrid Consistency Model in the Cloud Environment. *Future Gener. Comput. Syst.*, 2019, vol. 105, pp. 259-274. DOI: 10.1016/j.future.2019.11.038.

45. Khalfi, B., de Runz, C., Faiz, S. & Akdag, H. A New Methodology for Storing Consistent Fuzzy Geospatial Data in Big Data Environment. *IEEE Trans. Big Data*, 2021, vol. 7, no. 2, pp. 468-482. DOI: 10.1109/TBDATA.2017.2725904.

46. Abbaszadeh, M. Weak Consistency Model in Distributed Systems Using Hierarchical Colored Petri Net. *J. Comput.*, 2018 pp. 236-243. DOI: 10.17706/jcp.13.2.236-243.

47. Lima, D., Miranda, H. & Taiani, F. Simulation of partial replication in Distributed Transactional Memory. *Wireless Days, Porto, Portugal*, 2017, pp. 54-59. DOI: 10.1109/WD.2017.7918115.

48. Georgiou, M. A., Paphitis, A., Sirivianos, M. & Herodotou, H. Hihooi: A Database Replication Middleware for Scaling Transactional Databases Consistently. *IEEE Trans. Knowl. Data Eng.*, 2022, vol. 34, no. 2, pp. 691-707. DOI: 10.1109/TKDE.2020.2987560.

49. Georgiou, M. A., Panayiotou, M., Odysseos, L., Paphitis, A., Sirivianos, M. & Herodotou, H. Attaining Workload Scalability and Strong Consistency for Replicated Databases with Hihooi. *Int. Conf. on Management of Data*, 2021, pp. 2721-2725. DOI: 10.1145/3448016.3452746.

50. Guo, I., Li, C. & Luo, Y. Fast replica recovery and adaptive consistency preservation for edge cloud system. *Soft Comput.*, 2020, vol. 24, no. 19, pp. 14943-14964. DOI: 10.1007/s00500-020-04847-2.

51. Mansouri, N., Mohammad Hasani Zade, B. & Javidi, M. M. A multi-objective optimized replication using fuzzy based self-defense algorithm for cloud computing. *J. Netw. Comput. Appl.*, 2020, vol. 171, article no. 102811. DOI: 10.1016/j.jnca.2020.102811.

52. Liu, I., Peng, J., Wang, J., Liu, W., Huang, Z. & Pan, J. Scalable and Adaptive Data Replica Placement for Geo-Distributed Cloud Storages. *IEEE Trans. Parallel Distrib. Syst.*, 2020, vol. 31, no. 7, pp. 1575-1587. DOI: 10.1109/TPDS.2020.2968321.

53. Sun, S., Wang, X. & Zuo, F. RPCC: A Replica Placement Method to Alleviate the Replica Consistency under Dynamic Cloud. *Int. Conf.s on Internet of Things (iThings)*, 2020, pp. 729-734. DOI: 10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics50389.2020.00126.

54. Nwe, T., Yee, T. T., Htoon, E. C. & Nakamura, J., A Consistent Replica Selection Approach for Distributed Key-Value Storage System. *Int. Conf. on Advanced Information Technologies (ICAIT)*, 2019, pp. 114-119. DOI: 10.1109/AITC.2019.8921008.

55. Gomes, V. B. F., Kleppmann, M., Mulligan, D. P. & Beresford, A. R. Verifying Strong Eventual Consistency in Distributed Systems. *Proc. ACM Program. Lang.*, 2017, vol. 1, no. OOPSLA, pp. 1-28. DOI: 10.1145/3133933.

56. Dai, T., He, J., Gu, X. & Lu, S. Understanding Real-World Timeout Problems in Cloud Server Systems, *IEEE Int. Conf. on Cloud Engineering (IC2E)*, 2018, pp. 1-11. DOI: 10.1109/IC2E.2018.00022.

Received 04.03.2023, Accepted 20.05.2023

ПРОБЛЕМА УЗГОДЖЕНОСТІ ТА ВІДПОВІДНІ КОМПРОМІСИ В РОЗПОДІЛЕНИХ РЕПЛІКОВАНИХ СИСТЕМАХ І БАЗАХ ДАНИХ: ОГЛЯД

Джафар Ахмед, Андрій Карпенко, Ольга Тарасюк, Анатолій Горбенко, Акбар Шейх-Акбарі

Розподілені репліковані бази даних відіграють важливу роль у сучасних комп'ютерних системах, уможливлюючи масштабоване, стійке до збоїв і високопродуктивне керування даними. Однак досягнення цих якостей вимагає вирішення низки компромісів між різними властивостями під час проектування та експлуатації розподіленої системи. У статті розглядаються компроміси в розподілених реплікованих базах даних і надається огляд останніх наукових публікацій, які вивчають розподілені системи зберігання даних. У статті спочатку обговорюється компроміс між узгодженістю та часовими затримками, який є природним для розподілених реплікованих сховищ даних і безпосередньо випливає з теорем CAP і PACELC. Узгодженість означає

гарантію того, що всі клієнти в розподіленій системі одночасно спостерігають однакові дані. Щоб забезпечити строгу узгодженість, розподілені системи зазвичай використовують механізми координації та протоколи синхронізації, які включають зв'язок і узгодження між розподіленими репліками. Ці механізми обумовлюють додаткові накладні витрати та можуть значно збільшити час, необхідний для виконання операцій, особливо якщо репліки глобально розповсюджені в мережі Інтернет. Крім того, у статті розглянуто компроміси, які існують між іншими властивостями, включаючи доступність, довговічність, вартість, споживання енергії, продуктивність операцій читання та запису тощо. У цій статті ми також надаємо огляд і класифікацію останніх досліджень у галузі розподілених реплікованих баз даних. Проаналізовані статті демонструють кілька основних напрямків досліджень, починаючи від оцінки продуктивності та порівняння різних баз даних NoSQL до пропозиції нових стратегій реплікації даних і запровадження нових моделей узгодженості. Зокрема, спостерігається перехід до вивчення гібридних моделей узгодженості з причинно-наслідковим упорядкуванням завдяки їхній здатності досягти балансу між гарантіями впорядкування операцій і високою продуктивністю. Дослідники також пропонують різні алгоритми контролю узгодженості та протоколи досягнення консенсусу для координації стану розподілених реплік. Висновки цього огляду можуть допомогти практикам приймати обґрунтовані рішення щодо проектування розподілених систем зберігання даних і керування ними, а також визначити подальші напрями актуальних наукових досліджень.

Ключові слова: розподілені бази даних; великі дані, NoSQL; реплікація; узгодженість; затримки; пропускна здатність; готовність; компроміси; CAP; PACELC; огляд.

Ахмед Джафар – асп. університету Лідс Бекетт, Лідс, Велика Британія.

Карпенко Андрій Сергійович – асп. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Тарасюк Ольга Михайлівна – канд. техн. наук, доц., доц. Одеського технологічного університету «ШАГ», Одеса, Україна; Університет Ньюкаслу, Ньюкасл, Велика Британія.

Горбенко Анатолій Вікторович – д-р техн. наук, проф., проф. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна; Університет Лідс Бекетт, Лідс, Велика Британія.

Шейх-Акбарі Акбар – канд. техн. наук, доц. університету Лідс Бекетт, Лідс, Велика Британія.

Jaafar Ahmed – PhD student, School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds, United Kingdom,
e-mail: j.ahmed6546@student.leedsbeckett.ac.uk.

Andrii Karpenko – PhD student; Department of Computer Systems, Networks and Cybersecurity, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine,
e-mail:a.karpenko@csn.khai.edu, ORCID: 0000-0003-2789-1168.

Olga Tarasyuk – PhD, Docent, Associate Professor, Odesa Technological University STEP, Odesa, Ukraine; Newcastle University, Newcastle, United Kingdom,
e-mail: Olga.Tarasyuk@ncl.ac.uk, ORCID: 0000-0001-5991-8631.

Anatoliy Gorbenko – Doctor of Science on Engineering, Professor, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine; School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds, United Kingdom,
e-mail: a.gorbenko@leedsbeckett.ac.uk, ORCID: 0000-0001-6757-1797.

Akbar Sheikh-Akbari – PhD, Reader; School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds, United Kingdom,
e-mail: a.sheikh-akbari@leedsbeckett.ac.uk, ORCID: 0000-0003-0677-7083.