PERFORMANCE ANALYSIS OF VARIOUS CONCURRENCY TECHNIQUES ON THE DISTRIBUTED TRANSACTIONS

V. Vasanthraj[1], Dr.K.Ravikumar[2]

[1]Research Scholar, Department of Computer Science, Tamil University, Thanjavur, India

[2]Head & Assistant Professor, Department of Computer Science, Tamil University, Thanjavur, India

[1]mscvasanthcs@gmail.com, [2]ravikasi2008@gmail.com

Abstract— **Distributed computing is meant by the system is distributed in the local network or geographical. In the enterprise, distributed computing has often meant putting various steps in business processes at the most efficient places in a network of computers. Distributed computing has major impact of the networking communications. Fault tolerance, Scalability, Resource sharing, Transparency are some of the characteristics of Distributed Computing. There are some of the existing methodologies were investigated about the said characteristics. However the modern era of communication is being wanted to implement the accuracy and need to reduce the error and data loss. Concurrency control is the major role to govern the multi user transaction in the database .The concurrency property of the distributed computing is executing simultaneously such that collection of manipulated data item is left in a consistent state. In the existing system the single tier methodology with multithread is not giving better results among the multiple transactions. The proposed model is promised to full fill the secure and simultaneous transactions with multi-tier multithreading. The proposed mechanism has worked in the various database applications especially in distributed networks which gives good performances. The advantage of the proposed model satisfies the improvement in server responsiveness, minimum usage of system resources. The computation time among the multiple transactions is fast which compared with the existing model. The proposed model has simple programming structure to implement the results in the various distributed database transactions**

Keywords **-Distributed Systems, Concurrency control, Multi- tier multithreading, Distributed transactions.**

I. INTRODUCTION

Distributed computing is a computing concept that, in its most general sense, refers to multiple computer systems working on a single problem. In distributed computing, a single problem is divided into many parts, and each part is solved by different computers. As long as the computers are networked, they can communicate with each other to solve the problem. If done properly, the computers perform like a single entity [2]. The ultimate goal of distributed computing is to maximize performance by connecting users and IT resources in a cost-effective, transparent and reliable manner. It also ensures fault tolerance and enables resource accessibility in the event that one of the components fails. The idea of distributing resources within a computer network is not new. This first started with the use of data entry terminals on mainframe computers, then moved into minicomputers and is now possible in personal computers and client-server architecture with more tiers[2].

The main goal of a distributed system is to make it easy for users to access remote resources and to share them with others in a controlled way. It is cheaper to share a printer by several users rather than buying and maintaining printers for each user. Concurrency is the main characteristics in the distributed environment to make the database transactions without any loss.

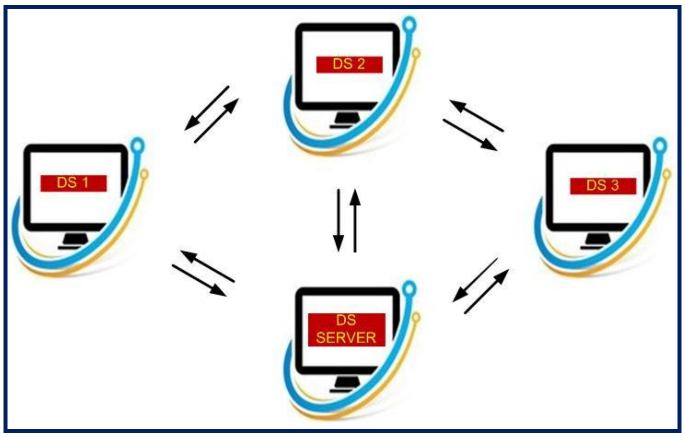

Fig. 1 Distributed System

The idea of distributing resources within a computer network is not new [3]. This first started with the use of data entry terminals on mainframe computers, then moved into minicomputers and is now possible in personal computers and client-server architecture with more tiers.

The main goal of a distributed system is to make it easy for users to access remote resources and to share them with others in a controlled way. It is cheaper to share a printer by several users rather than buying and maintaining printers for each user. Concurrency is the main characteristics in the distributed environment to make the database transactions without any loss. Concurrency control is a very important issue in distributed database system design. This is because concurrency allows many transactions to be executing simultaneously such that collection of manipulated data item is left in a consistent state. Concurrency control is the activity of co- coordinating concurrent accesses to a data- base in a multiuser database management system (DBMS). Concurrency control permits users to access a database in a multi- programmed fashion while preserving the illusion that each user is executing alone on a dedicated system.

Concurrency is meant by the multiple user can affect the database transactions without any violations. This property is used to isolate from the normal file system. Concurrency allows the multiple users request to access the same database item [3]. There are some various algorithms are involved in the concurrency management like shared lock, Exclusive lock, Time stamp and so on. Concurrency control is the safe guard against transaction interference. It enforces the isolation among the variousmutual exclusion. The concurrency control enables the database preservative through consistency on the executions on the database transactions [3].

II. BACKGROUND AND RELATED WORK

A. Time Stamp ordering protocols:

It ensures that any conflicting read and write operations are executed in time stamp order. There are two operations on the same data item, the algorithm compares the time stamps of two different operations that not violate each other's. Each transaction is issued a time stamp when it enters into system [4]. If an old transaction Ti has time stamp TS (T_i), a new transaction T_j is assigned time –stamp such that $TS(T_i) < TS(T_j)$. The protocol manages concurrent execution such that the time – stamps determine the serializability order. W- time stamp (Q) is the largest time stamp of any transaction that executed write (Q) successfully. R – Time stamp of any transaction that executed read (Q) successfully [4].

B. Two phase locking protocols:

There are two phases in the algorithm which they are growing phase and shrinking phase. All the executions are done in these phases. The growing phase takes the responsible to obtain locks and the shrinking phase takes the charge to release the locks [5]. A transaction locks an object before using it. When an object is locked by another transaction the requesting transaction must wait. When a transaction releases a lock, it may not request another lock. If T wants to read an object, it first obtains an S lock. If T wants to write an object, it first obtains an X lock. If T releases any lock, it can acquire no new locks. All these are done transparently to the user by the DBMS. It guarantees the serializability [6]. Simple the growing phase may obtain new locks and may not release any locks. Same like the shrink phase may release the locks or may not obtain locks.

C. Binary lock & Shared lock:

A Binary lock on a data item can either lock or unlocked states. A shared lock is also called a Read-only lock. With the shared lock, the data item can be shared between transactions. This is because you will never have permission to update data on the data item [6].For example, consider a case where two transactions are reading the account balance of a person. The database will let them read by placing a shared lock. However, if another transaction wants to update that account's balance, shared lock prevent it until the reading process is over.

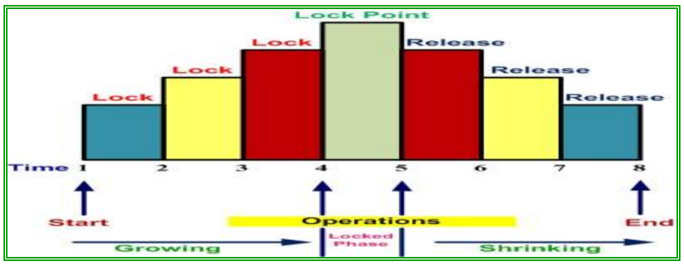

Fig. 2 Locking Protocols

III. EXISTING METHODOLOGY

A. Concurrency Control Algorithm Access:

The concurrency control algorithm improves the performance of real-time systems by evaluating data-deadline and transactions execution time, adjusting transaction validation rule and committing order. Theoretical analysis and example results demonstrate that the new algorithm can outperform the previous ones. The data set will be declared before transaction execution [5]. The temporary consistency will be checked for ensuring the transaction accessing to correct data. The concurrency control algorithm the key factors is adjusting validation rules during validation phase, which schedules the priority transactions that near to complete by asserting validation factor.

B. Mutual Authentication protocol:

This protocol has been designed which is safe from tracking, replay attack and spoofing using time stamp which is generated also in Back-End DB unlike other existing protocols. The reader sends query not only to the tag but to DB as well so DB generates time stamp value. Although all the information between the tag and the reader which is insecure channel is captured and used for replay attack, the time stamp value of the DB is renewed for every [6]. Tag only saves ID value. When the tag exists within the sensitive detection range, the reader simultaneously sends a query to both the tag and the DB. Tag, after receiving the query saves the query together with the transmitted reader random number in arbitrary storage in the tag [7]. DB receives the query from the reader and generates time stamp and sends it to the reader. Reader receives the time stamp from the DB and immediately sends it to the tag. Through this process, the tag will acquire the time stamp from the DB and the random number from the reader.

C. Global Concurrency Control in Multi Database Systems:

The site-locking global concurrency control method that uses the lock operation for access. The basic idea of site-locking method is that a global transaction cannot execute concurrently with otherglobal transactions if it has a possibility of indirect conflict. Our global concurrency control method uses the lock operation for indirect and direct conflicting operations, like basic 2PL[7]. We can easily find that the read operation of global transaction that is executed on the global data item is not in direct conflict with any local transactions. Therefore, the read operation of global transaction for global data item cannot make an indirect conflict with other operations. The remainder operations may causean indirect conflict with other global transactions because it may be in direct conflict with the local transaction. We define the site-conflict between two global transactions and propose site-locking method [8]. To present our site-locking method, we need some notation. Locking granularity is a site, rather than a data item. An operation of global transaction is submitted to a site either reading or for writing a data.

The existing algorithm is an effort to achieve the concurrent execution of transactions without leading to starvation. The algorithm is based on multi-version concurrency control protocols. In this algorithm version locks are attached to remove the retrieval anomaly and a logical degradation of locks is considered to reduce the priority of low level transaction and so as to increase the high level transaction's priority. This degradation leads to limit the starvation to some extent and also eliminates the covert channel. The conflict data-items are stored in c-data- items [8]. This data structure contains the set of data-items, read by high level secure transaction and a new version of these data-items is written by low level secure transactions. The retrieval anomaly is totally eliminated. To make the approach starvation free function is used. When the value of virtual lock becomes the un classified (the lowest security level), it starts executing the corresponding transaction rather than the transaction whose actual lock value is lock. In this way, it may introduce some covert channel [9].

IV. PROPOSED MODEL

The applications of database and other dbms applications have helped to manage the transaction by the help of various protocols. The many of the existing protocols has ensured to govern the transaction without any violations. The proposed model has a layer of multithreading with multi tire guard to ensure the transactions over the dbms applications. The multi-tier part has helped to monitor all dbms queries and services. The multi-threading is meant that each and every request from the application can split up into many threads with time stamps. All the requests from the applications have transferred through the proposed model of multi tire threading which can split up the various threads.

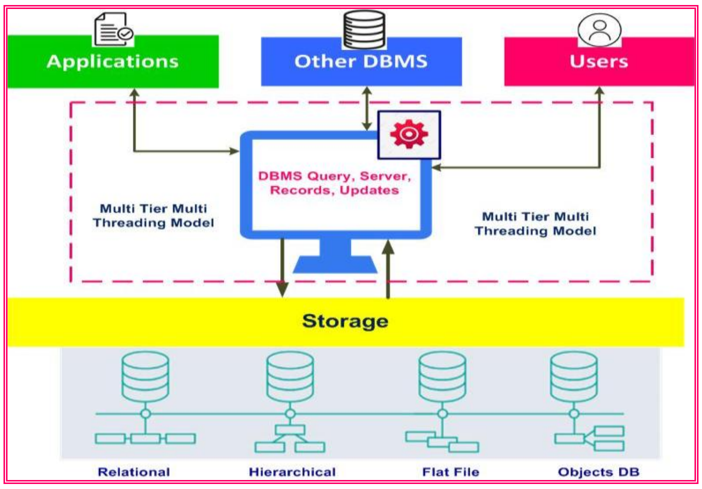
Applications
Other DBMS
Users
DBMS Query, Server, Records, Updates
Multi Tier Multi Threading Model
Multi Tier Multi Threading Model
Storage
Relational
Hierarchical
Flat File
Objects DB

Figure: 4.1 Concurrency model with multi-tier multi-threading

Threads are actually used to help the concurrency among the various processes. In case of more threads on the various processes, the concurrent execution has done in the execution. Likewise if the process has multi thread with multiple core machine the concurrency may increase as per the theory. The increases and decreases on the concurrency control may decide by the number of cores in the machine. If the machine has single core on the single thread, the performance may low to compare with multi thread with multi core.

In the same method the proposed model may apply on the dbms applications to the transactions. If a single request with single thread may have some delay when working with multi thread. The many requests from the various applications to work with the database operations through the existing system may normal. But they may not deal with the speed or increases on the concurrency. The proposed model has promised to handle all the requests from the applications through the multi tire model. The multi tire model has worked with the multi thread concept like multi core machines.

The distributed database is fully different from the ordinary database transactions. The distributed server may handle maximum requests from the server with no violations. The consistency and concurrency is the main thing in the distributed systems. The consistency may affect if the concurrency is not done. The concurrency of the ddbms may satisfied by the speed and accuracy of the transactions. The proposed model has done in various ways to handle multiple requests through the multi tire model. The multi thread may split the various small threads into multiple threads.

The multiple threads are working with parallel on the various time stamps. The proposed model split up the various requests among the server. If the machine with multiple core may easily handle the entire request but in the single core machine it is difficult. The multi-threading has ensure that all threads were split with time stamp. Based on the time stamp unit the resources may allocate and the execution may be done by according the minimum cpu usage and maximum throughput. On a multicore machine, this means two threads can really run in parallel, doing twice the work they'd do running one at a time. Ideally, on a 4 core machine, with 4 threads you'll get almost 4 times as much work done as with a single thread.

The proposed model is illustrated in figure 4.1, the multi tire with multi thread unit which get the requests from the applications.

V. CHALLENGES IN SECURITY

Security is important in distributed system which compared with the centralized system. The number of users on the distributed system is getting increased on day by day. At the same time we need to build the string security over the DS architecture. Commonly there are two types of attacks in the distributed system [11]. They are:

- Passive Attack
- Active Attack\

A. Passive Attack:

The passive attack is meant by intruder which they involve to get the information without any authorization. They may hold the private information of the System. The main aim of the passive attack is to obtain the information from the attacks. The release of message content is meanttelephone conversation, mail reading or nay confidential messages. The type of passive attack is traffic analysis. Masking of content may use in this traffic analysis type. The intruder may determine the location and identity of communicating hosts and may observe the frequency and length of messages being exchanged [11].

B. Active Attack:

Active attacks in the distributed system are a big challenge to face among the world wide communication. Simply the unknown or unauthorized person may attempt to insert the data in the existing data. The data may be altered or deleted by the hackers. Not only these types of challenges over there. An attacker can affect the protocol transmission, credential types, service provider and so on. In the active attack securities, there are mainly three types of attacks are considerable.

- Denial of Service

- Masquerading or Spoofing
- Packet Replay

The denial of service is meant by the services of the system did not make any response to the destination. Simply the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Masquerading or spoofing is meant that the actual sender is not claimed the sender. The sender does not take any responsibility for the sending actions. The past message pocket is transmitted over in order to gain access or otherwise cause damage. This property is known as packet replay in the distributed system [12].

In the security characteristics of distributed system should take care over the encryption, authentication, authorization and auditing. These characteristics must be taken to restrict some serious affect like active and passive attacks. Security features should describes the actions the entities in a distributed system allowed to take and which ones may prohibited. Moreover some of the design issues on the distributed system are focus on data control, layering of security mechanism and simplicity. The security methodology on the system should overcome the above said issues over the communication [12].

VI. Advantages of Multithreading

For a single thread process, whenever a blocking system call is executed, the process as whole is blocked. Using the multithread process, a program can process more than two tasks at same time, for example the spread sheet program. Multithreading also makes it possible to exploit parallelism when executing the program on a multiprocessor system. Thread switching can sometimes be done entirely in user space. In the web client multithreading is the inclusion of more than one unit of execution in a single process. In a multithreaded application, multiple simultaneous calls can be done made from the same process.

A. Concurrency:

In the multithread process on a single processor, the processor can switch execution resources between threads, resulting in concurrent execution [12]. In the same multithreaded process in a shared-memory environment, each thread in the process can run on a separate processor at the same time, resulting in parallel execution. When the process has fewer or as many threads as there are processors, the threads support system in conjunction with the operating environment ensure that each thread runs on a different processor. For example, in a matrix multiplication that has the same number of threads and processors, each thread (and each processor) computes a row of the result. In the physical concurrency is meant by the multiple processors on the same machine. They may distribute across the networked machines. In case of logical concurrency the multithread process may defined as illusion process or partial parallelism. Simply the physical concurrency is depended upon the multiple independent processors or multiple threads [12]. The logical concurrency or quasi concurrency is the same like but it depends upon time sharing of one processor.

B. Resource sharing:

In the DS platform resource sharing has important role to distribute and share the sources between process by the help of multithreading. Threads share the memory and other resources of the process to which they belong. Multithreading is an interactive application that must allow the program to continue running even if part of its blocked or doing lengthy operation. Threads may be running in parallel on different processors. Not only done these by the multithreading even also some important characteristics like responsiveness, economy and utilization of multiprocessors.

C. Speedup and Efficient communication:

The speed up the every process by the multithreading gives the much better experience to the end user. And also the threads of a specific process can communicate with each other efficiently because of the shared address space.

D. Multithreads in RPC:

In the remote procedure communication, multithread has helped much better performance in communication. The client calls a local procedure on the client stub and the stub calls the marshalls. The server stub unmarshalls the call and the arguments from the client. The server stub calls the actual procedure on the server. The server stub marshalls the reply and sends it back to the client. Thus the way the multithread satisfies the remote communication in the distributed process management.

VII. Results & Discussion

Slack time is meant by the interval time which occurs between the process completion and need of actual start time. It signifies the measure of time every movement can be deferred without abusing the whole task term. The leeway of a movement can be determined as the distinction between its most recent beginning and soonest start time, or on the other hand, as the contrast between its most recent and most punctual completing time. The hit ratio is meant by the calculation of cache hits and comparing them with how many total content requests were received. A miss ratio is the flip side of this where the cache misses are calculated and compared with the total number of content requests that ware received. Cache hit ratio is a measurement of how many content requests a cache is able to fill successfully, compared to how many requests it receives. Disk Access Time is defined as the total time required by the computer to process a read/write request and then retrieve the required data from the disk storage.In the case of disk storage, the access time is the average time taken for a disk drive to provide the first byte of data, measured from the time the host issues a read command. Overhead is any combination of excess or indirect computation time, memory, bandwidth, or other resources that are required to perform a specific task. It is a special case of engineering overhead. Overhead can be a deciding factor in software design, with regard to structure, error correction, and feature inclusion.

Table 7.1 Various concurrency control techniques analysis

Existing Techniques	Enforce Isolation	DB Consistency	Resolve read write Conflict	Success %	Fail %
2PL	84	77	75	**78**	22
Time Stamp	80	68	58	**69**	31
Optimistic	70	64	52	**62**	38
Multiversion	92	86	86	**88**	12

Let us take the analysis of various concurrency controls in the database transactions with subject to the constraints like isolation, consistency and read write conflict. These three properties are not the limitations for the database transactions. But however these are main transactions to deal with the transaction success rate and failure rates. Isolation is meant that the transactions have been affected and that should be visible to the other concurrent dbms users and the system. The integrity of the database is satisfied by this isolation property. And the other one main thing is consistency of the database. The entire transactions of the database should be affect in the user system and as well as the main database. But in the every transaction there is no more any violates in integrity rules. This is meant by the property of dbms integrity. The unrepeatable reads are knows as read write conflicts in the dbms transactions. When the transaction is going to be done before that all the dbms integrity rules should be satisfied by the constraints like consistency and isolation. In the above table 8.1 shows that the various concurrency techniques among the amount of some transactions. In this table we have focussed the success and fail rates of the existing concurrency techniques of 2PL, Time stamp, Optimistic and multiversion controls. We are taking the three important constraints of isolation, consistency and read write commits. In example for the 100 transactions multiversion technique takes the most satisfies percentage of success rate. And the other hand Optimistic methods gives the not so much performances to handle the read write conflicts. In overall, multiversion took the first priority next to the 2PL method. The 2PL method gives the less fail rate with compared to the time stamp and optimistic techniques. Let us illustrate through the graph which follows in Figure 8.1.

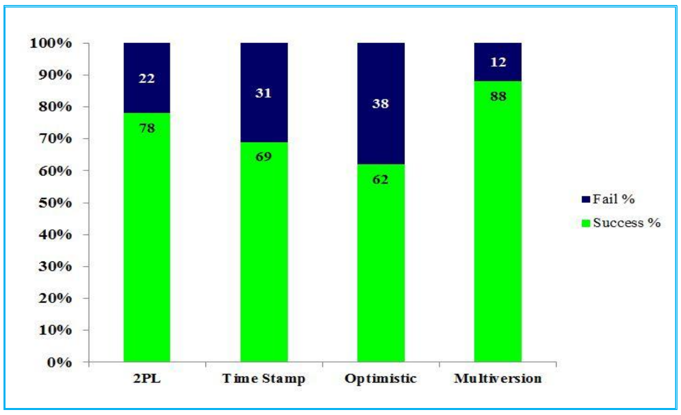

Figure 7.1 Success / fail rates of various concurrency techniques

The graph (Figure 7.1) has shown that the comparison of the various techniques involved in the concurrency techniques among the distributed database transactions. The multiversion technique has the high peek in success rate other than the 2PL. The optimistic and time stamp techniques are the more or less equal with their performances. The fail rate is also compared between 2PL and the multiversion techniques. The multiversion has less failure rate than the other techniques with respect to the consistency, read – write conflicts and isolation properties. The performance wise ratio between the 2PL and multiversion techniques takethe competition to achieve and satisfies the successful transaction of the user and as well as in the main systems.

IX. Conclusions

Partial failures of the database transactions are the major challenges in the distributed systems. One node can destroy while others doing job. The occasional network transactions may be late or lost by these kinds of failures. The proposed research may overcome in satisfies way to reduce or overcome these kind of part time failures. The switch failures should be carried out in the format of doing interrupt in the communication between some others replicas. The model may propose to implement the successful transactions among the ubiquitous distributed systems. The model may evaluate the dynamic system of communication with respect to the database transaction concurrency controls. The reliability on the database transactions on the distributed system may be evaluated to satisfy all the issues over on the partial and switch over transactions. The research model may be introduced to carry out the major securities like cryptography methods and channel issues. The authentication is the main design issue on the distributed system to do the transactions without failures. The model should carry out also leakages and tampering on the message security. These issues are main challenges to implement the successful concurrency techniques in the distributed computing transactions. The concurrency has the major role to satisfy the multitasking especially in the multi user database transactions. The quality of service is which meant by the successful transactions in the multi based user transactions in the user point of view as well as in the main systems. The model may help to design the openness which meant by the resource sharing services in the distributed systems and the degree of system may be increased in the future model.

References

[1] Tanabe, Takayuki, et al. "An Analysis of Concurrency Control Protocols for In-Memory Databases with CCBench (Extended Version)." arXiv preprint arXiv:2009.11558 (2020).

[2] SilaOzenGuclu, TanirOzcelebi, Johan Lukkien, "Distributed Fault Detection in Smart Spaces Based on Trust Management,"in Proceedings of the 2016 Elsevier Science Direct 7th International Conference (ANT 2016), pp.66-73. 2016.

[3] Barghouti, Naser S., and Gail E. Kaiser. "Concurrency control in advanced database applications." ACM Computing Surveys (CSUR) 23.3 (1991): 269-317.

[4] Bang, Tiemo, et al. "The tale of 1000 cores: An evaluation of concurrency control on real (ly) large multi-socket hardware." Proceedings of the 16th International Workshop on Data Management on New Hardware. 2020.

[5] Crowe, Malcolm, and Fritz Laux. "Reconsidering optimistic algorithms for relational DBMS." arXiv preprint arXiv:2110.03028 (2021).

[6] Gabriel, Bariyira Christopher, and Ledisi G. Kabari. "HYBRIDIZED CONCURRENCY CONTROL TECHNIQUE FOR TRANSACTION PROCESSING IN DISTRIBUTED DATABASE SYSTEM." (2020).

[7] Bang, Tiemo, et al. "AnyDB: An Architecture-less DBMS for Any Workload." arXiv preprint arXiv:2009.02258 (2020).

[8] Chaudhry, Natalia, and Muhammad MurtazaYousaf. "Concurrency control for realtime and mobile transactions: Historical view, challenges, and evolution of practices." Concurrency and Computation: Practice and Experience (2021): e6549.

[9] Pandey, Sarvesh, and UdaiShanker. "Performance issues in scheduling of real-time transactions." International Conference on Database Systems for Advanced Applications. Springer, Cham, 2021.

[10] Guo, Zhihan, et al. "Releasing Locks As Early As You Can: Reducing Contention of Hotspots by Violating Two-Phase Locking." Proceedings of the 2021 International Conference on Management of Data. 2021.

[11] Sethi, Jasmine, ShashankSrivastava, and DivyaUpadhyay. "A Review on P2P File System Based on IPFS for Concurrency Control in Hadoop." Emerging Technologies in Data Mining and Information Security. Springer, Singapore, 2021. 863-872.

[12] Daymude, Joshua J., Andréa W. Richa, and Christian Scheideler. "The Canonical Amoebot Model: Algorithms and Concurrency Control." arXiv preprint arXiv:2105.02420 (2021).