# BEYOND LOCKING: ADVANCED COLLISION DETECTION AND MITIGATION STRATEGIES FOR HIGH-VOLUME ENTERPRISE SYSTEMS

**JAYANT TYAGI**

Jtyagi@salesforce.com

**SALESFORCE**

## Abstract

Enterprise systems face primary operational challenges with collision resolution because they need to process real-time information and sustain their distributed systems at high volumes. The two-phase locking (2PL) system performs poorly when efficiently handling high concurrency levels. This research evaluates rest and compares collision detection and mitigation methods using Optimistic Concurrency Control (OCC) and decentralized models, machine learning techniques, and combination strategies. Examining different solutions takes place according to their ability to scale, processing speed, and immediate response times. The proposed framework integrates OCC methods with decentralized approaches because it improves system adaptability. The research produces findings about the cost-benefit analysis between approaches and recommends new ways to develop collision resolution methods for critical applications in the future

**Keywords**: Collision Resolution, High-Volume Systems, Optimistic Concurrency Control, Decentralized Algorithms, Distributed Systems, Conflict Mitigation, Machine Learning for Conflict Resolution, Hybrid Solutions.

## Introduction

Complex transaction processing, data warehouses, cloud, and similar applications are designed to process many customers ' requests [1]. These systems are becoming more crucial in finance, e-commerce, healthcare, and telecommunication industries, where data processing and business transactions must be smooth and effective. However, conflict or collision is one of the most significant issues in such systems, which happens when two or many transactions/operations want to use the same data simultaneously [2]. Such conflicts may result in the replication of data, malfunction of the systems, or even a decrease in the quality of the services such systems provide, making business-critical systems highly vulnerable.

Two key ways to avoid resource-sharing conflicts include two-phase locking (2PL) and deadlock-detection techniques [3]. These techniques involve locking the particular resources before they can be accessed, which allows only one operation to change the resource. Although these approaches have worked well, they fail to meet the requirements of currently highly scalable and performing distributed systems [4]. Since the lock-based mechanisms, there have been issues with concurrency, wastage of time, and high latency, mainly when large numbers of transactions or resources are located in different geographical regions.

Therefore, more researchers have focused on finding other collision detection and prevention techniques that can prevent the drawbacks of locking methods. This paper aims to provide more sophisticated collision resolution methods other than locks, as stated in the following sections. Notably, it explores such methods as optimistic concurrency control (OCC), which lets several transactions run simultaneously with the least effect on one another, and distributed algorithms that introduce concurrency control in distributed environments. However, machine learning models are also being advocated due to their effectiveness in identifying and preventing conflicts as they occur with the help of data collected from previous instances.

The main goal of this paper is to assess the efficiency of these new approaches in large-scale enterprise applications, especially regarding the high number of users, the load on the application, and the response time. Thus, using a literature review, the strengths and weaknesses of every method under consideration are identified, and they may be effectively utilized for improving collision resolution in large-scale systems. Once identified, the paper discusses future directions and lines of research in the field, thus giving the reader an outlook on the future of conflict management in enterprise settings. Therefore, this analysis will help ascertain a broad view of the most advisable approaches towards collision resolution in systems under the highest demand with critical missions by outlining possible and highly effective strategies in this case.

## Literature Review

Latching mechanisms have always been central to mediating concurrent activities and unifying operational DBMS [5]. Two of the most popular methods for avoiding or handling these problems are two-phase locking (2PL) and deadlock detection. The principle behind 2PL is that transactions must get all the required locks to manipulate a resource, and the locks are released

only when a transaction commits or aborts [6]. On a similar note, the purpose of 2PL is to prevent other transactions from accessing the same resources simultaneously. Employees realize that two transactions want to write data simultaneously; two-phase locking guarantees that one transaction gets an exclusive lock, thus avoiding inconsistencies or errors [7]. However, conventional record-locking methods are not without the problems that are inherent to large-scale, multi-tiered systems. As transactions and resources grow, the overhead of obtaining and releasing locks can be relatively high [8]. Every transaction has to ask for permission to lock the specific resources required, resulting in delays and contention. When resources may be located on different servers, the problem with the acquisition and release of locks is even magnified [9]. Also, the distribution of nodes and required synchronization to handle locks often leads to the point that is a significant source of delay that can significantly impact the whole system throughput.

Another disadvantage of traditional locking is the issue of providing a lower degree of concurrency in the system. In another case, when a particular resource is locked, other transactions that require this resource must enter a waiting stage and only be granted access once the holder releases the resource [10]. This results in reduced parallelism in the system and may lead to increased latency, especially when the system is loaded or more transactions require more locks. Under high congestion, when a great deal of transactions are being executed, this form of concurrency is likely to significantly affect the system's performance and efficiency [11]. Moreover, deadlocks are possible when two or more transactions cannot proceed, as each waits for the other party to liberate a lock. Intricate detection algorithms solve deadlocks and, sometimes, by undoing one or several transactions that may further slowdown performance.

Optimistic Concurrency Control, or OCC, is another strategy that uses locks [12]. The OCC believes that conflicts are the exception rather than the usual situation and that transactions can be made without acquiring locks. A conventional lock is not purchased before the transactions, and they can perform their operations on the data as desired [13]. However, before the transaction is committed, a validation check is done on the system for any possible conflict. If no conflicts are identified, the transaction will go through smoothly; however, if conflicts are seen—meaning that another transaction has altered its data—it is rolled back and retried. The main benefit of the OCC protocol is that it is possible to obtain a much higher degree of concurrency compared to the case with the utilization of locks [14]. Since the acquisitions of locks are not required to carry out these

transactions, more transactions can occur at once, making the throughput higher. OCC is best suited to those systems that do not experience high levels of conflict since no lock requests will be granted until the already initiated transactions are complete [15]. This leads to improved system efficiency, especially for systems with a high number of read operations compared to write operations.

OCC has specific cons, mostly when several processes require contention for resources; hence, the chance of meeting is high [16]. When conflicts are frequent, the cost of having to roll back transactions and then retry for a possible solution is usually very high in terms of the system's overall performance system's overall performance. This becomes worse when such systems have many transactions that might require retrying to complete their process, leading to more extended time and low throughputs. However, there are some issues in the traditional design of OCC, as has been pointed out by various researchers. Here are some improvements to OCC that have been called for in recent years: For instance, a priority mechanism has been proposed where priority-based transactions will be preferred in conflict [18]. The drawbacks of pure OCC have also led to the development of other hybrid methods of concurrency control that incorporate other algorithms in combination with OCC.

With the increasing complexity of systems, concurrency control methods decentralized in their structure have been adopted [19]. Unlike centralized systems, which incorporate a central coordinator who is supposed to keep records of locks and which agent to provide the lock, decentralized systems divide the jobs of conflict detection among the different nodes or agents. It allows each node to maintain an independent list and status of ongoing transactions and decide which subsequent transactions can be committed without infringement [20]. Vector clocks are one of the oldest decentralized algorithms for maintaining information about causality in the distributed environment. Each node in the system has its particular vector of the timestamps that are changed every time any transaction is performed. In case of a collision of two transactions, their vector clocks help decide between them which will happen first [21]. This facilitates the identification of possible conflict and ensures that the transaction takes place to get over the probation of the transaction coordinators' two-party system. Secondly, distributed timestamp ordering (DTO) is also used to order the transactions according to their timestamps, where every node has a single global order, and conflicts are resolved by a single node [22].

This helps avoid bottlenecks and allows the organization to scale out due to increased nodes and resources. Also, encrypted systems are more reliable for their scalability, which means that each node of the particular system deals with its transactions and does not share responsibility with the whole system. As a result, there is no issue with the single point of failure [23]. Although there are always some problems with decentralized algorithms, they are also vulnerable. There are challenges, one of them being the clock synchronization challenge, where the clocks in the different nodes of the system are not always synchronized [24]. This may confuse when conflicts are detected, their order in the transaction sequence, and other issues related to system states are addressed. Moreover, as discussed above, decentralized systems usually involve the B elements and require complex coordination protocols so that the P nodes can exchange the messages or transactions processed in the correct order [24]. These coordination requirements could complicate and increase the overhead, mainly when many nodes within the system exist.

For a long time, high-volume systems have posed a significant challenge for engineers due to complex collisions, but the development of machine learning has provided a solution to this [25]. Machine learning algorithms can also be used to forecast conflict, depending on the transaction information, system traits, and other attributes that are seen fit to be used. The clientele's machine learning models help identify trends from large databases, which may be challenging with regular analysis [26]. After learning, such models can offer responsive conflict solution approaches, particularly beneficial with intricate, many-parametric conflict resolution processes where conventional methods may prove ineffective [27].

For example, using machine learning algorithms, it is possible to estimate at what time two transactions may interfere with each other, and therefore, the system corrects its activity and/or priorities in the transaction processes [28]. This can decrease the chance of conflict, increase system performance, and decrease rollback and retry. Other than that, the system's architecture can learn at different instances whenever there is new data or changes in the pattern of transactions [29]. However, they also bring some challenges when implementing collision resolution using machine learning. Training large sets of databases is often not obtainable, especially if the learning formation is in a proprietary or real-time structure [30]. Furthermore, machine learning models can be computationally expensive models that take time and a lot of computational power to analyse what is given and make a prediction. Last but not least, the interpretability of this type of model

can be an issue, especially in critical applications where one needs to understand what is and is not happening.

Based on such considerations, realizing the strengths and weaknesses of individual approaches, the latest publications have suggested the use of the combined approach that would allow for the inclusion of several techniques into one particular algorithm, which would solve the problems of scalability, efficiency, and real-time operation [31]. For instance, hybrid models might use optimistic concurrency control associated with decentralized algorithms and vice versa; in this way, the system will be able to enjoy the features that are provided by OCC while at the same time gaining the advantages of decentralized methods [32]. Likewise, there is an opportunity to apply machine learning together with traditional locking mechanisms or OCC to make the priorities of the transaction dependent on different conditions. As for the advantages, they are flexible, as hybrids can be easily adjusted to the variations in demands and loads of the system. Hybrid models, as proposed above, can provide better performance in low and high contentions due to the combination of various techniques [33]. Nevertheless, the hybrid models are more complicated, and their design is more intricate because all the components must be well-coordinated. It may also increase the cost of implementation and maintenance since the system's overall functionality improves.

Table 1: Literature review summary

| Study | Approach | Key Findings | Limitations |
|---|---|---|---|
| Nookala (2023) | Two-Phase Locking (2PL) | Works well for small-scale systems | High latency, scalability issues |
| Fugkeaw and Hak (2024) | Decentralized Conflict Detection | Effective in distributed environments | High communication overhead |
| Hassan et al. (2023) | OCC with Priority-Based Execution | Reduces rollback overhead in some cases | Performance degrades under high contention |
| Eldeeb et al. (2023) | Distributed Timestamp Ordering (DTO) | Ensures transaction ordering in distributed systems | Requires precise time synchronization |
| Xie and Zhang (2024) | Machine Learning-Based Conflict Resolution | Predicts and mitigates collisions dynamically | High computational requirements |
| Azevedo et al. (2024) | Hybrid Approaches | Combines multiple methods for flexibility | Complex implementation, costly to maintain |

## Methodology

In an attempt to assess and compare advanced collision detection and mitigation in high-volume, high-end enterprise systems, this paper uses a qualitative review method. The techniques discussed in this paperwork are the set of beneficial KPIs that are vital to making large workloads functional and optimal. The reason for choosing a qualitative review approach is based on the need to identify and categorize the research studies on these advanced procedures. In this regard, although each technique has theoretical and practical implications, the review offers a deeper insight into the challenges and, therefore, the opportunities that come with each of them. The purpose is to evaluate the possible practicality of these strategies and their effects on the efficiency, availability, and capability of functioning as an efficient real-time system for more excellent platforms.

Table 2: Performance evaluation criteria

| Metric | Description | Importance |
|---|---|---|
| **Scalability** | Ability to handle increasing transactions efficiently | High |
| **Response Time** | Time taken to resolve a collision | High |
| **Computational Overhead** | Processing power required to implement the method | Medium |
| **Complexity** | Difficulty in integrating the approach into existing systems | Medium |
| **Consistency** | Accuracy in maintaining transaction integrity | High |

The above table provides a structured way to evaluate different collision resolution methods

*Selection Criteria for Literature*

The papers used in this review are research papers, Academic Journals, Conference Papers, Books, and Papers written by professionals. For method comparison, the selection was made based on the novelty of the approach presented in the paper concerning collision resolution, advanced techniques such as those beyond locking, and the applicability of the proposed ideas to high-volume systems. The papers were then vetted to determine shared characteristics and issues and

the prevalent strategies in conflict management in distributed systems. The review also dwells upon the suitability of such techniques concerning multiple system contexts such as cloud environments, distributed databases, and high throughput transactional systems. The research adopts a three-stage method which starts with identifying collision resolution methods beyond traditional locking techniques followed by performance criteria evaluation and ends with selected approach comparison.

*Comparative Analysis*

To evaluate advanced collision detection and collision mitigation, this paper has adopted a comparative analysis methodology that aims to determine the relevant solutions depending on the high volume of the system requirements.

Table 3: Comparative analysis of the approaches

| **Approach** | **Scalability** | **Performance in High Contention** | **Latency** | **Real-Time Adaptability** | **Implementation Complexity** |
|---|---|---|---|---|---|
| **Two-Phase Locking (2PL)** | Low | Moderate | High | Poor | Low |
| **Optimistic Concurrency Control (OCC)** | High | Poor | Low | Good | Moderate |
| **Decentralized Algorithms** | High | Moderate | Moderate | Good | High |
| **Machine Learning Approaches** | High | High | Low | Excellent | Very High |
| **Hybrid Solutions** | Very High | Very High | Low | Excellent | High |

| **Method** | **Advantages** | **Disadvantages** | **Best Use Case** |
|---|---|---|---|
| **Two-Phase Locking (2PL)** | Prevents conflicts, ensures strict serialization | High contention leads to bottlenecks | Low-concurrency environments |

| Method | Advantages | Disadvantages | Best Use Case |
|---|---|---|---|
| **Optimistic Concurrency Control** | High throughput, reduces locking overhead | Frequent rollbacks in high-contention scenarios | Low-contention environments |
| **Decentralized Conflict Handling** | Scalable, eliminates single points of failure | Complex synchronization required | Distributed databases |
| **Machine Learning-Based Resolution** | Predictive conflict detection, adaptive | High computational cost | Real-time financial transactions |
| **Hybrid Approaches** | Combines benefits of multiple methods | Expensive to implement, requires tuning | High-traffic enterprise applications |

This review shows that strengths and weaknesses reflect how the strategy will work in a given environment [34]. However, the decentralized algorithms discussed in this work have their advantages with scalability; they may have several problems, such as clock synchronization and communication overheads. Artificial intelligence and neural networks always provide intelligent and changing procedures for solving conflicts; however, issues of highly powerful computation and massive amounts of data [35]. The use of several techniques proves to be more flexible; also, it often outperforms the single-technique solutions but comes with more issues involved. It is to determine the best practices in addressing the collisions that have been described by defining the techniques in light of scalability, complexity, and real-time responsiveness in this paper. Thus, the findings from this review will help in three ways: For one, they will assist in the identification of best practices to be used in collision resolution, and secondly, they will help identify the emergent trends that should be hiked in large-scale environments; and third, can serve in future research and development endeavours.

## Results and Discussion

### *Optimistic Concurrency Control (OCC)*

Optimistic Concurrency Control (OCC), another collision resolution strategy, is the contrary since resource locking does not occur during transaction processing [36]. Instead of obtaining the locks at the start of a transaction, OCC presupposes that concurrency should rarely happen, leaving it uncontrolled. At the commit phase alone, the system must check for such merge conflicts. In the case of scanning for conflicts, if no conflict is indicated, the transaction is

completed; otherwise, the transaction is rolled back and restarted again [37]. Overall, this mechanism enhances the capability of the system to handle a large number of transactions with more consistency since several transactions can be processed simultaneously without the use of locks. OCC is most effectively used in low-contention system environments where the probability of transaction contention is very low [38]. They do not involve locks repeatedly to lock or unlock, enhancing the system throughput and decreasing latency. Secondly, transactions do not need to wait for locks. Therefore, thousands of transactions can be processed simultaneously or many more at a given time, and consequently, the system's performance improves [39].

The probability of transaction success (P) in OCC can be modeled as follows [40]

$$P_{success} = 1 - \frac{C}{T}$$

Where:

- CCC = Number of concurrent transactions accessing the same resource.
- TTT = Total transactions executed in a given time frame.

For hybrid approaches, an adaptive weight function can be introduced:

$$W_{hybrid} = \alpha P_{OCC} + \beta P_{Decentralized}$$

Where α\alphaα and β\betaβ are dynamic adjustment factors based on transaction workload and contention level.

In these scenarios, OCC must make many attempts to complete these aborted transactions, resulting in longer transaction response time and fewer transactions per a certain amount of time. This is especially the case if many transactions abort and retry, if the conflicts are of long duration, or if there are many concurrently active transactions. In this respect, various improvements have been suggested to develop OCC [41]. One typical type of optimization is priority-based; in best practice, the transactions with the highest priority are processed first in case of conflict. This

generally means that if a transaction has a higher priority, it will be processed even if it interferes with other transactions of lower priority and causes them to commit an abort [42]. It prevents certain transactions from being heavily slowed down any time that contention emerges to ensure that there will always be a way of resolving it to continue with the computation [43]. They can also decrease the number of aborted transactions, specifically in a highly contentious environment, since it is possible to prioritize those committed transactions and prioritize those committed transactions with a higher probability of success.

***Decentralized Algorithms***

Timestamp Ordering (DTO) is more of a collision resolution technique [43]. In decentralized systems, there is no sense of a coordinator or lock manager that can identify collision; hence, decentralized approaches are considered best because they are more scalable. Instead, it means that each node in a distributed system may independently coordinate the concurrency management with others when needed [45]. This is the case because decentralized algorithms efficiently scale within large operational systems. Unlike centralized solutions, any node can decide the conflict between two nodes independently. Thus, the lock manager does not become a problem regarding performance and resource utilization [46]. Every node contains local-state information and protocols to recognize and work with conflicts that may arise with other nodes in the system. Similarly, this approach minimizes the chances of having many constraints that could slow down the system and increase the amount of transactions and resources processed.

For instance, vector clocks are a technique that may be adopted to log the relationship of events in a distributed system [47]. Every research centre has a vector of timestamps in the system and each time stamp possesses the local time in the node. In this transaction, the vector clock shows changes made when the transactions are performed in the system. Suppose the transactions interfere with each other; then, the vector clocks are used to decide the order of execution to eliminate interference. This approach is vital to ensure that several successive transactions are accomplished sequentially; this is mainly applied in a network of nodes where the nodes do not necessarily have a standard time [48]. Likewise, Distributed Timestamp Ordering (DTO) imposes global ordering of the transactions with the help of timestamps. DTO ensures that transactions are executed according to issues to eliminate the risk of concurrent transactions' interference as they are processed [49]. In this way, through this global ordering, DTO assists in keeping the

consistency of different distributed systems, which means that all nodes recognize the same series of transactions.

However, there is one shortcoming that comes with a decentralized algorithm despite the great opportunities it holds. Hence, one of the significant difficulties is clock synchronization. AA distributed systems may contain more than one node, and each node may have an individual time counter that eventually runs at different rates [42]. This synchronized time drift may cause a problem with the order of transactions because the time stamps do not correctly indicate the chronological order. To overcome this problem, precise synchronization among different nodes is critical and requires sophisticated ways to be applied to the nodes. A disadvantage of distributed algorithms is the problem of transmitting messages across the network [45]. Since each node is aware of and responsible for its state and data, transactions frequently need to contact other nodes to organize a search for conflicts. This should include a sound messaging system for the overhead of passing data from one node to another, which becomes a tall order in a large-scale system with many nodes [44]. Due to the expected high transaction rates, the established communication channel must be highly effective in reducing response time.

***Machine Learning Approaches***

Conflict-solvers hold great potential in increasing process throughput within large, heavy-load systems incorporating real-time, complicated workloads [46]. By withdrawing the models' ability to learn from past transactions by analysing patterns, one can determine conflicts and automatically control systems based on such patterns. Such models can understand different patterns arising from previous transactions and prevent conflict in the process in real-time [43]. It is important to note that by using machine learning-based approaches, adaptation to changes in the IT system can easily be made. Most conflicts are designed in the form of norms and standard procedures that lay down requirements for conflict resolution, and thus, unable to adapt to different levels of workloads in transaction flow [24]. In contrast, the given example of the machine learning models can develop an appropriate strategy depending on new information and then apply it while being sensitive to the given context and searching for a solution to the emerging conflict [44]. This facilitates the adaptation of performance with time, and the system can better counter changes in conditions.

For instance, it is possible to design algorithms and machine learning models in such a way that these offer the potential to predict when two transactions are most likely to clash by assessing the transaction type and the pattern in which specific resources are accessed, among other considerations [45]. Knowing the potential conflicts in advance can help to change the priorities of the transaction or make some changes in the allocation of resources, thus preventing conflict situations and enhancing the system's efficiency. Nevertheless, conflict resolution systems with the help of machine learning have several issues [36]. Training data is an essential element of accurate models, and although there is often plenty of data available, accumulating enough data for a model can be challenging, especially in proprietary or real-time systems. Moreover, training and running the machine learning models might require sizeable computational power, depending on the scale of the system [47]. The necessity for such predictions additionally increases computational problems, as the system should analyse large amounts of transaction data and make decisions on the fly.

*Hybrid Solutions*

The chosen integration of different conflict resolution methodologies helps address collisions in high-volume systems depending on various situations [48]. These models combine strategies, including OCC, other distributed algorithms, and machine learning, to develop a superior system. When integrating the best qualities of multiple approaches, hybrid solutions can be proven more efficient in several situations [49]. For instance, concurrency and distributed scalability can be achieved when using OCC with decentralized algorithms. OCC can be used for transactions with much less conflict, and decentralized algorithms can take care of distributed resources properly, which, in turn, can organize the system to handle many transactions [40]. This hybrid can be used since it is a flexible system that can change depending on the conditions and the amount of work within it. In the same way, the application of machine learning in resource locking brings dynamic prioritization of the transaction by capturing real-time data.

While machine learning can learn to learn transactions and modify the modifying parameters, locking prevents conflicts according to traditional methods. This approach is very suitable for organizing workloads because it will always be able to scale depending on the number of queries; it also can be more dynamic to the number of queries that may be processed in real-time [41]. Indeed, even if the hybrid models present the main benefits concerning flexibility and performance, they are more difficult to enforce and manage. Any attempt at integrating many

conflict resolution techniques implies a perfect design and proper coordination of the various activities involved in an integration process. As a result, there are usually a lot of problems encountered when integrating the different activities. [42] Also, if the solution is complex, it will require more money for its implementation and further support, making it less appropriate for particular systems regarding hybrids.

## Conclusion

The research investigated traditional locking constraints in spatial distribution while assessing new-resolution techniques that relied on OCC and decentralized algorithms and machine learning schemes as well as hybrid systems. Research evidence shows OCC offers good performance in situations with minimal contentions though hybrid solutions between OCC and decentralized algorithms demonstrate better adaptability and better performance results. Machine learning operates efficiently for conflict resolution yet needs sophisticated computer systems to work effectively.

Table 4: Research Areas

| Research Area | Open Challenges | Potential Solutions |
|---|---|---|
| **Scalability in High-Load Systems** | Handling exponential transaction growth | AI-driven adaptive conflict resolution |
| **Hybrid Model Efficiency** | Balancing trade-offs between methods | Optimized resource allocation |
| **Computational Overhead** | Reducing processing costs for ML-based methods | Hardware acceleration (e.g., GPUs) |
| **Blockchain for Collision Resolution** | Ensuring decentralization without latency | Smart contract optimization |

Research in the field should concentrate on uniting smart contracts from blockchain systems with collision detection methods to create secure and efficient distributed network platforms. Advanced strategies will enable large-scale enterprise systems to implement real-time tools for scalable robust collision management.

## References

[1] Nookala, G.,. Real-Time Data Integration in Traditional Data Warehouses: A Comparative Analysis. *Journal of Computational Innovation*, *3*(1). (2023)

[2] Fugkeaw, S. and Hak, L., 2024. PPAC-CDW: A privacy-preserving access control scheme with fast OLAP query and efficient revocation for cloud data warehouse. *IEEE Access*.

[3] Hassan, M.H., Darwish, S.M. and Elkaffas, S.M., 2023. Type-2 Neutrosophic Set and Their Applications in Medical Databases Deadlock Resolution. *Computers, Materials & Continua*, *74*(2).

[4] Eldeeb, T., Xie, X., Bernstein, P.A., Cidon, A. and Yang, J., 2023. Chardonnay: Fast and general datacenter transactions for {On-Disk} databases. In *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)* (pp. 343-360).

[5] Lin, Q., Zhang, M., Ren, J. and Hua, Q., 2023. Investigation on a new type of latching mechanism on the satellite-rocket docking system and locking dynamic analysis. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, *237*(4), pp.992-1003.

[6] Naidu, M.H., Beohar, A., Thomas, S.G. and Subramanian, U.A., 2024, January. Design, Synthesis and Optimization of Latch Mechanism Using Mathematical Model for Closing Doors in Aerospace Vehicles. In *International Conference on Mechanical Engineering* (pp. 773-791). Singapore: Springer Nature Singapore.

[7] Wang, M.Y., Hang, L.B., Zhong, C.L. and Qin, P.C., 2024. Research on a novel compliant mechanism applied in the vehicle side door latch and stability analysis via Lyapunov exponents. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, *238*(14), pp.6977-6992.

[8] Peng, J. and Hou, C., 2025. Experimental and numerical study on a novel self-locking inter-module connection with spring-loaded plunger latches: Under axial tension. *Engineering Structures*, *324*, p.119364.

[9] Wang, H. and Zhu, S., 2024. Latching control: A wave energy converter inspired vibration control strategy. *Mechanical Systems and Signal Processing*, *208*, p.110912.

[10] Fernandez, G.I., Gessow, S., Quan, J. and Hong, D.W., 2024. Self-aligning rotational latching mechanisms: Optimal geometry for mechanical robustness. *Journal of Mechanisms and Robotics*, *16*(1), p.011007.

[11] Wang, H. and Zhu, S., 2025. Latching control for pendulum tuned mass damper: Theoretical analysis and proof-of-concept experiment. *Mechanical Systems and Signal Processing*, *224*, p.112158.

[12] Alhajri, A., 2023. *Performance and forensic applications of a novel optimistic concurrency control algorithm* (Doctoral dissertation, University of Warwick).

[13] Neha and Banita, 2024, November. High-Performance Concurrency Control in Multi-User Database Systems: Analysis and Optimization Strategies. In *2024 3rd Edition of IEEE Delhi Section Flagship Conference (DELCON)* (pp. 1-5). IEEE.

[14] Waudby, J., 2024. *High Performance Concurrency Control and Commit Protocols in OLTP Databases* (Doctoral dissertation, Newcastle University).

[15] Alhomssi, A.A., 2024. *Concurrency Control for High-Performance Storage Engines*. Friedrich-Alexander-Universitaet Erlangen-Nuernberg (Germany).

[16] Ali, A.M., Nabot, A., Jebreen, I., Alauthman, M., Alangari, S., Almomani, A., Chamola, V. and Aldweesh, A., 2025. Balancing Consistency and Performance in Edge-Cloud Transaction Management. *Computers in Human Behavior*, p.108601.

[17] Alhajri, A., Jhumka, A. and Kirk, R., 2023, March. OCC2T: An Early-Read Dual-Track OCC Algorithm For Mixed Mode Systems. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing* (pp. 321-330).

[18] Thomasian, A., 2024. Heterogeneous Data Access Model for Concurrency Control and Methods to Deal with High Data Contention. *arXiv preprint arXiv:2404.02276*.

[19] Theodorakopoulos, L., Theodoropoulou, A. and Halkiopoulos, C., 2024. Enhancing decentralized decision-making with big data and blockchain technology: A comprehensive review. *Applied Sciences*, *14*(16), p.7007.

[20] Feng, B., Zhou, Q., Xing, J. and Yang, Q., 2024. Distributed chaotic bat algorithm for sensor fault diagnosis in AHUs based on a decentralized structure. *Journal of Building Engineering*, *95*, p.110031.

[21] Kopic, A., Perenda, E. and Gacanin, H., 2024. A collaborative multi-agent deep reinforcement learning-based wireless power allocation with centralized training and decentralized execution. *IEEE Transactions on Communications*.

[22] Liu, J., Zhang, N., Zha, L., Xie, X. and Tian, E., 2023. Reinforcement learning-based decentralized control for networked interconnected systems with communication and control constraints. *IEEE Transactions on Automation Science and Engineering*, *21*(3), pp.4674-4685.

[23] Beltrán, E.T.M., Pérez, M.Q., Sánchez, P.M.S., Bernal, S.L., Bovet, G., Pérez, M.G., Pérez, G.M. and Celdrán, A.H., 2023. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*, *25*(4), pp.2983-3013.

[24] Hussein, Z., Salama, M.A. and El-Rahman, S.A., 2023. Evolution of blockchain consensus algorithms: a review on the latest milestones of blockchain consensus algorithms. *Cybersecurity*, *6*(1), p.30.

[25] Rezaee, M.R., Hamid, N.A.W.A., Hussin, M. and Zukarnain, Z.A., 2024. Comprehensive review of drones collision avoidance schemes: Challenges and open issues. *IEEE Transactions on Intelligent Transportation Systems*.

[26] Xie, S. and Zhang, J., 2024. Handling highly imbalanced data for classifying fatality of auto collisions using machine learning techniques. *Journal of Management Analytics*, *11*(3), pp.317-357.

[27] Mondal, S. and Goswami, S.S., 2024. Machine learning applications in automotive engineering: Enhancing vehicle safety and performance. *Journal of process management and new technologies*, *12*(1-2), pp.61-71.\

[28] Soori, M., Arezoo, B. and Dastres, R., 2023. Artificial intelligence, machine learning and deep learning in advanced robotics, a review. *Cognitive Robotics*, *3*, pp.54-70.

[29] Patalas-Maliszewska, J., Dudek, A., Pajak, G. and Pajak, I., 2024. Working toward solving safety issues in human–robot collaboration: a case study for recognising collisions using machine learning algorithms. *Electronics*, *13*(4), p.731.

[30] Thakur, A. and Mishra, S.K., 2024. An in-depth evaluation of deep learning-enabled adaptive approaches for detecting obstacles using sensor-fused data in autonomous vehicles. *Engineering Applications of Artificial Intelligence*, *133*, p.108550.

[31] Azevedo, B.F., Rocha, A.M.A. and Pereira, A.I., 2024. Hybrid approaches to optimization and machine learning methods: a systematic literature review. *Machine Learning*, *113*(7), pp.4055-4097.

[32] Mikram, H., El Kafhali, S. and Saadi, Y., 2024. HEPGA: A new effective hybrid algorithm for scientific workflow scheduling in cloud computing environment. *Simulation modelling practice and theory*, *130*, p.102864.

[33] Jia, J., Liang, W. and Liang, Y., 2023. A review of hybrid and ensemble in deep learning for natural language processing. *arXiv preprint arXiv:2312.05589*.

[34] Hashmi, E., Yamin, M.M. and Yayilgan, S.Y., 2024. Securing tomorrow: a comprehensive survey on the synergy of Artificial Intelligence and information security. *AI and Ethics*, pp.1-19.

[35] Sahoo, S.K. and Goswami, S.S., 2023. A comprehensive review of multiple criteria decision-making (MCDM) Methods: advancements, applications, and future directions. *Decision Making Advances*, *1*(1), pp.25-48.

[36] Singh, A.A., Khan, A., Mehrotra, S. and Nawab, F., 2023. TransEdge: Supporting Efficient Read Queries Across Untrusted Edge Nodes. *arXiv preprint arXiv:2302.08019*.

[37] Faria, N. and Pereira, J., 2023. MRVs: enforcing numeric invariants in parallel updates to hotspots with randomized splitting. *Proceedings of the ACM on Management of Data*, *1*(1), pp.1-27.

[38] Zhao, Z., Zhao, H., Zhuang, Q., Lu, W., Li, H., Zhang, M., Pan, A. and Du, X., 2023. Efficiently supporting multi-level serializability in decentralized database systems. *IEEE Transactions on Knowledge and Data Engineering*, *35*(12), pp.12618-12633.

[39] Flowers, S., 2023. Designing and Implementing Cloud-native Applications Using Microsoft Azure Cosmos DB. *Gede Suacana, IW, Sudana, IW, Wiratmaja, IN, & Rukmawati, D.(2024). Urban Land Consolidation Policy in the Context of Creating a Good Environment According to Spatial Planning in Indonesia. Journal of Wood Science*, *3*(2).

[40] de Freitas, D.C.A., 2023. *Towards Causal Consistency in Read-Heavy Cloud-Native Systems* (Master's thesis, Universidade do Porto (Portugal)).

[41] Kröll, M. and Burova-Keßler, K., 2023. Creativity, Innovation and Entrepreneurship, Vol. 74, 2023, 95–104. *Creativity, Innovation and Entrepreneurship*, *95*.

[42] Jang, E.H.B., 2024. *Infrastructuring at the Margins: Studies in Community Networking* (Doctoral dissertation, University of Washington).

[43] Mabrouk, H., Chen, Y., Adams, M., Burnie, J., Millward, M., Van Hoa, T., Minh, T.V.N., Pundale, N. and Kashin, A., 2023. Special issue on blockchain. *European Journal of Law Reform*, *25*, pp.1-2.

[44] De Silva, D.P., 2023. *System for Monitoring Physical Therapy Exercises Interactive Mobile Application for Monitoring Wrist Device Measurements* (Master's thesis, Universidade NOVA de Lisboa (Portugal)).

[45] Nadeem, A., 2024. *Understanding adversary behavior via XAI: Leveraging sequence clustering to extract threat intelligence* (Doctoral dissertation, TU Delft).

[46] Jánki, Z.R., 2023. *Telemedicine Engineering: Modeling and Analyzing Data Quality in Telemedicine Systems and the Impact of Novel Design Patterns on Productivity in Modern Web Application Development* (Doctoral dissertation, Szeged University (Hungary)).

[47] Gussner, S., 2024. *Requirements Analysis, System Architecture and Evaluation of a Privacy-First Web Application Framework* (Doctoral dissertation, Technische Universität Wien).

[48] Abolfathi, M., 2024. *Enhancing Encrypted Network Traffic Security Against Advanced Traffic Analysis Attacks* (Doctoral dissertation, University of Colorado at Denver).

[49] Tiusanen, R., Heikkilä, E., Välisalo, T. and Malm, T., 2023. Safety analyses on the use of tram doors in GoA1 and GoA4 autonomy levels.