DEVELOPMENT OF NEURAL NETWORKS FOR
CONCURRENCY CONTROL IN COMPUTER AIDED
DESIGN DATABASE

Synopsis
Submitted by
MUTHUKUMARAVEL A.

In partial fulfillment for the award of the degree

of

DOCTOR OF PHILOSOPHY

Under the guidance of
Dr. Purushothaman S.
BE, ME, PhD (Indian Institute of Technology Madras)

VELS UNIVERSITY
CHENNAI - INDIA

VELS INSTITUTE OF SCIENCE, TECHNOLOGY AND ADVANCED STUDIES (VISTAS)
estd. u/s 3 of the UGC Act, 1956

DEPARTMENT OF MCA
SCHOOL OF COMPUTING SCIENCES
PALLAVARAM, CHENNAI, INDIA – 600 117.
1. INTRODUCTION

The thesis presents research outputs of concurrency control in advanced database transactions that involve Computer Aided Design (CAD) of mechanical components. The CAD database is centrally located and design and development engineers will be located at different places. More than one engineer will be responsible in developing a particular mechanical component for which standard CAD software like Autodesk inventor, CATIA, PROE, Solid works and many more will be used. Completion of the drawing of such component using such software will involve time as the drawing undergoes modifications and approval at various levels of people.

A database system (DBS) is a process that executes read and write operations on data items of a database. A transaction is a program that issues reads and writes to a DBS. When transactions execute concurrently, the interleaved execution of their reads and writes by the DBS can produce undesirable results. Concurrency control is the activity of avoiding such undesirable results. Specifically, the goal of concurrency control is to produce an execution that has the same effect as a serial (noninterleaved) one. Such executions are called serializable. A DBS attains a serializable execution by controlling the order in which reads and writes are executed. When an operation is submitted to the DBS, the DBS can either execute the operation immediately, delay the operation for later processing, or reject the operation. If an operation is rejected, then the transaction that issued the operation is aborted, meaning that all of the transaction’s writes are undone, and transactions that read any of the values produced by those writes are also aborted.

Locks are used for accessing objects. In a database operation lock manager plays an important role whether one or more transactions are reading or writing any part of ‘I’ where ‘I’ is an item. It is the part of that record, for each item ‘I’. Gaining access to ‘I’, is controlled by lock manager
and ensures that there is no, access (read or write) would cause a conflict. The lock manager can store the current locks in a lock table which consists of records with fields (<object>, <lock type>, <transaction>) the meaning of record ('I', 'L', 'T') is that transaction ‘T’ has a lock of type ‘L’ on object ‘I’ [1-2].

The process of managing simultaneous operations on the database without having them interfere with one another is called concurrency [3-4] when two or more users are accessing database simultaneously. Concurrency prevents interference. Interleaving of operations may produce an incorrect result even though two transactions may be correct. Some of the problems that result in concurrency [7-11] are lost update, inconsistent analysis and uncommitted dependency.

There is inability to provide consistency in the database when long transactions are involved. It will not be able to identify if there is any violation of database consistency during the time of commitment. It is not possible to know, if the transaction is with undefined time limit. There is no serializability when many users work on shared objects. During long transactions, optimistic transactions and two phase locking will result in deadlock. Two phase locking forces to lock resources for long time even after they have finished using them. Other transactions that need to access the same resources are blocked. The problem in optimistic mechanism with Time Stamping is that it causes repeated rollback of transactions when the rate of conflicts increases significantly. Artificial neural network [5] with Counter Propagation Network (CPN) has been used to manage the locks allotted to objects and locks are claimed appropriately to be allotted for other objects during subsequent transactions.

Inbuilt library drawing for the dial of fork are available in Autodesk Inventor 9. Drawings considered for this research work are: a) Fork used in a two wheeler front structure, b) Bolted connection and c) Bearing
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong> Lower end, <strong>2.</strong> Height of the end part, <strong>3.</strong> External support, <strong>4.</strong> Height of the external support, <strong>5.</strong> Support for the wedge, <strong>6.</strong> Height of the support for the wedge, <strong>7.</strong> Wedge, <strong>8.</strong> Thickness of the wedge, <strong>9.</strong> Slope of the wedge, <strong>10.</strong> Wedge lock, <strong>11.</strong> Height of the wedge lock, <strong>12.</strong> Concentric hole, <strong>13.</strong> Separator, <strong>14.</strong> Guideway</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1.</strong> Locking slot, <strong>4.</strong> Numbers, <strong>2.</strong> Bolt fixing, <strong>3.</strong> Bolt fixing, <strong>4.</strong> Lower base support width of annular, <strong>5.</strong> Upper base support height, <strong>6.</strong> Height of bolt fixing, <strong>7.</strong> Square upper late, <strong>8.</strong> Inner diameter, <strong>9.</strong> Outer diameter, <strong>10.</strong> Square lower plate, <strong>11.</strong> Concentric 1, <strong>12.</strong> Concentric 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1.</strong> Top diameter, <strong>2.</strong> Gear teeth, <strong>3.</strong> Support shank, <strong>4.</strong> Slotted groove, <strong>5.</strong> Stepped diameter, <strong>6.</strong> Single slot step, <strong>7.</strong> Locking slot.</td>
<td></td>
</tr>
</tbody>
</table>
2. LITERATURE SURVEY

The requirement for concurrency control (CC) arose two decades ago to ensure correctness when a shared database is updated by multiple transactions (Txns) concurrently [12]. Txn concurrency or multiprogramming is required to take advantage of multiple processors and CPU-I/O overlap to attain high Txn throughputs. The universally accepted correctness criterion for processing Txns against a database is serializability; that is, the interleaved execution of a set of concurrent Txns is tantamount to a serial execution [13,14]. This correctness criterion is not acceptable for some applications (e.g., stock trading bids may have a FCFS (first-come, first-served) processing requirement [15]. Standard locking (i.e., strict two-phase locking (2PL) with on-demand lock requests and the general waiting method on lock conflict) is almost exclusively used by current database management systems (DBMS). Strict 2PL requires locks to be released only when the Txn is committed or aborted, since this prevents cascading aborts. Cascading aborts occur when a Txn accesses an object modified by an uncommitted Txn, which aborts after releasing the lock. Standard locking with some deviations is used in most commercial DBMSs.

Research in the area of concurrency control for database systems has led to the development of many concurrency control algorithms. Most of these algorithms are based on one of three basic mechanisms: locking, timestamps and optimistic concurrency control (also called commit-time validation or certification). Survey many of the algorithms that have been developed and describe how new algorithms may be created by combining the three basic mechanisms. Given the ever-growing number of available concurrency control algorithms, considerable research has recently been devoted to evaluating the performance of concurrency control algorithms. The behavior of locking has been investigated using both simulation and analytical models.
An algorithm that uses blocking [16,17] instead of restarts is preferable from a performance viewpoint, but studies by [18], suggest that restarts lead to better performance than blocking. The research that led to the development of the many currently available concurrency control algorithms was guided by the notion of serializability as the correctness criteria for general-purpose concurrency control algorithms. Transactions are typically viewed as sequences of read and write requests and the interleaved sequence of read and write requests for a concurrent execution of transactions is called the execution log. Proving algorithm correctness then amounts to proving that any log that can be generated using a particular concurrency control algorithm is equivalent to some serial log (i.e., one in which all requests from each individual transaction are adjacent in the log).

3. OBJECTIVES

1. To incorporate intelligent techniques in the locking mechanism for minimizing loss of the data of the entities in the CAD file.
2. To increase concurrency in the database.
3. To minimize deadlock leading to hanging of the system.

4. METHODOLOGIES

1. Implementation of Functional Update Back Propagation Neural Network for locking of entities.
2. Implementation of Locally Weighted Projection Regression Network for locking of entities.
3. Implementation of Fuzzy logic for locking of entities.
4.1 FUNCTIONAL UPDATE BACK PROPAGATION NEURAL NETWORK

An Artificial Neural Network is an abstract simulation of a real nervous system that contains a collection of neuron units communicating with each other via axon connections. Such a model bears a strong resemblance to axons and dendrites in a nervous system. Due to this self-organizing and adaptive nature, the model offers potentially a new parallel processing paradigm. This model could be more robust and user friendly than the traditional approaches. ANN can be viewed as computing elements, simulating the structure and function of the biological neural network. These networks are expected to solve the problems in a manner which is different from conventional mapping. Neural networks are used to mimic the operational details of the human brain in a computer. Neural networks are made of artificial neurons which are actually simplified versions of the natural neurons that occur in the human brain. It is hoped that it would be possible to replicate some of the desirable features of the human brain by constructing networks that consist of a large number of neurons. A neural architecture comprises massively parallel adaptive elements with interconnection networks which are structured hierarchically.

Artificial neural networks are computing elements which are based on the structure and function of the biological neurons. These networks have nodes or neurons, which are described by difference or differential equations. The nodes are interconnected layer wise or intra-connected among themselves. Each node in the successive layer receives the inner product of synaptic weights with the outputs of the nodes in the previous layer. The inner product is called the activation value. The activation value is passed through a non-linear function.

When the vectors are binary or bipolar, hard-limiting non-linearity is used. When the vectors are analog a squashed function is used. Some of the squashed functions are sigmoid (0 to 1), tanh (-1 to +1), Gaussian,
logarithmic and exponential. A network with two states of a neuron 0 or 1 and \(-1\) or 1 is called discrete and the same with a continuous output is called analog. In a discrete network at a particular time the state of every neuron is updated, the network is said to be synchronous. If the state of only one neuron is updated, the network is said to be asynchronous. A network is feed forward, if there is no closed chain of dependence among neural states. The same network is feed backward, if there is such a closed chain. When the output of the network depends upon the current input the network is static. If the output of the network depends upon past inputs or outputs, the network is dynamic. If the interconnection among neurons changes with time, the network is adaptive. The synaptic weight updation of the networks can be carried out by supervised methods or by unsupervised methods or by fixed weight association networks methods. In the case of the supervised methods, inputs and outputs are used in the unsupervised methods, only the inputs are used and in the fixed weight association networks methods, inputs and outputs are used along with pre-computed and pre-stored weights.

In the conventional back propagation algorithm (BPA), when the network is trained with analog data, the number of iterations is large for the objective function \((J)\) to reach the desired mean squared error \((\text{MSE})\). The objective function does not reach the desired MSE due to some local minima, whose domains of attractions are as large as that for the global minimum. The network converges to one of those local minima, or the network diverges. The updating of the weights will not stop, unless every input is outside the significant update region \((0.1\) to \(0.9)\), and the outputs of the network will be approaching either 0 or 1. This requires much iteration for the network to converge. To overcome these difficulties, a functional criterion, which results in faster convergence of the network, is used.
The main idea of this method is that the weights of this network are updated only when any one of the nodes in the output layer of the network is misclassified. Even if one of the nodes in the output layer is not misclassified, no updating of weights is done. A node in the output layer is misclassified, if the difference between the desired output and the network is greater than 0.5.

The number of layers and the number of nodes in the hidden layers are decided. The weights among layers are initialized. A training pattern is presented to the input layer of the network, and the difference between the network’s output and the target output is calculated for each node in the output layer. If the difference obtained for each node is greater than the value of a functional criterion, a counter is incremented and the weights are updated. If the difference of not even one node is greater than 0.5, no updating for the weights is done. The MSE of the network for each pattern is calculated only when at least one node in the output of the network is misclassified. Remaining training patterns are presented to the network. Training of the network is stopped when a performance index of the network is reached.

4.2. **LOCALLY WEIGHTED PROJECTION REGRESSION NETWORK**

Locally Weighted Projection Regression (LWPR) [6] is an algorithm that achieves nonlinear function approximation in high dimensional spaces even in the presence of redundant and irrelevant input dimensions. At its core, it uses locally linear models, spanned by a small number of univariate regressions in selected directions in input space. The nonparametric local learning system

i) Learns information rapidly with second order learning methods based on incremental training.
ii) Uses statistically sound stochastic cross validation to learn information.

iii) Adjusts its weighting kernels based on local information only.

iv) Has a computational complexity that is linear in the number of inputs, and

v) Can deal with a large number of possibly redundant and irrelevant inputs.

4.3 FUZZY LOGIC

Fuzzy Logic is a multi valued logic that allows intermediate values to be defined between conventional evaluations like true/false, yes/no, high/low. Notions like rather tall or very fast can be formulated mathematically and processed by computers, in order to apply a more human-like way of thinking in the programming of computers. Fuzzy systems are an alternative to traditional notions of set membership and logic that. In this research work, membership functions have been created that suits locking and transactions.

5. RESULTS

Formulation of concurrency control for dial of fork and training the proposed algorithms are presented. Let us assume that there are two users editing the dial of the fork. User1 edits O₁ and hence O₂ will be locked sequentially (Table I). Immediately user2 wants to edit O₂, however he will not get transaction as already O₂ is locked.

<table>
<thead>
<tr>
<th>Group</th>
<th>First feature</th>
<th>Remaining feature to be locked</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>G2</td>
<td>10</td>
<td>11,12,13,14</td>
</tr>
<tr>
<td>G3</td>
<td>5</td>
<td>6,7,8</td>
</tr>
<tr>
<td>G4</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
However, user2 or any other user can try to access O₃ to O₁₄. The variables used for training the ANN about locks assigned to different objects are transaction id, object id, lock mode (Table 2). Transaction id represents the client or any other intermediate transactions. Object id represents the entire feature or an entity in the file. Mode represents type of lock assigned to an object. In Table 2, column 1 represents the lock type. column 2 represents the value to be used in the input layer of the ANN. Column 3 gives binary representation of Lock type to be used in the output layer of ANN. The values are used as target outputs in the module during lock release on a data item.

<table>
<thead>
<tr>
<th>Lock type</th>
<th>(Input layer representation numerical value).</th>
<th>Binary representation in target layer of the LWPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Not locked</td>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>001</td>
</tr>
<tr>
<td>X</td>
<td>2</td>
<td>010</td>
</tr>
<tr>
<td>IS</td>
<td>3</td>
<td>011</td>
</tr>
<tr>
<td>IX</td>
<td>4</td>
<td>100</td>
</tr>
</tbody>
</table>

Initially, user 1 and user 2 have opened the same dial of fork file from the common database. The following steps shows sequence of execution and results. T₁ edits O₁ with write mode. Table 3 shows pattern formed for the training.

<table>
<thead>
<tr>
<th>Object number</th>
<th>Input pattern</th>
<th>Target output pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>O₁</td>
<td>[1 1]</td>
<td>[0 1 0]</td>
</tr>
</tbody>
</table>
Step 1: The transaction manager locks objects mentioned in the third column of Table 1. Repeat step 1 with the patterns given in Table 4.

<table>
<thead>
<tr>
<th>Object number</th>
<th>Input pattern</th>
<th>Target output pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>O₁</td>
<td>[1 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₂</td>
<td>[2 1]</td>
<td>[0 1 0]</td>
</tr>
</tbody>
</table>

Step 2: A new transaction T₂ access O₂. A pattern is formed to verify if lock has been assigned to O₂ and its associated objects O₁. Only when the locks are not assigned to O₂ and O₁ then T₂ is allowed. The following input patterns are presented to the testing module to find if the output [0 0 0] is obtained in the output layer. During testing, the final weights obtained during training will be used. Otherwise it means that lock has been assigned to either O₂. In such case, transaction is denied for T₂. Else the following Table 5 is presented in step 1.

<table>
<thead>
<tr>
<th>Object number</th>
<th>Input pattern</th>
<th>Target output pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>O₁</td>
<td>[1 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₂</td>
<td>[2 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₃</td>
<td>[3 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₄</td>
<td>[4 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₅</td>
<td>[5 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₆</td>
<td>[6 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₇</td>
<td>[7 1]</td>
<td>[0 1 0]</td>
</tr>
<tr>
<td>O₈</td>
<td>[8 1]</td>
<td>[0 1 0]</td>
</tr>
</tbody>
</table>
**Step 3:** To know the type of lock value assigned to an object and for a transaction, testing is used. Testing uses the final weights created by training. The proposed ANN for lock state learning and lock state finding has been implemented using Matlab 10.

The performance of the algorithms proposed has been presented on the following criteria.

1. Locking time for each object.
2. Releasing time for each object.
3. Total Locking time for each transaction group.
4. Arrival rate.
5. Response time.

**6. CONCLUSION**

An artificial neural network with FUBPA, LWPR and Fuzzy logic have been implemented for providing concurrency control to maintain consistency in the CAD database. A dial of fork used in a two wheeler, Bolted connection and Bearing have been considered. The FUBPA, LWPR and Fuzzy logic method requires less memory based on the topology used for storing objects and its transactions when compared with conventional method.

**THESIS LAYOUT**

1. Introduction and scope of the work
2. Experimental Data
3. Concurrency control using Functional update BPA
4. Concurrency control using LWPR
5. Concurrency control using Fuzzy Logic
6. Comparison of the results
7. Conclusion and scope of the work
LIST OF PUBLICATIONS


INTERNATIONAL CONFERENCE


NATIONAL CONFERENCE

REFERENCES


