International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 08 | Aug - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Conflicts Resolution in Mobile Distributed Real-Time Database Systems: Issues and Challenges

Divya Gupta[1], Sarvesh Pandey[2], and Udai Shanker[1]
[1]Department of Computer Science & Engineering, M. M. M. University of Technology, Gorakhpur, India
[2]Computer Science – MMV, Banaras Hindu University, Varanasi, INDIA
guptadivya28@gmail.com, sarveshpandey@bhu.ac.in, udaigkp@gmail.com

Abstract
The proliferation of cellular communication technology has been facilitated by the reduction in hardware costs, which has enabled a wide range of applications for mobile users, including but not limited to e-commerce, online transactions, and smart cities. Integrating mobility into databases has resulted in the creation of mobile distributed real-time database systems (MDRTDBS), which encounter recurrent disconnections that pose challenges to the realization of timely transactions and data consistency. This article presents a comprehensive review of existing solutions, emphasizing data conflict resolution, and proposes future strategies to meet the growing demands of society. Our ultimate objective is to establish an efficient MDRTDBS that offers seamless mobile experiences to individuals in our interconnected world.

Keywords: MDRTDBS, transaction processing, data conflict, mobile database, and concurrency control.

1 Introduction

A set of organized information arranged logically is called a database. Moreover, the software that serves as a medium for manipulating the database is called a Database Management System (DBMS) [1]. A Distributed Database System (DDBS) is characterized as a geographically dispersed database across multiple locations and interconnected by a communication network [2]. A real-time system (RTS) ensures timely completion of processing within the set deadline, failing which the process is terminated. The growth of real-time systems has led to a significant expansion in the scope and scale of applications that rely on them. Consequently, the demand for accessing data has increased manifold. To address this need, the DBMS has evolved to integrate with real-time systems, resulting in the emergence of real-time database systems (RTDBS) [3], which provide all database distributed Real-Time Database Systems (DRTDBS) are characterized by the inclusion of a deadline constraint for completing all database operations. Such constraint accurately reflects the realities of the internetworked world [4]. Within this particular system, the concept of time assumes an exceedingly critical and pivotal role. The fate of all transactions is delicately and intricately tied to meeting and adhering to highly stringent and demanding deadlines. It is quite surprising to note that even in the complete absence of any computational errors, transactions may face an abrupt and sudden termination if they fail to comply with the prescribed and stipulated timeliness.

The field of computing has experienced significant growth with the emergence of mobile computing in the 1990s, which is attributed to the advancements in telecommunication technology. Integrating four key technologies, namely DBMS, RTS, Distributed Systems, and Mobile Computing, has resulted in the developing of a powerful and versatile system known as MDRTDBS. This system now plays a crucial role in facilitating online transactions and location-based services. MDRTDBS accommodates clients and servers that support mobility by enabling timely data processing across remote locations connected through a computer network. Although this integration offers several benefits, it also presents inherent limitations, challenges, and issues that require careful consideration and resolution. [2][3][4][5]. The MDRTDBS system manifests a noteworthy attribute of uniformity, a trait that is not obligatory in RTS because of its allowance of task pre-emption. Nonetheless, it becomes imperative to harmonize these two ostensibly conflicting characteristics in creating such systems and ensure their concomitant fulfillment.

2 Overview of Mobile Distributed Real-Time Database Systems

The diagrammatic representation of the MDRTDBS's architecture is depicted in Figure 1. The system comprises diverse elements, including a fixed host (FH), a moving host (MH), a mobile support station (MSS), base stations (BS), and servers. The MH possesses mobility, while the other components exhibit a fixed disposition. A cell denotes a geographic region where various hosts are managed by an MSS (Mobile support Station). The MSS enables the complete communication of mobile hosts (MH). When an MH transitions from one cell to another, its communication is smoothly transferred from the current MSS to the new MSS of the destination cell. However, a temporary communication loss may arise during this transition, leading to disconnection. Notably, there is an asymmetrical bandwidth between the MH and the server, with the MH having a lower bandwidth than the server. The server deploys a broadcasting technique to optimize the server-to-MH communication with high bandwidth. Instead of transmitting data on request, the server broadcasts data to all MH within the cell, guaranteeing efficient data distribution.

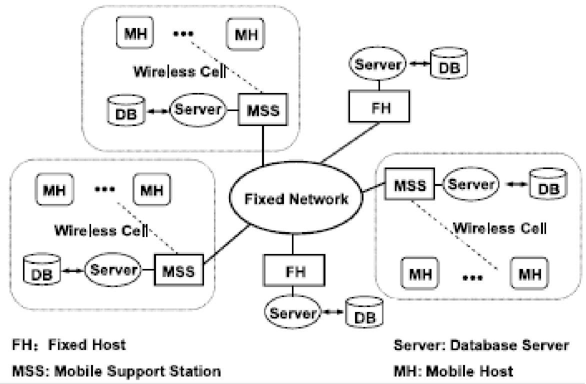

Fig. 1. The architecture of Mobile Distributed Real-Time Database System [15]

The present scenario involves mobile clients accessing data hosted on fixed servers, as observed in mobile commerce, e-banking, and video-on-demand applications. These applications require data consistency, confidentiality, and availability. To efficiently manage various mobile applications, it is possible to classify them based on their data management necessities.

Most mobile applications can be categorized as client-fixed hosts, where the client interacts with fixed servers. On the contrary, some applications involve a mobile client and a mobile host, such as phonebooks, smartwatches, health bands, and similar technologies. For such applications, data durability and synchronization are of utmost importance. The usage of diverse mobile applications is increasing rapidly, and the user base is expanding at an extraordinary rate. Mobile databases have evolved significantly, extending even to fixed clients and hosts, as evidenced by location-based services, tracking, and related applications.

In the context of a soft real-time database system, a transaction that fails to meet its deadline may be accommodated by the system if such occurrences are infrequent. This is particularly true in cases where the completed transaction remains an integral part of the system, such as live video streaming. However, when the system operates as an infirm real-time database system, missed deadlines may occur more frequently. Although the system may continue to function, the result may be rendered worthless [15][16][17], as is the case concerning weather forecasting systems. Deadlines tightly bind hard real-time database systems. The processing of results is only deemed valuable if it is completed within the deadline. Any results that exceed the deadline are rejected. The primary design objectives of such systems are to ensure the timely completion of transactions, as failure can lead to catastrophic consequences [18][19][20], as seen in missile firing, aircraft flight, and the management of nuclear power plants. Thus, the categorization of MDRTDBS into soft, firm, or hard is ascertained based on the system's approach toward transactions that do not adhere to their prescribed time limits.

The introduction of MH brings forth two significant challenges: bandwidth limitations and frequent disconnections. As previously attempted in other systems, addressing disconnection as a characteristic rather than a failure presents a formidable challenge. In the case of MDRTDBS, numerous factors contribute to the complexity of meeting transaction deadlines, such as network performance, resource scheduling, concurrency control, and the commit procedure. Designing such a system necessitates effectively managing a distributed environment while accommodating all the constraints of a mobile setting. The combined approach of processing transactions in distributed and mobile computing has proven to be a boon for humanity in the present era. The emergence of MDRTDBS significantly eases various aspects of society, ultimately rendering life more convenient and efficient.

3 Performance Issues and Research Challenges in MDRTDBS

Considerable research has been conducted in the domains of Real-Time Systems (RTS) and Distributed Databases (DDBS). Predominantly, the focus of research in RTS pertains to the meticulous analysis of task scheduling to meet the stipulated deadline. [6][7][8][9][22][23][24][25][26]27][28][29]. To ensure the efficient functioning of the Distributed Database Systems (DDBS), it is imperative to incorporate robust and effective Concurrency Control Protocols (CCPs) because the entire database is partitioned and distributed across multiple geographically distant locations. This has led to significant research in this domain, with numerous works being undertaken to enhance the system's efficacy. [10][11][12][13][30][31][32][33][34]. Research endeavors in both domains are reciprocally advantageous. Nevertheless, there are also certain undesirable outcomes. The most formidable among them is the need to sustain the ACID property for the transaction. In recent times, researchers have encountered heightened complexities in concentrating their efforts on DRTDBS, and more specifically on MDRTDBS [14][15][16], to tackle the latest societal applications.

Concurrency control protocols (CCP) ensure consistency and timeliness in transactions. Nonetheless, CCPs designed for traditional databases are unsuitable for DRTDBS. Specifically, two-phase locking and optimistic CCPs have not proven effective.[41] scheduling transactions without taking into account any hierarchy can result in priority inversion. In this scenario, a transaction with a higher priority is prevented from executing due to a lower priority transaction holding the lock being allowed to continue its execution. Various solutions to this issue have been reported in the literature.[48][49] [50][51].

The studies conducted in the area of CCPs have predominantly focused on DRTDBS [1][8][9]; unfortunately, literature on the subject of promising developments in the mobile environment is scarce.

Due to the evolving concepts and technologies, such as cloud computing, block chaining, replication, security, reliability, and the changing locations of DDBS users, the previous CCPs proposed are no longer applicable in facilitating tasks for mobile computing. The combination of deadline and consistency poses significant challenges and problems in enhancing the performance of MDRTDBS.[19][20][21]. Most frequently, the established CCPs designed for DRTDBS have been expanded to encompass these rapidly expanding applications in a mobile setting. However, their efficacy may fluctuate and not meet societal expectations owing to the inherent features of MDRTDBS.[9]. The primary aim of MDRTDBS, indeed, is to fulfill the transaction's temporal and consistency constraints.[27] the issue of frequent disconnection, limited resources, and unpredictability is attributed to user mobility and other factors.[22]. It was discovered that the two-phase locking (2PL) protocol was inadequate in fulfilling its intended function.[48]. Hence, it is imperative to devise an innovative and unprecedented mechanism for managing concurrency and protocols for regulating other operations in MDRTDBS. This has emerged as a highly dynamic field of research in contemporary times.

4 Research Directions for MDRTDBS

The initial objectives of computer networks were fundamentally oriented toward wired technology. Nevertheless, the transition from wired technology to wireless, and subsequently to mobile networks, has brought about a consequential shift in the research domain and its associated technologies. Consequently, a significant gap has emerged that requires attention and resolution. All aspects of database systems, including CCP, data replication, transaction processing, and data security, must be modified to the fullest extent to accommodate the constraints posed by MDRTDBS. Numerous issues inherent to the mobile environment contribute to the challenges faced by MDRTDBS, such as site failures, communication delays, network congestion, data conflicts among transactions, and priority inversion. These issues further exacerbate the likelihood of meeting transaction deadlines.

The exponential growth in the use of mobile applications, particularly in the financial sector, has escalated the complexity of challenges related to MDRTDBS. Therefore, it demands greater efforts for effective handling. Given the trend of ubiquitous access to distributed mobile devices and the crucial need for real-time data, the foundation of MDRTDBS is established. To provide insight into the main directions for future work in this area, we summarize key aspects as follows [50][51].

1. The optimization of network latency, prevention of system malfunction, minimization of context switches, and efficient inter-process communication are all significant factors in enhancing system performance.

2. Developing energy efficient (CCP), in conjunction with locking and commit protocols, has been explored extensively in academic research.

3. An effective mechanism for the recovery of databases.

4. A novel mechanism has been proposed for cooperative transactions.

5. The present study concerns the development of an algorithm for adaptive CPU scheduling.

6. A more effective mechanism for addressing the issue of priority inversion is required.

7. Consistency management.

8. Handling of issues associated with multiple read-read operations.

9. The crafting of proficient and adaptable protocols for mobile environments concerning voluminous and semi-structured databases is paramount.

10. Fault-tolerant procedures designed for the mobile environment are the subject of inquiry.

11. New policies for assigning priorities tailored to meet the demands of the mobile landscape.

12. Additional dynamic replication techniques for the mobile computing context.

13. A computational process aimed at determining the optimal duration of time in advance to improve the efficiency of scheduling and pre-emptive measures.

14. In database management, one of the essential aspects is considering both the size and type of the database.

15. the impact of the magnitude of primary memory, disk storage, and buffer capacity has been examined.

16. Developing a proficient mechanism for allocating priorities and deadlines to transactions that encompass system intelligence is a crucial undertaking.

17. Enhancing resilience and reliability (CCPs) is imperative for big data systems with extensive transactions.

18. The present study aims to establish a set of protocols and mechanisms that leverage the capabilities of machine learning and artificial intelligence capabilities to enhance the system's efficiency under consideration.

19. Creating protocols and mechanisms that seamlessly integrate with emerging technologies, including but not limited to blockchain, big data, cloud computing, IoT, grid computing, edge computing, and fog computing, is a crucial development area.

Henceforth, it is imperative to devise novel mechanisms and protocols for MDRTDBS that seamlessly incorporate emerging technologies such as Big Data, Cloud Computing, IoT, Grid Computing, Edge Computing, and the like.

5 Conflicts Resolution

Among the manifold factors contributing to the failure to meet the established timeline, the most perplexing one is the occurrence of data discrepancies amidst the transactions [4]. The existence of a mobile transaction can be bifurcated into two distinctive stages, specifically, the stage of execution and the stage of commitment. During the execution stage, data processing occurs at multiple locations within a mobile setting. Upon the completion of the execution stage, the commitment stage ensues. During this latter stage, diverse sites pledge to guarantee the atomicity of failure [3]. If a conflict in data arises amidst transactions during the execution stage, it is labeled as an executing-executing conflict. Very few researchers have proposed CCPs for MDRTDBS [4][16][17] that solve the executing-executing conflict. Another form of conflict arises during the execution and commitment stages of transactions, called the executing-committing conflict. To resolve this type of data conflict, blocking may be employed. Specifically, if a transaction has acquired a lock on a data item in the committing stage, and another transaction seeks access to the same data item during the execution stage, said transaction will be prevented from proceeding until the initial transaction has relinquished the lock. Two fundamental methods exist for resolving data conflicts that arise between executing and committing transactions. The primary approach involves restarting the committing transaction with lower priority, as with executing-executing conflicts. This approach can, however, result in the regrettable outcome of squandering valuable resources and time. The second approach entails impeding the higher priority executing transaction and assigning the committing transaction a superior priority to all other executing transactions. Nevertheless, blocking transactions can deleteriously impact the functionality of a Mobile Distributed Real-Time Database System. Given the frequent network failures in a mobile environment, the likelihood of a blocked transaction meeting its deadline is not particularly promising. Most existing RT CCPs address the executing-executing data conflict limited to DRTDBS only. Few have worked on executing-committing data conflict [25][26], but again for the DRTDBS only. No substantial research has been conducted on MDRTDBS.

Therefore, it can be delineated that the prime concerns of MDRTDBS pertain to data utilization and can be broadly classified into two distinct categories [5].

- Data management
- Transaction management

In data management, topics such as data caching, data broadcast, and data classification have received considerable attention. A multitude of research endeavors have been documented in this domain, yielding a plethora ofinnovative methodologies.[30][31][32][33][34][35][36][37][38].

Transaction management generally deals with query processing [35], transaction processing [28][34][38][41][45], and database recovery [45][46]. The impact of mobility has significantly influenced all aspects of transaction processing. However, scholars have yet to undertake a comprehensive investigation into its effects and potential remedies. The techniques and procedures developed for traditional wired systems or wireless settings are unsuited for deployment in the context of Mobile Distributed Real-Time Databases Systems (MDRTDBS). Merely revisiting the work done earlier or tuning them for mobility [40][41][44][45][48] is not sufficient. Hence, conducting a comprehensive inquiry into the matters at hand, along with their potential resolutions, is imperative as it is the current necessity.

A commit protocol is employed to guarantee atomicity during transaction processing. The Two-Phase Commit Protocol (2PC) is the most commonly implemented in Distributed Real-Time Database Systems (DRTDBS) [1][5][6][10][11]. In a mobile environment, the Two-Phase Commit (2PC) protocol experiences protracted delays, primarily due to two fundamental factors. Firstly, the distributed nature of the sites, accompanied by varying processing power and resource availability, contributes to this effect. Secondly, the frequent disconnection of the coordinator or participants further complicates the mobile environment. Consequently, establishing an appropriate time-out period for the system poses a significant challenge. It is imperative to optimize this time-out period in a distributed environment for optimal results. These delays lead to an increased blocking time, affecting other transactions waiting for resources held by blocked transactions. This scenario creates a chain of blocks, commonly called "thrashing," which exacerbates the performance issues overall.

To overcome these hurdles, careful consideration and optimization of the time-out period are indispensable in enhancing the efficiency and reliability of the 2PC protocol in the mobile environment [3]. The matter at hand has yet to receive sufficient scrutiny from scholars. Certain scholars have recommended the unilateral authorization of transactions and, in the event of complications, the retraction of the compensatory transaction as a means of remediation [17][18]; however, it contradicts one of the fundamental ACID properties of the transaction, specifically Durability [9]. Additionally, it is worth noting that the compensatory transaction is often ineffective in most real-time systems. In situations where a committing transaction is obstructed, a sequence of dependencies arises as other executing transactions contend with conflicting data. In comparison, although researchers have attempted to address this issue [49], a comprehensive resolution has not yet been achieved. In [25], the author has posited a novel framework, namely DCCR, which is confined solely to DRTDBS and fails to conform to the standards set forth by MDRTDBS.

Concurrency control is one of the key issues for ensuring timeliness for any DBMS. The conventional CCP such as two-phase locking protocol (2PL) or its various variants [4][7][11][14][15][17][19][25][4][43] locks the transaction to ensure isolation property. In this process, it is not uncommon for a lower transaction to impede the progress of a higher transaction, thereby leading to a phenomenon widely acknowledged as priority inversion [14]. The solution to blocking and priority inversion has been given by many researchers [50][51], but that works well for a single site of DRTBDS only [52][53].

Therefore, transaction processing within the context of Mobile Distributed Real-Time Database Systems (MDRTDBS) necessitates an increased focus and research to satisfy contemporary requirements effectively. To enhance the efficiency of MDRTDBS, the following vital areas mandate further investigation:

1. Ensuring ACID compliance while developing relaxed ACID properties for mobile transactions.

2. Innovating new mobile transaction models tailored to the mobile environment's unique characteristics.

3. Ensuring the serializability of transactions to enable seamless concurrent execution.

4. Maintaining global consistency across distributed sites.

5. Designing efficient concurrency control protocols and commit procedures.

6. Optimizing CPU scheduling to improve system performance.

7. Enhancing predictability and consistency in MDRTDBS operations.

By addressing these critical points, MDRTDBS can be better equipped to meet the demands of modern applications and deliver a more robust and reliable performance. To succinctly outline the concerns and research obstacles within the realm of MDRTDBS, a considerable amount of effort has been expended in DRTDBS, but these endeavors have not been thoroughly extended to MDRTDBS. It is plausible that this discrepancy is attributed to the conflicting properties of distributed real-time and mobility.

The unpredictability inherent in MDRTDBS is instigated by blocking, which impedes real-time (RT) operations yet is simultaneously indispensable for consistency. Additional factors exacerbating unpredictability include site failures, priority inversion, communication delays, and disconnections. Managing resources in MDRTDBS necessitates the regulation of shared data items, input/output (I/O) devices, main memory, and central processing units (CPU). When designing a Mobile Distributed Real-Time Database System (MDRTDBS), it is imperative to take into account several critical factors. These factors include security, fault tolerance, recovery, and availability. In addition, it is noteworthy that the current system lacks an effective mechanism for CPU scheduling. Therefore, our plan involves the development of Concurrency Control Protocols (CCPs) and commit procedures tailored for the mobile environment, ensuring resistance to priority inversion. It is worth mentioning that all proposed protocols and techniques must remain compatible with ever-evolving technologies such as 5G/6G. Moreover, in today's shift towards a mobile environment, exploring the locality aspect of mobile nodes becomes imperative. This investigation may encompass on-mobile service search, on-move path locator, and satellite data link integration, among other aspects. By addressing these considerations, our MDRTDBS aims to deliver enhanced performance and responsiveness, keeping pace with the dynamic landscape of mobile technology. [54][55][56][57].

5 Conclusions

The escalating size and utilization of databases have resulted in users expecting immediate access through mobile devices. This expectation pertains to various domains, including but not limited to e-commerce, online transactions, internet browsing, social networking, location-based applications, and more. As a result, Mobile Distributed Real-Time Database Systems (MDRTDBS) has emerged as a highly coveted area of research in computer science. This paper examines the properties and challenges of MDRTDBS, with a specific focus on data conflicts. Additionally, it provides a comprehensive overview of current research endeavors. In essence, this investigation accentuates the significant gap and imbalance in the research landscape of MDRTDBS, which calls for substantial further exploration. To effectively tackle the evolving technologies and societal demands, future research must keep pace with the rapid changes in this dynamic field. Bridging this gap will play a critical role in advancing the capabilities and efficiency of MDRTDBS to meet the growing expectations of mobile users.

References

[1] Ullman, Jeffrey D. Principles of database systems. Galgotia publications, 1984.

[2] U. Shanker, M. Misra, and A. K. Sarje, “Distributed real-time database systems: Background and literature review,” *Distrib. Parallel Databases*, 2008.

[3] B. Kao and H. Garcia-Molina, “AN OVERVIEW OF REAL-TIME DATABASE,” *Real Time*

Comput. Springer Berlin Heidelb., pp. 1–23, 1994.

[4] Lam, Kam-Yiu, et al. "Concurrency control in mobile distributed real-time database systems." Information systems 25.4 (2000): 261-286.

[5] V. Swaroop and U. Shanker, "Mobile Distributed Real-Time Database Systems: Research Challenges," Proceedings of International Conference on Computer and Communication Technology (ICCCT 2010), Motilal Nehru National Institute of Technology, Allahabad, India, pp. 421-424, 2010.

[6] Pandey, Sarvesh, and Udai Shanker. "Transaction execution in distributed real-time database systems." Proc. Int. Conf. Innov. Inf. Embed. Commun. Syst. 2016.

[7] Aldarmi, Saud Ahmed. Scheduling soft-deadline real-time transactions. Diss. University of York, 1999.

[8] U. Shanker, M. Misra, and A. K. Sarje, "Some performance issues in distributed real-time database systems," *VLDB Ph.D. Work.*, 2006.

[9] Barbará Daniel (1999): Mobile Computing and Databases-A Surve, IEEE Transactions on Knowledge and Data Engineering.11(1).pp 108-117.

[10] P. A. Laplante, Real-Time Systems Design and Analysis, 3rd ed., IEEE Press, 2004

[11] G. C. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, 3rd ed., Springer, 2005.

[12] R. L. Panigrahi and M .K. Senapaty, "Real Time System for Software Engineering: An Overview," Global Journal for Research Analysis, Vol. 3, Issue 1, pp. 25-27, January 2014.

[13] J. W. S. Liu, Real-Time Systems, Prentice Hall, 2000

[14] Prabhu, Nitin, and Vijay Kumar. "Concurrency control in mobile database systems." 18th International Conference on Advanced Information Networking and Applications, 2004.AINA 2004. Vol. 2. IEEE, 2004.

[15] Moiz, Salman Abdul, et al. "Concurrency control in mobile environments: Issues & challenges." International Journal of Database Management Systems 3.4 (2011): 147.

[16] Lei, Xiangdong, et al. "Concurrency control in mobile distributed real-time database systems." Journal of Parallel and Distributed Computing 69.10 (2009): 866-876.

[17] Lam, Kam-Yiu, et al. "Concurrency control strategies for ordered data broadcast in mobile computing systems." Information Systems 29.3 (2004): 207-234.

[18] Zhao, Wei, Krithi Ramamritham, and John A. Stankovic. "Preemptive scheduling under time and resource constraints." IEEE Transactions on Computers 100.8 (1987): 949-960.

[19] C. Hsueh, K. Lin, and I. C. Society, "Scheduling Real-Time Systems with End-to-End Timing Constraints Using the Distributed Pinwheel Model," vol. 50, no. 1, pp. 51–66, 2001.

[20] G. Bernat, A. Burns, and S. Member, "Weakly Hard Real-Time Systems," *IEEE Trans. Comput.*, vol. 50, no. 4, pp. 308–321, 2001.

[21] F. Liberato, R. Melhem, and D. Mosse, "Tolerance to Multiple Transient Faults for Aperiodic Tasks in Hard Real-Time Systems," vol. 49, no. 9, pp. 906–914, 2000.

[22] S. M. Shatz, S. Member, J. Wang, and M. Goto, "Task Allocation for Maximizing Reliability of Distributed Computer Systems," vol. 11, no. 0, 1992.

[23] J. R. Haritsa, K. Ramamritham, and R. Gupta, "The PROMPT real-time commit protocol," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 2, pp. 160–181, 2000.

[24] M. Xiong, K. Ramamritham, J. R. Haritsa, and J. A. Stankovic, "MIRROR: A state-conscious concurrency control protocol for replicated real-time databases," *Inf. Syst.*, vol. 27, no. 4, pp. 277–297, 2002.

[25] S. Poledna, A. Burns, A. Wellings, and P. Barrett, "Replica Determinism and Flexible Scheduling in Hard Real-Time Dependable Systems," vol. 49,

no. 2, pp. 100–111, 2000.

[26] S. K. Baruah and J. R. Haritsa, “Scheduling for Overload in Real-Time Systems,” vol. 46, no. 9, pp. 1034–1039, 1997.

[27] S.-L. H. Y. Zhang, H. Kameda, “Comparison of dynamic and static load-balancing strategies in heterogeneous distributed systems,” vol. 144, no. 2, 1997.

[28] J. Jonsson, “Deadline Assignment in Distributed Hard Real-Time Systems with Relaxed Locality Constraints,” pp. 432–440, 1997.

[29] M. C. W. B.R. Swim, M. Benmaiza, M. Tayli, “Predictable distributed dynamic scheduling in RTDOS,” *IEEE Proc.-Comput.Digit.Tech.*, vol. 144, no. 3, 1997.

[30] D. Kang, R. Gerber, and M. Saksena, “Parametric design synthesis of distributed embedded systems,” *Comput. IEEE Trans.*, vol. 49, no. 11, pp. 1155–1169, 2000.

[31] H. R. M. Systems, S. Ghosh, S. Member, and I. C. Society, “Fault-Tolerance Through Scheduling of Aperiodic Tasks in,” vol. 8, no. 3, pp. 272–284, 1997.

[32] Srinivasan, Santhanam, and Niraj K. Jha. "Safety and reliability driven task allocation in distributed systems." IEEE Transactions on Parallel and Distributed Systems 10.3 (1999): 238-251.

[33] T. F. Abdelzaher, I. C. Society, and K. G. Shin, “Combined Task and Message Scheduling in Distributed Real-Time Systems,” vol. 10, no. 11, pp. 1179–1191, 1999.

[34] Dominic, M., and Bijendra N. Jain. "Conditions for online scheduling of hard real-time tasks on multiprocessors." journal of Parallel and Distributed Computing 55.1 (1998): 121-137.

[35] Sagar, G., Anil K. Sarje, and Kamal U. Ahmed. "Task allocation techniques for distributed computing systems: a review." Journal of microcomputer applications 12.2 (1989): 97-105.

[36] Sagar, G., and Anil K. Sarje. "Task allocation model for distributed systems." International journal of systems science 22.9 (1991): 1671-1678.

[37] R. Harding, D. Van Aken, A. Pavlo, and M. Stonebraker, “An Evaluation of Distributed Concurrency Control,” *Vldb*, vol. 10, no. 5, pp. 553–564, 2016.

[38] S. Bahrami, M. Shahrokhzadeh, and S. M. Mirhoseini, “An improved 2PL-HP based concurrency control algorithm in the real-time database,” vol. 6, no. 7, pp. 1677–1684, 2011.

[39] J. M. Faleiro and D. J. Abadi, “Rethinking serializable multi-version concurrency control,” *Vldb’14*, vol. 8, no. 11, pp. 1190–1201, 2014.

[40] S. Mohan, M. K. Yoon, R. Pellizzoni, and R. B. Bobba, “Integrating security constraints into fixed priority real-time schedulers,” *Real-Time Syst.*, vol. 52, no. 5, pp. 644–674, 2016.

[41] P. A. Bernstein, D. W. Shipman, and J. Rothnie, “Concurrency Control in a System for Distributed Databases (SDD-1),” vol. 5, no. 1, pp. 19–51, 1980.

[42] Menasce, Daniel A., and Richard R. Muntz. "Locking and deadlock detection in distributed databases." IEEE Transactions on Software Engineering 3 (1979): 195-202.

[43] Reddy, P. Krishna, and S. Bhalla."Non-blocking concurrency control in distributed database systems." Proceedings of the First Aizu International Symposium on Parallel Algorithms/Architecture Synthesis.IEEE, 1995.

[44] Lu, Hongjun, and K. Tan. "Buffer and load balancing in locally distributed database systems." [1990] Proceedings.Sixth International Conference on Data Engineering.IEEE, 1990.

[45] Lin, Min-Sheng, and Deng-Jyi Chen. "The reliability problem in distributed database systems." Proceedings of ICICS, 1997 International Conference on Information, Communications and Signal Processing. Theme: Trends in Information Systems Engineering and Wireless Multimedia Communications (Cat..Vol. 2. IEEE, 1997.

[46] P. Bailis, A.. Davidson, A..Fekete, A..Ghodsi, J. M..Hellerstein, and I..Stoica, “Highly available transactions: Virtues and limitations,” Vldb ’13,

vol. 7, no. 3, pp. 181–192, 2013.

[47] Bhargava, Bharat. "Concurrency control in database systems." IEEE transactions on knowledge and data engineering 11.1 (1999): 3-16.

[48] Lam, K-W., and S-L. Hung. "Analytic modeling of locking protocols in database systems." IEEE Proceedings-Computers and Digital Techniques 146.3 (1999): 161-167.

[49] Zhang, Aidong, Marian Nodine, and Bharat Bhargava. "Global scheduling for flexible transactions in heterogeneous distributed database systems." IEEE Transactions on Knowledge and Data Engineering 13.3 (2001): 439-450.

[50] Pandey, Sarvesh, and Udai Shanker. "A one-phase priority inheritance commit protocol." International Conference on Distributed Computing and Internet Technology. 4Springer, Cham, 2018.

[51] Pandey, Sarvesh, and Udai Shanker. "On Using Priority Inheritance-Based Distributed Static Two-Phase Locking Protocol." Advances in Data and Information Sciences. Springer, Singapore, 2018.179-188.

[52] Sarvesh Pandey and Udai Shanker, “RACE: A Concurrency Control Protocol for Time-Constrained Transactions,” The Arabian Journal of Science and Engineering, Springer, Vol. 45, No. 12, May 26, 2020

[53] Sarvesh Pandey and Udai Shanker, “Transaction Scheduling Protocols for Controlling Priority Inversion: A Review,” Journal of Computer Science Reviews, Elsevier, Volume 35, February 2020, 100215

[54] Ajay Kumar Gupta and Udai Shanker, “MAD-RAPPEL: Mobility Aware Data Replacement & Prefetching Policy Enrooted LBS,” Journal of King Saud University-Computer and Information Sciences. Elsevier, May 20, 2020

[55] Ajay Kumar Gupta and Udai Shanker, “OMCPR: Optimal Mobility Aware Cache Data Pre-Fetching and Replacement Policy using spatial K-anonymity for LBS,” International Journal of Wireless Personal Communications, Springer, Vol. 114, No. 2, 949-973, September 2020

[56] Ajay K. Gupta, Udai Shanker, “Some Issues for Location Dependent Information System’s Query in Mobile Environment,” 29th ACM International Conference on Information and Knowledge Management, CIKM Doctoral Consortium (DC), Galway, Ireland, October 20, 2020

[57] Ajay K. Gupta and Udai Shanker, “Location Dependent Information System's Queries for Mobile Environment,” Database Systems for Advanced Applications-DASFAA2018 International Workshops: BDMS, BDQM, GDMA, and SeCoP, Gold Coast, QLD, Australia, May 21-24, 2018, Proceedings. Lecture Notes in Computer Science 10829, Springer 2018, ISBN 978-3-319-91454-1.