ÝSTANBUL ÜNÝVERSÝTESÝ MÜHENDÝSLÝK FAKÜLTESÝ
ELEKTRÝK-ELEKTRONÝK DERGÝSÝ

YIL : 2001-2002 (84 – 110)
CÝLT : 1
SAYI : 1

Random Neural Network Approach in Distributed Database Management Systems

Paylaþtýrýlmýþ Veri Tabanlý Sistemlerde Rasgele Hücresel Að Yaklaþýmý

Adem Karahoca (*)
(*) Istanbul University
Engineering Faculty
Computer Engineering
Department
Avcýlar 34850 Istanbul - Turkey
E-mail: kadem@istanbul.edu.tr

Osman N. Ucan(**)
(**) Istanbul University
Engineering Faculty
Electrical & Electronics Engineering
Department
Avcýlar 34850 Istanbul – Turkey
E-mail: uosman@istanbul.edu.tr

Erkan Danacý (***)
(***) The Scientific and Technical
Research Council of Turkey
(TUBITAK), Kocaeli
And Selcuk University Electrical &
Electronic Engineering Department,
Konya-Turkey
E-mail: danaci@btae.mam.gov.tr

Abstract

In this paper, Random Neural Network (RNN) approach has been applied to the distributed database design of technology-corridor prototype project for Avcýlar Campus of Istanbul University in Turkey. This project includes university, industry and government collaboration. Here, we need a distributed environment for designing sub databases and fragmenting them on the sites. Therefore, different techniques are considered for a database fragmentation. When techniques are described, eight different properties are controlled for database process behaviors. Fragmentation techniques are ordered for each property. These orders help us to make decision about which fragmentation technique is the best for distributed database system. Here RNN approach and Radial basis functions networks are used for generalization of selection of partitioning techniques. Training data of Radial basis function networks and RNN are provided from the programs, which are executing under Oracle database. In this paper, firstly we used Neural Networks approaches at distributed environments for automatic database fragmentation selection operation and designed two non-linear algorithms. Then, Random Neural Network Methods have been applied to the same problem and obtained satisfactory results.

Key Words: Database, database design, distributed database, database fragmentation, neural networks, radial basis function networks, random neural network

I. INTRODUCTION

University and industry collaboration is required for combination of the theoretical and implicational information. Therefore we need a distributed database environment that integrates theoretic and practical values.

Distributed database systems generally include more than two geographic remote sites. Interrelated sites have partially its hardware and software which includes database management systems and applications [1].

Distributed database design process performs global conceptual design and then local conceptual design

that fragments databases on the sites [2]. Distribution design includes requirements analysis, view analysis and integration and these inputs are oriented towards to the distributed database design [3]. Database fragmentation, replication and allocation steps are occurred in distribution design decisions. Fragmentation and allocation issues simultaneously are considered and integrated in the work [4].

In our paper, we dealed with database fragmentation step which is very important and thus we only considered to suggest generalization of selection on partitioning techniques by non-linear algorithms of Radial Basis Function Neural Networks [5] and Random Neural Network(RNN) that automatically detect the best fragmentation alternatives.

There are many database fragmentation techniques developed for organizing data physically in storage devices. Every technique firstly divides the data into groups then assigns those groups to physical pages which can be divided into six categories: Horizontal fragmentation, group horizontal fragmentation, single vertical fragmentation, physical vertical fragmentation, group vertical fragmentation and mix fragmentation.

Horizontal fragmentation is used for enabling a relation into same attributes with different tuples. In [6] horizontal fragmentation, data partitioning is used for database design objective. In this paper we firstly present an algorithm that depends on a simple knowledge base system and this algorithm uses relations to divide group horizontal fragments. Fragmenting relations horizontally using knowledge-based systems are considered in [7]. Formal approaches for horizontal fragmentation explained in [8].

Single vertical fragments are composed of a column from relation and key attributes or a tuple id. Vertical fragmentation algorithms are presented in [9] and [10]. Also, a formal approach with vertical fragmentation is presented in [11] for distributed database design. In distributed databases vertical fragmentation is very vital for database design and analyzing. Therefore, a work is related with this problem in [12] to determine an objective function.

Physical vertical fragmentation method is sub fragments of constant size of physical groups. This method does not appear in any database management system [13]. Group vertical fragmentation depends on attribute affinity matrix, which is used firstly in Bond Energy Algorithm by Mc.Crormick and et'al.[2]. Mix fragmentation method is considered for composition of both horizontal and vertical fragmentation methods' additions [1].

Random Neural Network (RNN) is a simple form of homogeneous neural network whose characteristics are expressed in terms of probabilistic assumptions. The networks considered operate in an asynchronous manner and receive the influence of the environment in the form of external stimulations. The operation of the network is described by means of a Markovian process whose steady-state solutions yields several global measures of the network's activity [14].

The paper organization is as follows: Section II defines neural networks and its special branch random neural network. Section III is detailing radial basis function neural networks and neural network approaches to the database fragmentation problem. Section IV describes training of neural network model and Section V includes comparison between RNN and other approaches. Section VI, focuses on the result sets which obtained from neural nets. Section VII includes conclusions about our work.

II. RANDOM NEURAL NETWORK APPROACH

Artificial neural networks have been used for learning and therefore generalize by massively distributed structures. Neural networks solve complex problems by training sets. Neural networks includes following important topics that are useful for complex problem solving such as nonlinearity, input-output mapping, adaptivity, evidential response, and contextual information.

Random Neural Network (RNN) model have defined in 1989 and extended and generalized in 1990 by [14]. It has very interesting features. It seems to be closer to real biophysical neural network, since signals of the scheme are as voltage spikes rather than fixed levels as in previous classical network structures. It is more easly computed where each neuron is simply represented by a counter. Thus hardware implimentation is practical. It carries more information on system states since each neuron potential and level of excitation are chosen as an integer instead of a binary variable. If the system is stable, it is computationally efficient. In RNN model, there are positive and negative signals (Figure 1). These signals travel among the neurons in the form of spikes of unit amplitude. Positive signals represent excitation and negative signals represent inhibition. These signals can be transmitted either from other neurons or from outside world and then they are summed at the input of

each neuron and produce its signal potential. Each neuron's state is a non-negative integer number called its potential, which increases when an excitation signal arrives to it, and decreases when an inhibition signal comes. An excitatory spike is evaluated as a ``+1" signal at a receiving neuron, while an inhibitory spike is interpreted as a ``-1" signal. If neuron potential is positive, it fires and sends out signals to the other neurons of the network or outside world. In the case of firing, neural potential decreases. A neuron emitting a spike, whether it is an excitation or an inhibition, will lose potential of one unit, which results as going from some state to previous state . The state of the n-neuron network at time t, is represented by the vector of non-negative integers $k(t) = (k_1(t),..., k_n(t))$, where $k_i(t)$ is the potential or integer state of neuron i. Arbitrary values of the state vector and of the i-th neuron's state are shown by k and k_i.

Let assume, potential of neuron i is positive and fires. It is then excited and sent out spikes. The spikes are sent out with independent, identically and exponentially distributed inter-spike intervals at a rate of r(i). These spikes can reach any neuron j with probability $p^+(i,j)$ as excitatory signals, or with probability $p^-(i,j)$ as inhibitory signals. A neuron may also send signals out of the network with probability d(i). The probabilitiy ratios of $p^+(i,j)$, $p^-(i,j)$ and d(i) are as,

$$d(i) + \sum \left[p^+(i, j) + p^-(i, j) \right] = 1. \qquad (1)$$

We can also give wight values ω^+ and ω^- for neurons as a multiplication of spike rate r(i) with probability of being excitatory and inhibitary respectively.

$$w^+_{ij} = r(i)\, p^+(i, j), \quad w^-_{ij} = r(i)\, p^-(i, j) \qquad (2)$$

These ωs are similar to synaptic weights in classical connectionist models. The signals that arrive to the considered neuron from outside world can reach at rates $\Lambda(i)$ and $\lambda(i)$ in the case of their being excitary and inhibitary signals respectively. So we can say that RNN is a "recurrent network" model with feedback loops.

The signal flow equations which yield the rate of signal arrival and hence the rate of firing of each neuron in steady-state are non-linear. Computations related to this model are based on the probability distribution of network state $p(k,t) = \Pr[k(t) = k]$, or with the marginal probability that neuron i is excited $q_i(t) = \Pr[k_i(t) > 0]$. The time-dependent behavior of the model is described by an infinite system of Chapman-Kolmogorov equations for discrete state-space continuous Markovian systems [15].

In RNN model, the frequency of the travelling of the spikes carry the information. Let assume that neuron j, has positive potential and sends spikes to neuron i at a frequency $\omega_{ij} = \omega_{ij}^+ + \omega_{ij}^-$. Here as it is expressed before, ω_{ij}^+ is the multiplication of probability of the excitatory signal $p^+(i,j)$ and data rate r(i). In the same manner ω_{ij}^- is the multiplication of probability of the inhibitary signal $p^{+-}(i,j)$ and data rate r(i). These spikes will be emitted at exponentially distributed random intervals. Each neuron acts like a non-linear frequency modulator by forming an amplitude quantity, namely $q_i(t)$ related with the incoming ω_{ij}. In this model, neuron i sends out excitatory and inhibitory spikes at rates (or frequencies) $q_i(t) r(i) p^+(i, j)$, $q_i(t) r(i) p^-(i, j)$ to any neuron j.

As $t \to \infty$, the stationary probability and quantity can be expressed as,

$$p(k) = \lim_{t\to\infty} p(k,t),\ q_i = \lim_{t\to\infty} q_i(t),\ i = 1,2,..,n \qquad (3)$$

Where q_i denote the quantity and defined as,

$$q_i = l^+(i) / [r(i) + l^-(i)] \qquad (4)$$

Here $\lambda^+(i)$ and $\lambda^-(i)$ are defined as,

$$\begin{aligned} l^+(i) &= \sum_j q_j r(j) p^+(i, j) + \Lambda(i), \\ l^-(i) &= \sum_j q_j r(j) p^-(i, j) + l(i) \end{aligned} \qquad (5)$$

Here q_j is the j^{th} quantity , r(i) is data rate, $p^+(i,j)$ and $p^-(i,j)$ are the probability values. Let k(t) be a vector of neuron potentials at time t and $k=(k_1, ..., k_n)$ be a particular value of the vector, then Equation (3) can be rewritten as,

$$p(k) = \lim_{t\to\infty} \Pr[k(t) = k]$$

If a nonnegative solution $\{ l^+(i), l^-(i) \}$ exists for Equations (4) and (5) such that each $q_i<1$, then

p(k) is expressed as,

$$p(k) = \prod_{i=1}^{n} [1 - q_i] q_i^{ki} \qquad (6)$$

Then the quantities, q_i which are most useful for computational purposes are directly obtained from:

$$\lim_{t \to \infty} \Pr[k_i(t) > 0] = q_i = \lambda^+(i) / [r(i) + \lambda^-(i)]$$

if $q_i < 1$.

We can now define N(i) and D(i) as follows,

$$N(i) = \sum_j q_j \omega^+(i, j) + \Lambda(i),$$

$$D(i) = r(i) + \sum_j q_j \omega^-(i, j) + \lambda(i)$$

where,

$$\omega^+(i, j) = r(i) p^+(i, j) \geq 0, \quad \omega^-(i, j) = r(i) p^-(i, j) \geq 0$$

$$r(i) = \sum_j \left[\omega^-(i, j) + \omega^+(i, j) \right]$$

Then Equation (4) can be rewritten as,

$$q_i = \frac{N(i)}{D(i)} \qquad (7)$$

In the learning algorithm that will be told in the following Section 2.1, we will use Equation (7) to minimize cost function.

2.1. The Learning Algorithm

In the learning algorithm, Random Neural Network chooses the set of network parameters (ω^+, ω^-) in order to learn a given set of K input-output pairs (i , Y). The set of successive inputs are denoted , $i = \{i_1, ..., i_k\}$ and $i_k = (\Lambda_k, \lambda_k)$ [15]. These are pairs of positive and negative signal flow rates entering each neuron and can be written as,

$$\Lambda_k = [\Lambda_k(1), ..., \Lambda_k(n)], \qquad l_k = [l_k(1), ..., l_k(n)]$$

The successive desired outputs are $\mathbf{Y}=\{y_1, \ldots\ldots y_k\}$, where each y_k is composed of $\{y_{1k}, \ldots..y_{nk}\}$ whose elements take values in the range of [0,1]. The network approximates the set of desired output vectors in a manner that minimizes a cost function,

$$E_k = \frac{1}{2} \sum_{i=1}^{n} \alpha_i (q_i - y_{ik})^2 \qquad \alpha_i \geq 0 \qquad (8)$$

where q_i is defined in Equation (7). If we wish to remove some neuron j from network output, and hence from the error function, it suffices to set $\alpha_j = 0$. Both of the n by n weight matrices $\mathbf{W}_k^+ = \{w_k^+(i, j)\}$ and $\mathbf{W}_k^- = \{w_k^-(i, j)\}$ have to be learned after each input is presented, by computing for each input , a new value $\mathbf{W}_k^+$ and $\mathbf{W}_k^-$ of the weight matrices, using gradient descent. We try to find only solutions for which all these weights are positive. Let

$$w(u, w) \equiv w^-(u, v) \text{ or } w(u, w) \equiv w^+(u, v).$$

The weights can be updated as ,

$$w_k(u, w) = w_{k-1}(u, v) - h \sum_{i=1}^{n} a_i (q_{ik} - y_{ik}) \left[\frac{\partial q i}{\partial w(u, v)} \right]_k \qquad (9)$$

where $\eta > 0$ is some constant, and

1. q_{ik} is calculated using the input i_k and $w(u, v) = w_{k-1}(u,v)$, as given in Equation (3).
2. $[\partial q_i / \partial w(u,v)]_k$ is evaluated at the values $q_i = q_{ik}$ and $w(u,v)=w_{k-1}(u,v)$

To compute $[\partial q_i / \partial w(u,v)]_k$ we turn to the expression 3, form which we derive the following equation:

$$\partial q_i / \partial w(u, v) = \sum_j \partial q_j / \partial w(u, v) \left[w^+(j, i) - w^-(j, i) q_i \right] / D(i) + a \qquad (10)$$

where,

$$a = \begin{cases} -q_i / D(i), & \text{if } u = i \\ +q_u / D(i), & \text{if } w(u,v) = w^+(u,i) \\ +q_u q_i / D(i), & \text{if } w(u,v) = w^-(u,i) \end{cases}$$

Let $q=(q_1, \ldots\ldots., q_n)$, and define the n x n matrix

$$W = \{[w^+(i,j) - w^-(i,j) q_j] / D(j)\} \quad i.,j=1, \ldots\ldots, n \qquad (11)$$

We can now write the vector equations:

$$\partial q / \partial w^+(u, v) = \partial q / \partial w^+(u, v) W + \mu^+(u, v) q_u$$
$$\partial q / \partial w^-(u, v) = \partial q / \partial w^-(u, v) W + \mu^-(u, v) q_u \qquad (12)$$

Where the elements of the n-vectors

$$m^{+}(u,v) = \left[m_1^{+}(u,v),.....,m_n^{+}(u,v)\right] \quad (13)$$
$$m-(u,v) = \left[m_1^{-}(u,v),.....,m_n^{-}(u,v)\right]$$

$$\mu_i^{+}(u,v) = \begin{cases} -1/D(i), & \text{if } u = i, v \neq i \\ +1/D(i), & \text{if } u \neq i, v = i \\ 0, & \text{for all other values of (u,v)} \end{cases} \quad (14)$$

$$m_i^{-}(u,v) = \begin{cases} -(1+q_i)/D(i), & \text{if } u = 1, v = i \\ -1/D(i), & \text{if } u = i, v \neq i \\ -q_i/D(i), & \text{if } u \neq i, v = i \\ 0, & \text{for all other values of } (u,v) \end{cases} \quad (15)$$

Notice that

$$\partial q/\partial w^{+}(u,v) = \mu^{+}(u,v)q_u[I-W]^{-1}$$
$$\partial q/\partial w^{-}(u,v) = \mu^{-}(u,v)q_u[I-W]^{-1} \quad (16)$$

Where I denotes the n x n identity matrix. Thus the main computational work is to obtain $[I-W]^{-1}$. This is of time complexity $O(n^3)$, or $O(mn^2)$ if an m-step relaxation method is used.

III. RADIAL BASIS FUNCTION NETWORKS AND NEURAL NETWORK APPROACHES TO THE DATABASE FRAGMENTATION PROBLEM

Radial-basis functions were first introduced in the solution of the real multivariate interpolation problem by Powell (1985). Broomhead and Lowe (1988) were he first to exploit the use of radial-basis functions in the design of neural networks.

The construction of radial basis function (RBF) network in its most basic form involves three entirely different layers. The input layer made up of source nodes. The second layer is a hidden layer whose dimension is high enough. This layer is used for a different aim than multilevel neural networks. The output layer creates the answer of the network from the activation vectors in the input layer. Transformation from the input space to the hidden unit space is non-linear and it is linear from the hidden layer space to the output unit space.

If the centers of the hidden unit space are especially chosen to be adaptive it is possible to reduce the dimension of the hidden layer space [16]. We will see how to place our test results in the RTF network in Figure 2, and discuss the training input vectors in the next sub section. You will find detail algorithms for distributed design, which includes database fragmentation methods in algorithm 1 and 2.

3.1. Input and Output Parameters

We have taken the work made by Gruenwald and Eich in 1993 as an example and accepted the heap of parameters below as input to our Neural Network Model [13].

1) Number of the physical pages necessary for storing relations,

2) Cost of reorganization. Cost of reorganizing a relation (deleting from memory and reloading),

3) Cost of deleting a column from a schema,

4) Cost of adding a column to a schema,

5) Cost of accessing columns. This is actually the cost of removing the columns in all records from the given relation,

6) Cost of reselecting a record. This is actually the cost of selecting a record from the given relation,

7) Cost of adding a new record to a relation,

8) Cost of modifying a record. This is the cost of modifying some columns of a given record.

Basic values we want to obtain as a result using the above basic inputs are the basic values that will directly effect the performance when we distribute relations like the results of the projection processes on relations and selection processes.

Because of that, our vector in the output layer should include the basic values in Table 1:

Input vectors are representing as a,

$\tilde{x} = [\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_8]^T$, also output vector

that is called as y representing with

$\tilde{y} = [f_1, f_2, f_3, f_4, f_5, f_6, f_7]$

3.1.1. Radial Basis Function Networks Algorithms:

Algorithm 1-Static training model

1. $\tilde{x} = [\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_8]^T$ Input vectors calculated by developed programs.
2. $\tilde{y} = [f_1, f_2, f_3, f_4, f_5, f_6, f_7]$ Output vectors calculated by developed programs.
3. $\tilde{\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$ Constant vector is reading from a file.
4. All X input vectors selected as a center vector which is called C.
5. $\tilde{y} = \tilde{\lambda} e^{-\frac{\|\tilde{\alpha}(\tilde{x} - c_i)\|}{2\sigma^2}}$ Equation is used for calculating λ learning coefficients for each fragmentation techniques.
6. $\frac{1}{N+1} \sum_{t=1}^{N+1} \varepsilon_t^2$ Stopping criteria is controlled.

If the iteration is not ended, new X vector is added iteration [5].

Algorithm 1 calculates learning coefficients for every fragmentation method for cost analyzing. After learning process, for every input set is used for calculating output sets by learning coefficients. Each input sets represents different fragmentation technique. Input parameters are calculating by Oracle programs and output parameters too. But, output parameters are calculated only once. Algorithm 1 is called as a static training model, which includes constant Alfa matrix.

3.1.2. Radial Basis Function Networks Algorithms:

Algorithm 2-Dynamic training model

X input vectors and Y output vectors are calculated by developed programs.[5]

$$\begin{bmatrix} y_1^1 \\ . \\ . \\ . \\ y_1^N \end{bmatrix} = \begin{bmatrix} 1 & \|X^1 - c_1\| & . & . & \|X^1 - c_N\| \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & \|X^N - c_1\| & . & . & \|X^N - c_N\| \end{bmatrix} \begin{bmatrix} w_0 \\ . \\ . \\ . \\ w_N \end{bmatrix} + \begin{bmatrix} e_1 \\ . \\ . \\ . \\ e_N \end{bmatrix} \quad (16)$$

$$\|X_i - c_j\| = \exp(-(X_i - c_j)^T \, stdev(inv((X_i - c_j).(X_i - c_j)^T)).(X_i - c_j)) \quad (17)$$

$\frac{1}{N+1} \sum_{t=1}^{N+1} \varepsilon_t^2$ Stopping criteria is controlled.

If the iteration is not ended, new X vector is added iteration as follow;

$$\begin{bmatrix} x_1^1 \\ . \\ x_8^1 \\ 0 \\ . \\ 0 \end{bmatrix} ..... \begin{bmatrix} x_1^N \\ . \\ x_8^N \\ \varepsilon_{N-1}^8 \\ . \\ \varepsilon_{N-5}^8 \end{bmatrix}$$ and then go to the equation (10).

In the first algorithm is not used a error correction mechanism. Therefore, algorithm1 heaped errors. Algorithm 2 is evaluated by error correction mechanism. When the model accessed to the fourth step, ε errors heaped and then according to the stopping criteria process is controlled. Algorithm 1 is used Euclidean distance as well as algorithm 2. But, algorithm 2 includes a simple value for eliminating 1/0 elements in Euclidean distance. According to Alfa constant, algorithm 2 automatically detects which output parameters depends on input parameters. Algorithm 2 uses firstly evaluating ω vectors for each fragmentation techniques.

3.1.3. Random Neural Network Algorithm

We now have the information to specify the complete learning algorithm for the network as explained in

Section 2.1. We first initialize the matrices W_0^+ and W_0^- in some appropriate manner. This initation will be made at random. Choose a value of η, and then for each successive value of k, starting with k=1 proceed as follows:

1. Set the input values to $i_k = (\Lambda_k, \lambda_k)$.
2. Solve the system of nonlinear Equations (3) with these values.
3. Solve the system of linear Equations (16) with the results of (2).
4. Using Equation (9) and the results of (2) and (3), update the matrices W_k^+ and W_k^-. Since we seek the best matrices (in terms of gradient descent of the quadratic cost function) that satisfy the nonnegativity constraint, in any step k of the algorithm, if the iteration yields a negative value of a term, we have two alternatives:
   a) Set the term to zero, and stop the iteration for this term in this term in this step k; in the next stop k+1 we will iterate on this term with the same rule starting from its current null value;
   b) Go back to the previous value of the term and iterate with a smaller value of η.

3.2 Defining First Input Parameter, the number of Physical Pages

Heap of parameters in Table 2 are used to calculate the value of the physical pages necessary for storing fragmented databases, and the Oracle program in Figure 3 implements the necessary calculations using these parameters and stores in the database.

User can define each of the above parameters according to their limits via the program in Figure 3 and clicking the buttons created for each fragmentation technique can make calculations. After these calculations, the obtained number of physical pages necessary for each fragmentation technique are stored in the database these values are then entered in our neural network as training parameters when organizing x vector.[5]

3.3 Defining Reorganization Cost and The Other Parameters

We created a dynamic SQL Wizard program for calculating the cost necessary for reorganizing a relation, deleting it from the memory and reloading it. With this Oracle program, the user can relate the tables without any knowledge of SQL and in the next step all values are measured respectively for the remaining 7 parameters and then stored in the database (Figure 4).

We can define the join relation between the columns of table(s) of which we find the fragmentation parameters dynamically by running the above SQL statement on Oracle database.

SQL Generator button of the program we use for obtaining the input data of the neural network's training heap is used for obtaining SQL statement after specifying the table and columns and storing them in FRAGMENT table as in Figure 5 and Figure 6.

Costs for reorganizing a relation, deleting columns from a schema, adding columns to a schema, accessing columns, recreation of a record, adding a record and updating a record are obtained by storing the above SQL statement, i.e. storing in the database the values of these costs in terms of time. We obtain the data of the training heap with these operations.

IV. ENTERING TRAINING DATA TO THE NEURAL NETWORK AND CALCULATION OF WEIGHT CONSTANTS

While a distributed database environment made up, according to our model firstly we select input and output values and then neural network model is being trained. When w, weight constants vector is calculated, our database fragmentation methods general cost value is optimized. Therefore, our proposed neural network models are platform independent approaches.

Different from the linear approaches chosen for fragmentation techniques, the first non-linear approach in the literature is being implemented with this work. In this method, because the calculation of the weight constants obtained is sensitive and the environment for comparison with the other fragmentation techniques is provided, the base structure necessary for fragmentation and distribution of the database on the distributed database system is independent from the platform, and most important, it is independent from the database schemas.

Neural network is being run for each fragmentation technique and so the data sets obtained can be used for comparing fragmentation techniques. Another advantage of this model is that, weight constants and vectors will be obtained for each fragmentation

technique after finding the parameters of fragmentation techniques and output values for the design of a specific number of database schemas. We will reach the base of obtaining the outputs directly from only input parameters and the weights specified before during the process of fragmenting any database schema after obtaining system constants. This is the principal feature of neural networks.

If we are testing our system very sensitively and providing input and output values, the neural networks algorithms will give us very efficient total cost ranks. We will see in the next sections that radial basis function networks algorithms calculated very fastly, but generally ranks overlapped in the single and horizontal approaches. The random neural network's and RBFNN Algorithm 2's training time are long than radial basis function neural network algorithm 1. But, it gives us non-overlapped outputs for the fragmentation selection decisions. In the next section we will focus on this comparisons.

V. PERFORMANCE COMPARISON OF NEURAL NETWORK MODELS

Linear approach resolutions couldn't obtain real solves as accepted according to our problem. In Figure 33, we have been shown that Horizontal Fragmentation method is suitable for every approach. But, when we change parameters, we have seen that linear approach couldn't make a decision about which method is the best. Therefore, firstly we used Radial basis function neural networks for training a net and then changing parameters to trace differentials. In this approach, we suggest two sub method: static and dynamic algorithms.

In Static Radial Basis Functions NN approach accepts that particular input values related with only particular outputs. Therefore, process time is decreased. But also, was suggested dynamic algorithm for define a different characteristics included in. Maybe process time was increased, but we could control that our α matrix is realized. Thus, both two algorithms are selectable. Unfortunately, there are problems in this method; in column access parameters are the same for single vertical fragmentation method with physical fragmentation method. Depends on the problems, general cost function computations in these methods may have been chosen horizontal or single vertical fragmentation methods. For eliminating this overloaded problem, we suggest RNN approach and RBFNN Algorithm 2.

In Random Neural Network method same strategy is followed as in radial neural networks. Firstly we train net, then testing our inputs by weight constants and exploring results. In this method, we can eliminate overloaded output values and therefore without computing general cost function, we make decision that which fragmentation is the best.

If we use classical approach as presented as in [13], the best of the fragmentation techniques in terms of "column deletion (Figure 9)" and "column addition (Figure 10)" are the Physical vertical and the Single Vertical. The best of the fragmentation techniques in terms of column access are the Physical Vertical and the Single Vertical (Figure 11). In Figure 7, shows that optimum "the number of pages" in fragmentation is the Single Vertical and Figure 8 shows that the best of the fragmentation techniques in terms of "reorganization" is the Single Vertical. In Figure 12. The best of the fragmentation techniques in terms of "record recreation" is the Horizontal. Figure 13. The best of the fragmentation techniques in terms of "record addition" is the Horizontal. Figure 14. The best of the fragmentation techniques in terms of "column modification" is the Horizontal. Figure 15. General cost function for linear approach. In this approach, the total cost generally optimized in the single or horizontal fragmentation method as seen in Figure 15.

Radial Basis Function Networks algorithms as used in [5], the best of the fragmentation techniques in terms of "column deletion (Figure 18)" and "column addition (Figure 19)" are the Physical vertical and the Single Vertical. The best of the fragmentation techniques in terms of column access are the Physical Vertical and the Single Vertical (Figure 20). In this approach, couldn't determine exactly which fragmentation method is best in three statuses. But, in total cost analysis, is determine that horizontal fragmentation method is suitable for our distributed environment.

Figure 16-The best of the fragmentation techniques in terms of "the no. of pages" is the Single Vertical. Figure 17-The best of the fragmentation techniques in terms of "reorganization" is the Single Vertical. Figure 18-The best of the fragmentation techniques in terms of "column deletion" are the Physical vertical and the Single Vertical. Figure 19-The best of the fragmentation techniques in terms of "column deletion" are the Physical Vertical and the Single Vertical. Figure 20-The best of the fragmentation techniques in terms of column access are the Physical Vertical and the Single Vertical. Figure 21-The best of the fragmentation techniques in terms of "record recreation" is the

Horizontal. Figure 22-The best of the fragmentation techniques in terms of "record addition" is the Horizontal. Figure 23-The best of the fragmentation techniques in terms of "column modification" is the Horizontal. Figure 24-General cost function for radial basis functions neural networks.

Thus we were considering random neural network model which is represented in Figure 1. In this model, gave us the best solutions that every input parameter represented a single fragmentation method. In total cost, horizontal fragmentation method is the best choice for our environment. Figure 25-The best of the fragmentation techniques in terms of "the no. of pages" is the Single Vertical. In Figure 26-The best of the fragmentation techniques in terms of "reorganization" is the Single Vertical. Figure 27-The best of the fragmentation techniques in terms of "column deletion" is the Single Vertical. Figure 28-The best of the fragmentation techniques in terms of "column deletion" is the Single Vertical. Figure 29-The best of the fragmentation techniques in terms of column access is the Single Vertical. Figure 30-The best of the fragmentation techniques in terms of "record recreation" is the Horizontal. Figure 31-The best of the fragmentation techniques in terms of "record addition" is the Horizontal. Figure 32-The best of the fragmentation techniques in terms of "column modification" is the Horizontal. Figure 33-General cost function for random neural network. Figure 34-General cost comparsion within three approaches.

VI. DISCUSSION OF EXPERIMENTAL RESULTS OBTAINED BY NEURAL NETWORK MODELS

6.1 Classical and RBFNN Algorithm 1 Approach for Ordering of Fragmentation Techniques

In this approaches, the first feature is based on only the total number of pages in a relation. Here the only single horizontal has the maximum degree. However, there are great differences between the results based on variables like record size, column size group size and column dependence. A database designer who knows certain relation and column sizes may make clearer suggestions using formulas told in that part for database size. The second feature adds re-grouping cost to this I/O cost. This changes the virtual order of four techniques in the middle. Considering the worst cost for re-grouping makes this order. If no re-grouping is made the orders shown in Table 3-4 will remain unchanged. The third group of the three features includes access to one or more columns for all records in a relation. In those cases, vertical technique is the best one. Single vertical is always the best since it requires the minimum number of pages for storing a column. However, when the number of columns accessed approaches the number of features in a relation, orders in the first feature gain straightness. The last three features define the expected performance of typical selecting, adding, modifying and deleting operations. Because they require full examination of a record, the vertical approach is the best one here. We see results that conflict with the each other. A better performance may be obtained when single vertical design is used on a system that runs projection processes on one or two columns.

Systems that implement re-obtaining or updating operations should use the horizontal technique. Which technique is the best for your environment? Using the appearing frequency of every database operation and giving regular weight to every feature may then obtain a general order.

Order of fragmentation methods by cost is given in Table 3-4. The method with the highest value is the one with the lowest cost.

6.2 RBFNN Algorithm 2 and RNN Approach for Ordering of Fragmentation Techniques

As you will see in Table 5, column deletion, addition and access costs have got a small difference between single vertical and physical vertical fragmentation techniques. But in Table 6, every first rank various from others with a great difference. Table 5 and 6 shows the reason of why we are focusing on RNN.

VII. CONCLUSION

The first feature to consider in design of a distributed database system is the fragmentation of the central databases into sites with the minimum cost. Research operations and methods that we try to explain above provide comparison of fragmentation techniques in Oracle environment even without knowing SQL. By trainable neural network model, it is possible to calculate results with the help of weight constants instead of applying fragmentation tests for all tables and relations of the system. Due to this, determination of a fragmentation technique becomes very simple after training the neural network. The research we start is establishing the schema in the central database schema and automatic determining automatically the

fragmentation technique by an intelligent system under the highlight of the criterions specified for sites.

REFERENCES

1. Ceri, S., Pelagatti, G. (1986). Distributed Databases Principles and Systems, McGraw Hill.

2. Özsu, M., T., Valduriez,P. (1991). Principles of Distributed Database Systems, Prentice Hall, New Jersy.

3. Ceri,S.,Navathe, S., Widerhold, G.(1983). Distribution Design of Logical Database Schemas, IEEE Transactions on Software Engineering, Vol.9,No.4,487-504.

4. Tamhankar,A.,M.,Ram,S. (1998). Database Fragmentation and Allocation: An Integrated Methodology and Case Study, IEEE Transactions On Systems, Man, and Cybernetics, Vol.28,No.3,288-305.

5. Karahoca, A.(1998). Distributed Database System Design in A Technology-corridor, PhD thesis, University of Istanbul.

6. Ceri, S., Negri, M., Pelagatti, G.(1982). "Horizontal Data partitioning in Database Design", ACM SIGMOD, 128-136.

7. Shin, D., Irani, K.(1991). Fragmenting Relations Horizontally Using a Knowledge-Based Approach, IEEE Transactions on Software Engineering, Vol.17,No.9,872-883.

8. Seetharaman, A., Kai Y.,N. (1996). A Formal Approach for Horizontal Fragmentation of Distributed Disjunctive Deductive Database Design, http://lantern.cs.byu.edu.

9. Ceri, S., Navathe, S., Wiederhold, G., Don, Y.(1984). "Vertical Partitioning Algorithms for Database Design", ACM Transactions on Database Systems, 9 (4), 680-710.

10. Cornell, D. & Yu, P.(1987). A Vertical Partitioning Algorithm for Relational Databases, IEEE Database Engineering, 30-35.

11. Muhturaj, R., Chakravarthy, S., Varadarajan, R., Navathe, S., B.(1993). A Formal Approach to the Vertical Partitioning Problem In Distributed Database Design, In Proceedings of Parallel and Distributed Information Systems (PDIS-2), San Diego, 26-34.

12. Chakravarthy, S., Muthuraj, J., Vardarajan. R., Navathe, S.(1994). An Objective Function for Vertically Partitioning Relations in Distributed Databases and Its Analysis, Distributed and Paralles Databases,2,183-207.

13. Gruenwald, L. & Eich, M.(1993). Selecting a Database Partitioning Technique, Journal of Database Management, Summer, 27-39.

14. Gelenbe E., Stafylopatis A., "Global behavior of homogeneous random neural systems", Vol.15, pp. 534-541, 1991

15. Gelenbe E., "Learning in the recurrent random neural network", Neural Computation, Vol. 5, No. 1, pp 154-164, 1993.

16. Haykin S.(1999). Neural Networks Second Edition, Prentice Hall Inter. Inc., New York.

LIST OF FIGURES:

Table 1. Output parameters according to input parameters

Output parameter	***Related input parameter(s)***
Selecting columns from a relation, f_1	1, 5
Selecting a record, f_2	6
Updating a record, f_3	8
Deleting a record, f_4	6
Joining two relations, f_5	1, 5, 6
Altering a schema, f_6	1,2, 3, 4
Inserting a record, f_7	7

Source: Gruenwald, L. & Eich, M.(1993)

Table 2. Analysis Parameters

Parameter	**Definition**	**Values**
TAB_BOY	Relation length	1000, 2000, 3000, ..., 10000 record
SATIR_BOY	Record length	50, 100, 200, 300, 450 byte
SAYFA_BOY	Page length	512 byte
$KOLON\_BOY_k$	Length of column k	5, 10, 15 byte
FGRUP_BOY	Physical vertical group length	10, 20, 30 byte
KOLON_SAYI	Number of column	10, 20, 30

GRUP_SAYI	Number of group horizontal and group vertical	2, 4, 8
$GRUP\_BOY_g$	Length of group g in group vertical	5-120 byte ($KOLON\_BOY_k$ and GRUP_SAYI are used)
$SATIR\_SAYI\_GY_g$	Record number of group g in Group horizontal	125-50000 record (TAB_BOY and GRUP_SAYI are used)

Source: Gruenwald, L. & Eich, M.(1993)

Table 3-Classical Approach for Ordering of Fragmentation Techniques

Cost (Rank)	**Horizontal**	**Single Vertical**	**Physical**	**Group Vertical**	**Group Horizontal**	**Mix**
Number of pages	**4**	**6**	**2**	**5**	**1**	**3**
Reorganization	**5**	**6**	**4**	**3**	**1**	**2**
Column deletion	**2**	**5**	**5**	**4**	**1**	**3**
Column addition	**2**	**5**	**5**	**4**	**1**	**3**
Column access	**2**	**5**	**5**	**4**	**1**	**3**
Record recreation	**6**	**3**	**3**	**3**	**5**	**3**
Record addition	**6**	**3**	**3**	**3**	**5**	**3**
Record modification	**6**	**3**	**3**	**3**	**5**	**3**

Rank 6 is best, 1 is worst

Table 4-Ordering of Fragmentation Techniques for Eight Features According to Algorithm 1

Cost (%)	**Horizontal**	**Single Vertical**	**Physical**	**Group Vertical**	**Group Horizontal**	**Mix**
Number of pages	8	**84**	4	27	0	6
Reorganization	45	**71**	23	8	0	5
Column deletion	8	**45**	**45**	34	0	13
Column addition	8			34	0	13
Column access						

	8	**45** **45**	**45** **45**	34	0	13
Record recreation Record addition Record modification	**48** **48**	23 23 23	23 23 23	23 23 23	35 35 35	23 23 23

Table 5-Ordering of Fragmentation Techniques For Eight Features According to Algorithm 2

Cost (%)	**Horizontal**	**Single Vertical**	**Physical**	**Group Vertical**	**Group Horizontal**	**Mix**
Number of pages	8	**84**	4	27	0	6
Reorganization	45	**71**	23	8	0	5
Column deletion Column addition Column access	8 8 8	**50** **50** **50**	45 45 45	34 34 34	0 0 0	13 13 13
Record recreation Record addition Record modification	**48** **48** **48**	23 23 23	23 23 23	23 23 23	35 35 35	23 23 23

Table 6-Ordering of Fragmentation Techniques For Eight Features According to Random NN

Cost (%)	**Horizontal**	**Single Vertical**	**Physical**	**Group Vertical**	**Group Horizontal**	**Mix**
Number of pages	7,4	**71,73**	23,98	45,87	0	5
Reorganization	30	**47**	36	5	0	3
Column deletion Column addition Column access	13 13 13	**42** **41** **40**	32 34 32	18 17 17	0 0 0	9 7 10
Record recreation Record addition Record modification	**48** **48** **48**	19 19 19	18 18 18	18 18 18	27 27 27	19 19 19

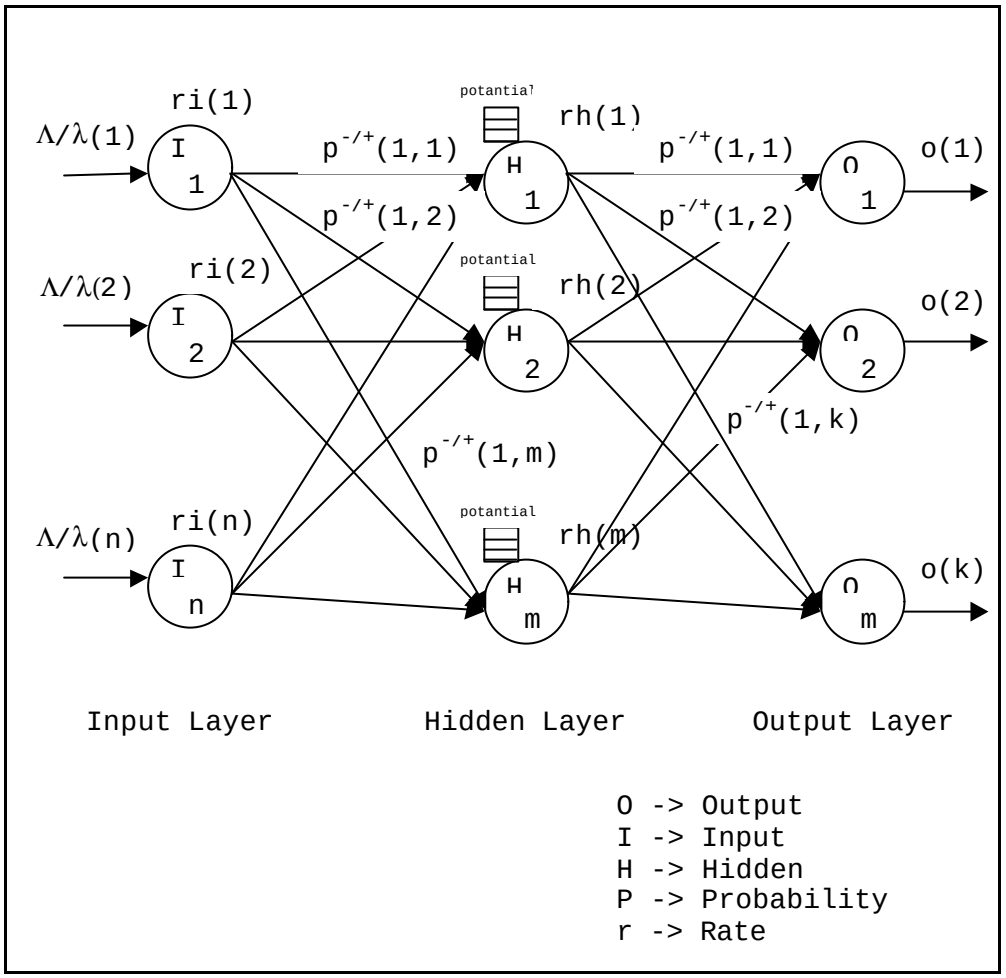

Figure 1-Representation Of RNN Model.

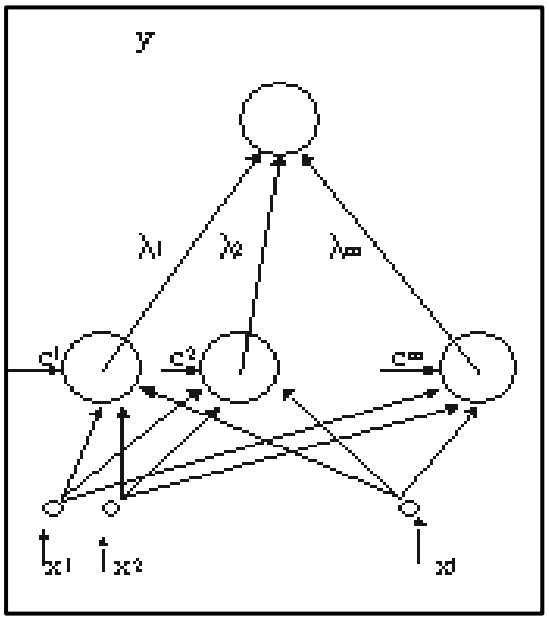

Figure 2-Radial Basis Function Networks

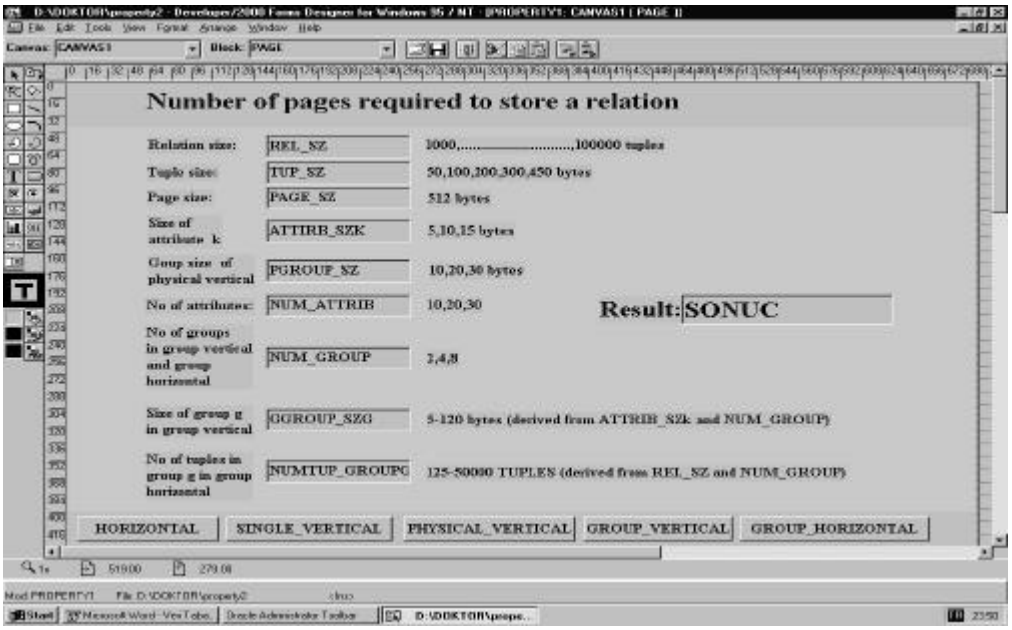

Figure 3- Calculation of the number of pages necessary for storing a relation according to the fragmentation techniques

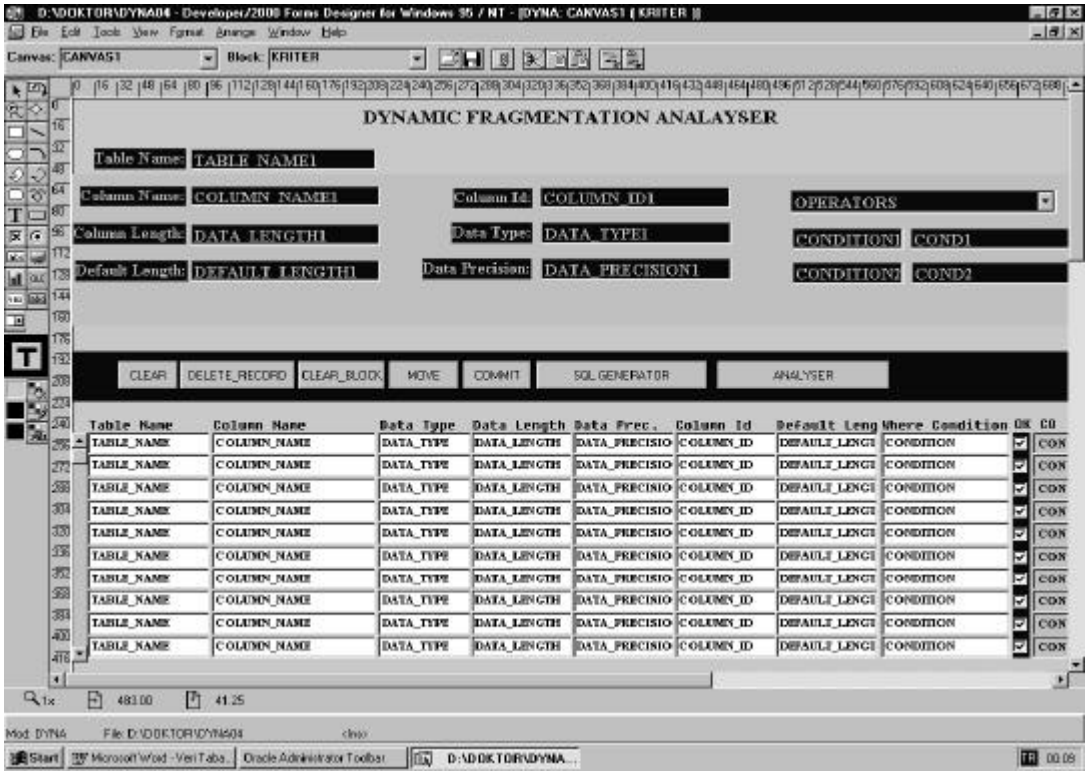

Figure 4-Program that generates the SQL statement dynamically and calculates fragmentation parameters

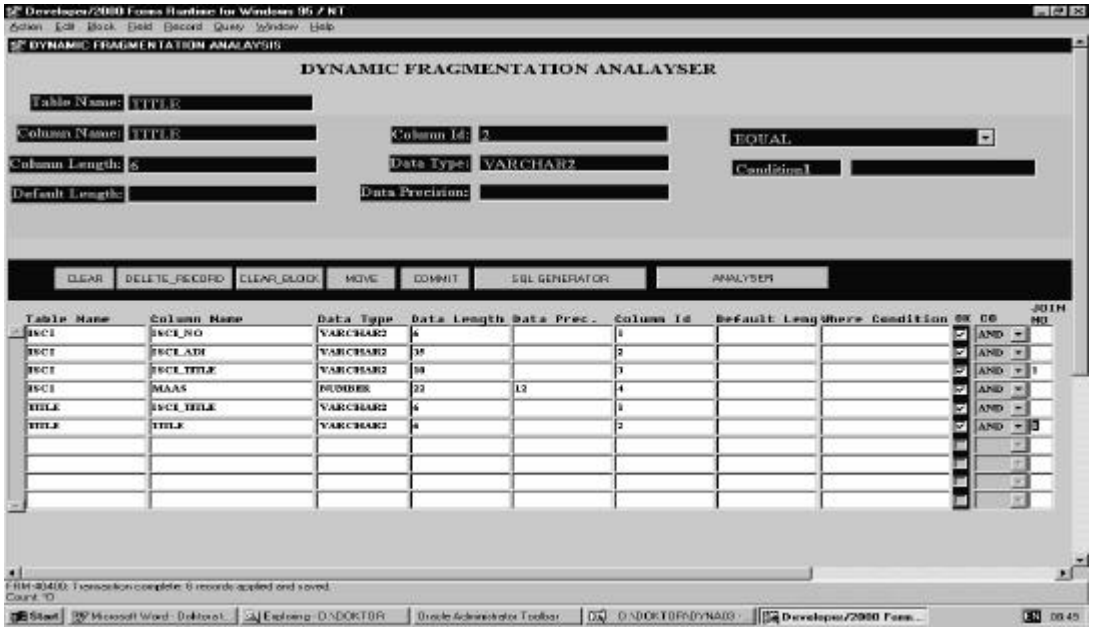

Figure 5-Selection of the columns of table and creation of the join relation between them

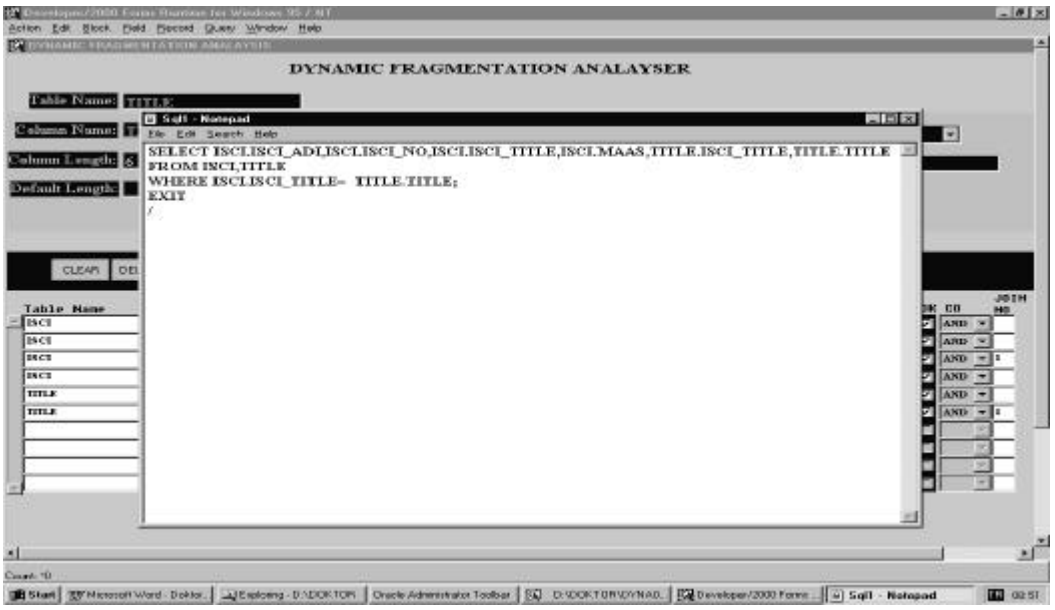

Figure 6-Obtaining the SQL statement with SQL Generator button

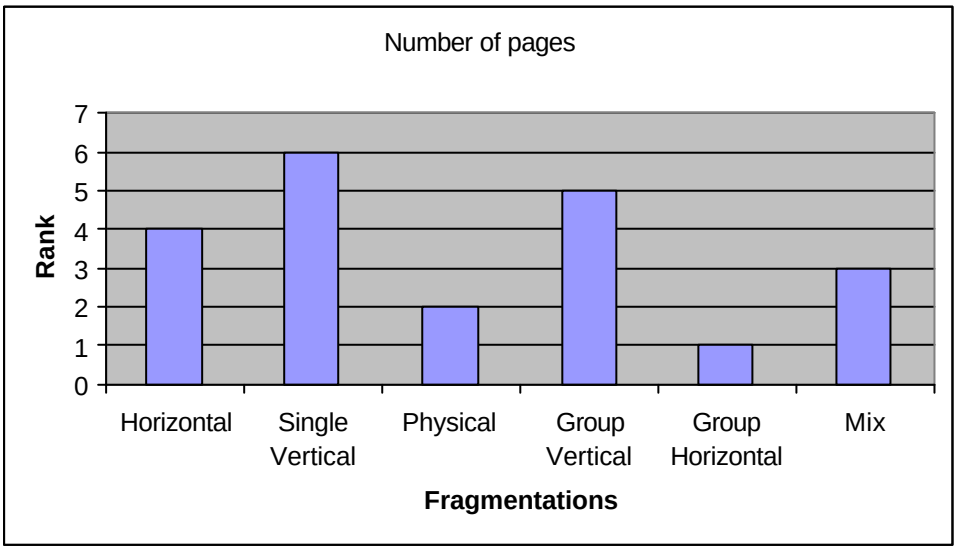

Figure 7-The best of the fragmentation techniques in terms of "the no. of pages " is the Single Vertical

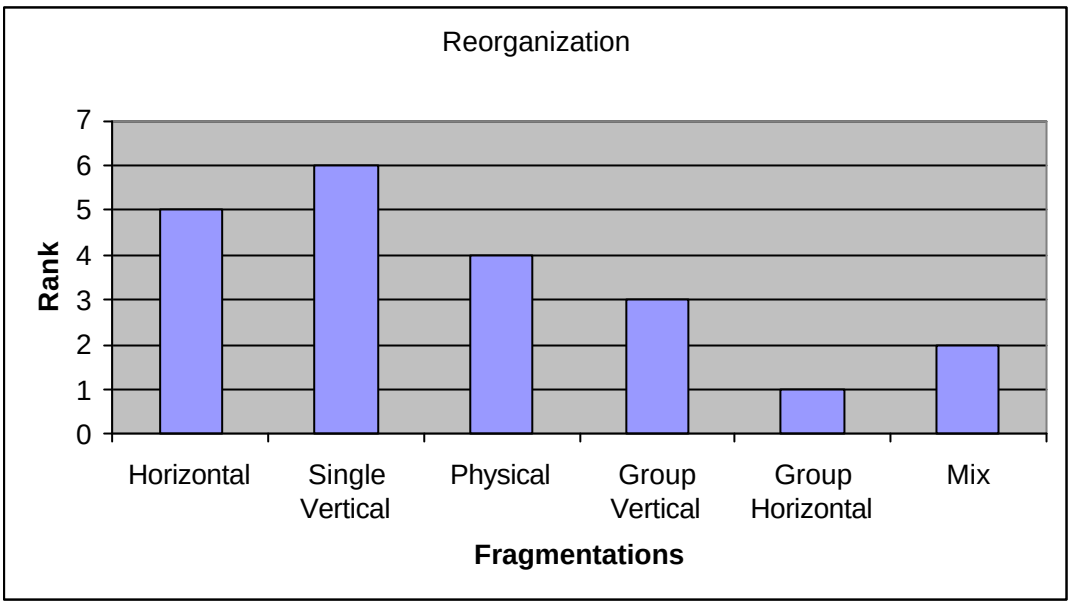

Figure 8-The best of the fragmentation techniques in terms of "reorganization" is the Single Vertical

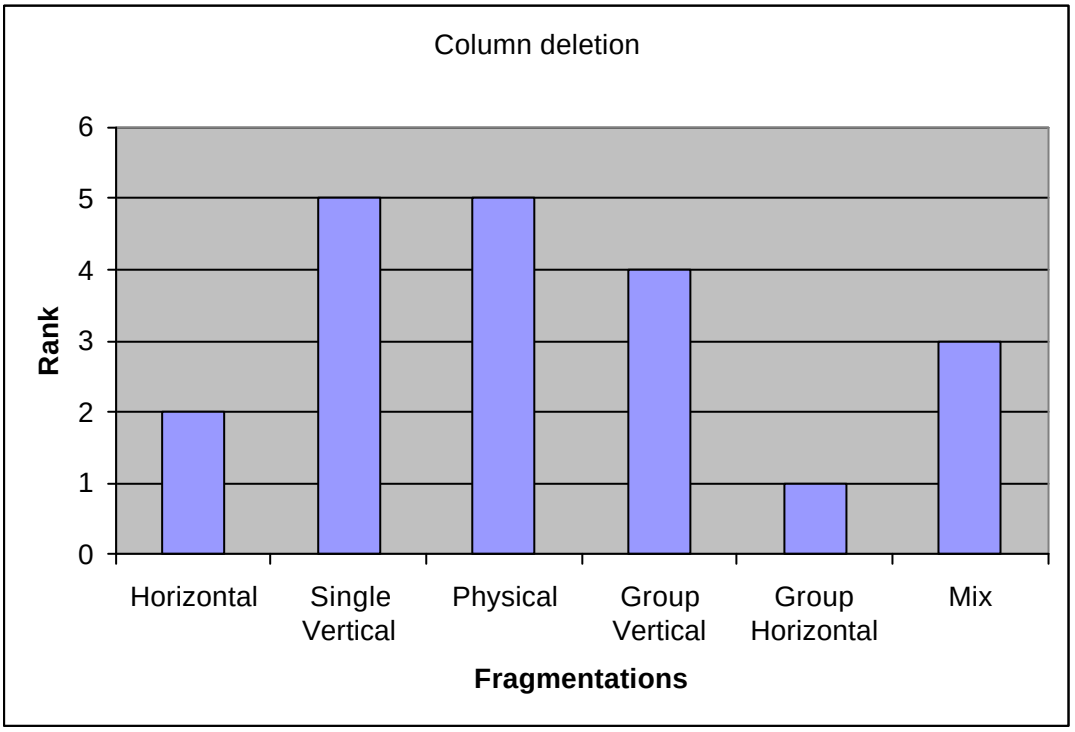

Figure 9-The best of the fragmentation techniques in terms of "column deletion" are the Physical vertical and the Single Vertical

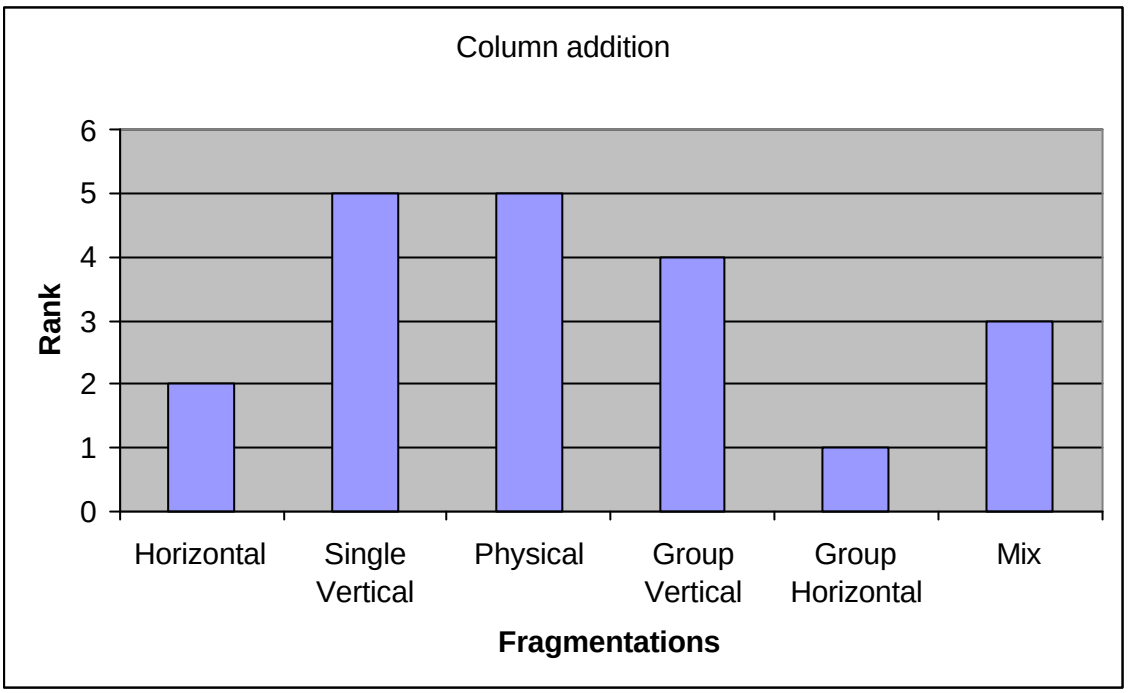

Figure 10-The best of the fragmentation techniques in terms of "column deletion" are the Physical Vertical and the Single Vertical

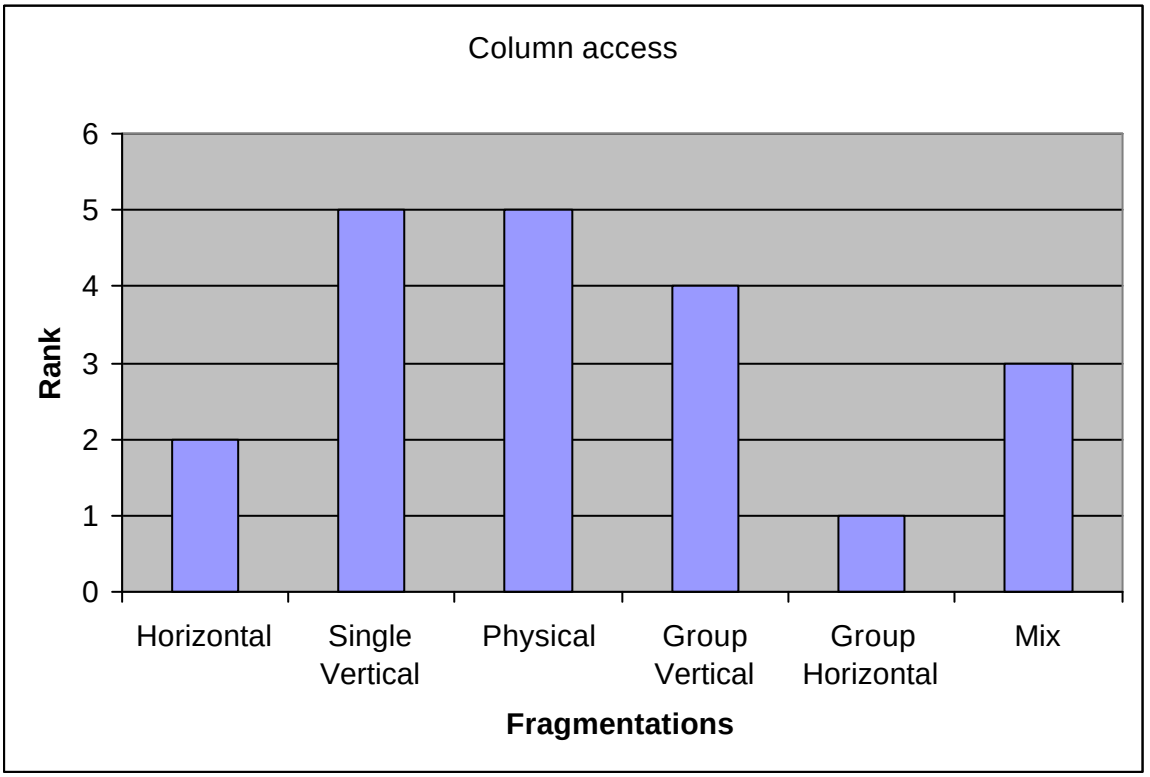

Figure 11-The best of the fragmentation techniques in terms of column access are the Physical Vertical and the Single Vertical

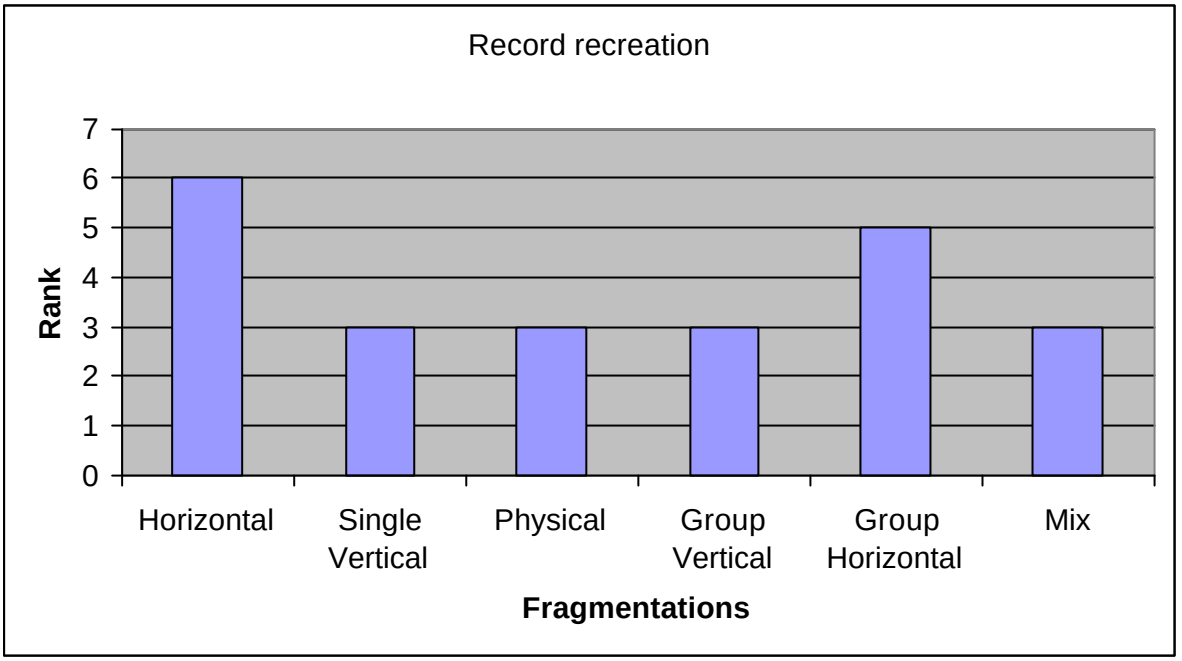

Figure 12-The best of the fragmentation techniques in terms of "record recreation" is the Horizontal

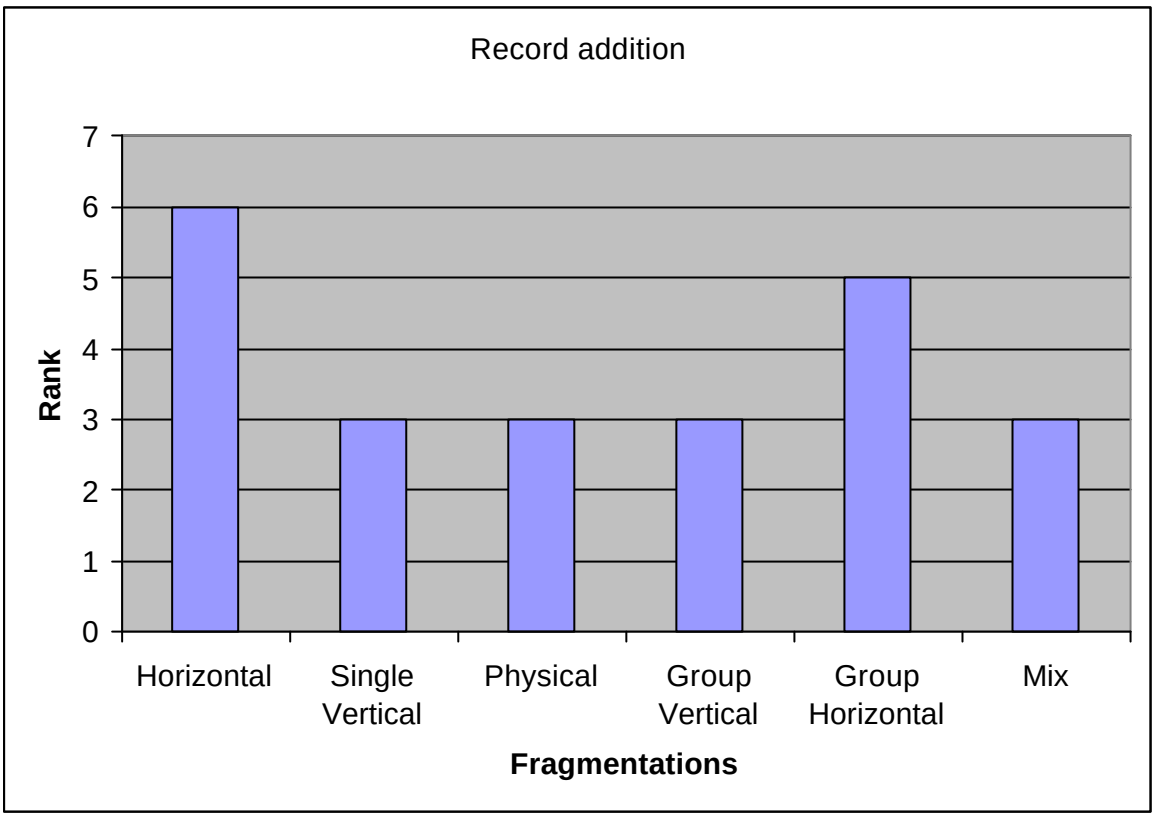

Figure 13-The best of the fragmentation techniques in terms of "record addition" is the Horizontal

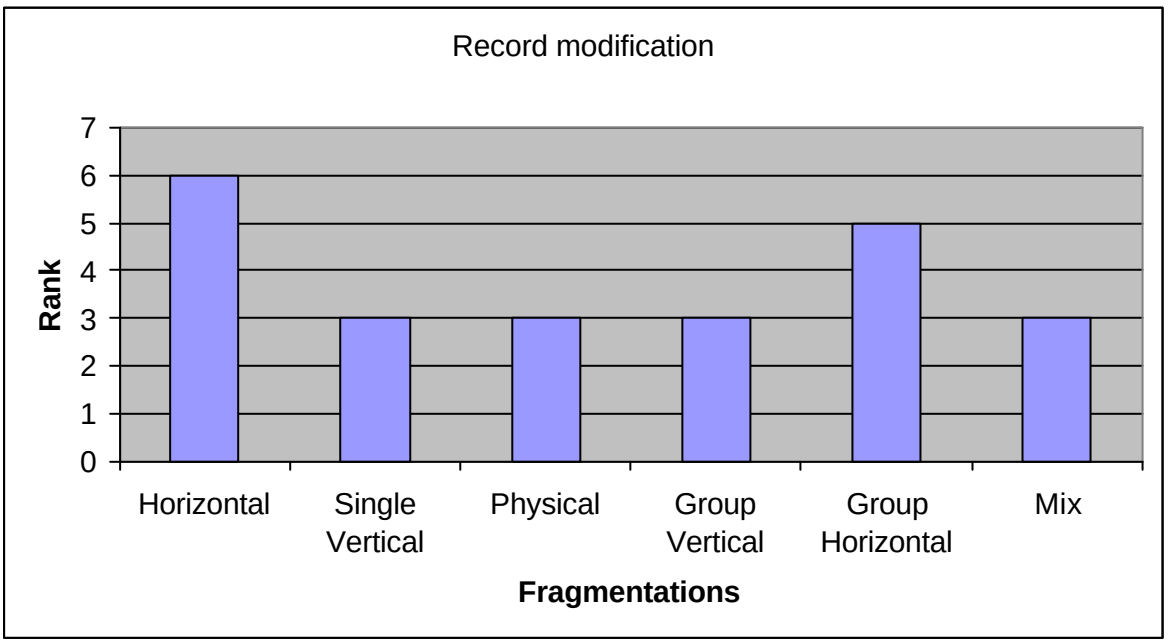

Figure 14-The best of the fragmentation techniques in terms of "column modification" is the Horizontal

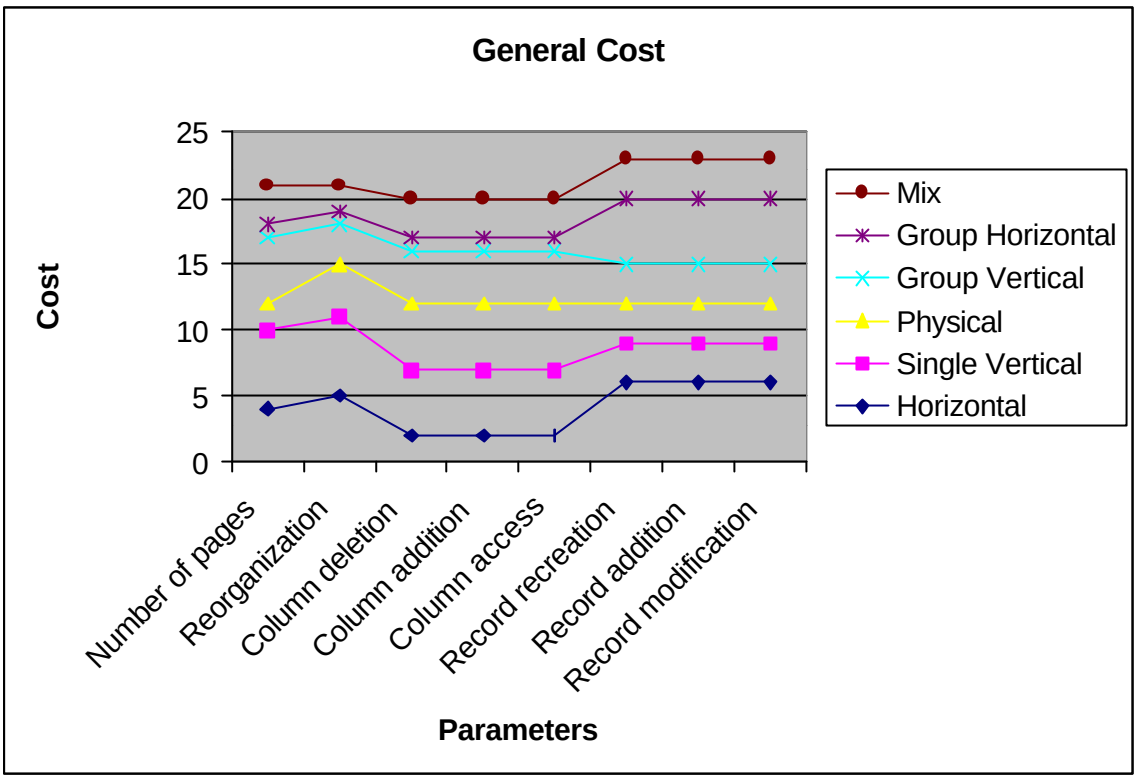

Figure 15-General cost function for linear approach

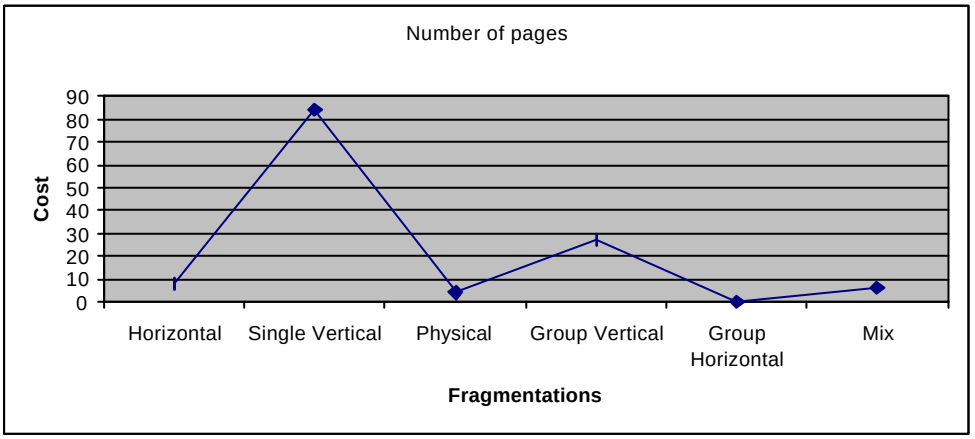

Figure 16-The best of the fragmentation techniques in terms of "the no. of pages" is the Single Vertical

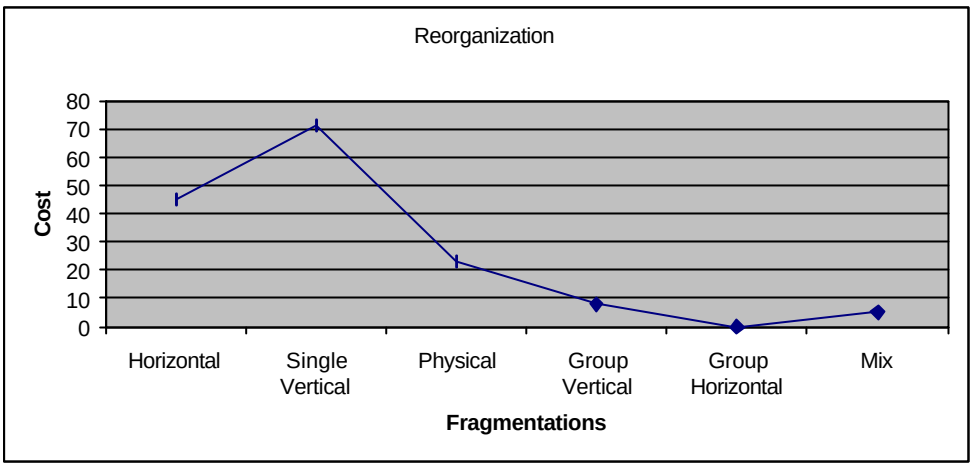

Figure 17-The best of the fragmentation techniques in terms of "reorganization" is the Single Vertical

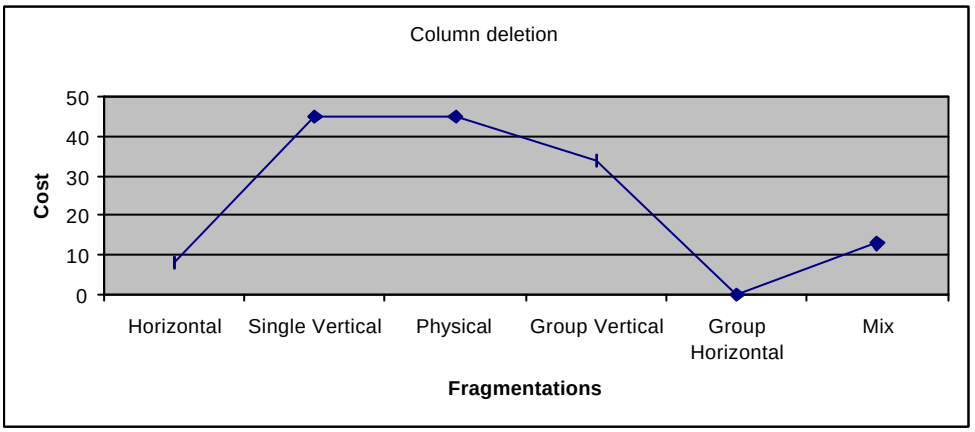

Figure 18-The best of the fragmentation techniques in terms of "column deletion" are the Physical vertical and the Single Vertical

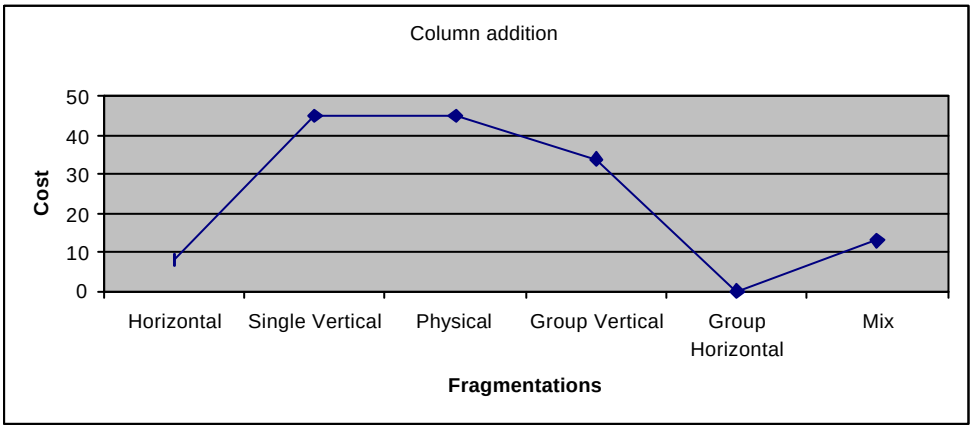

Figure 19-The best of the fragmentation techniques in terms of "column addition" are the Physical Vertical and the Single Vertical

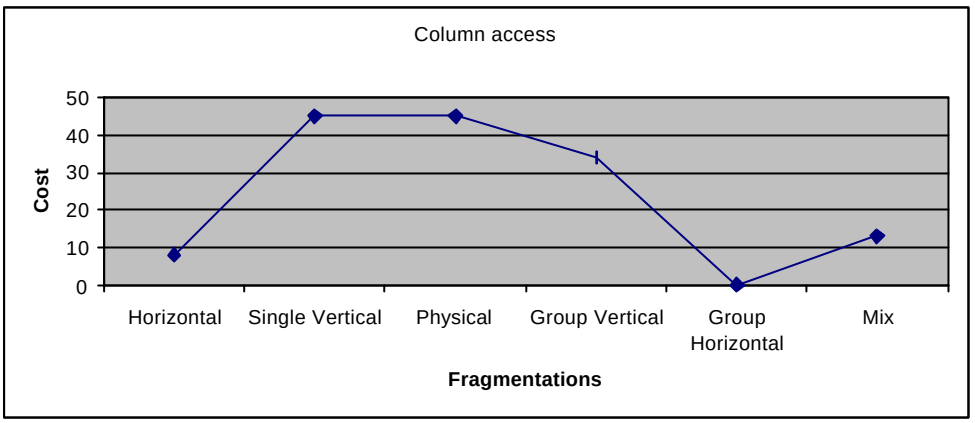

Figure 20-The best of the fragmentation techniques in terms of column access are the Physical Vertical and the Single Vertical

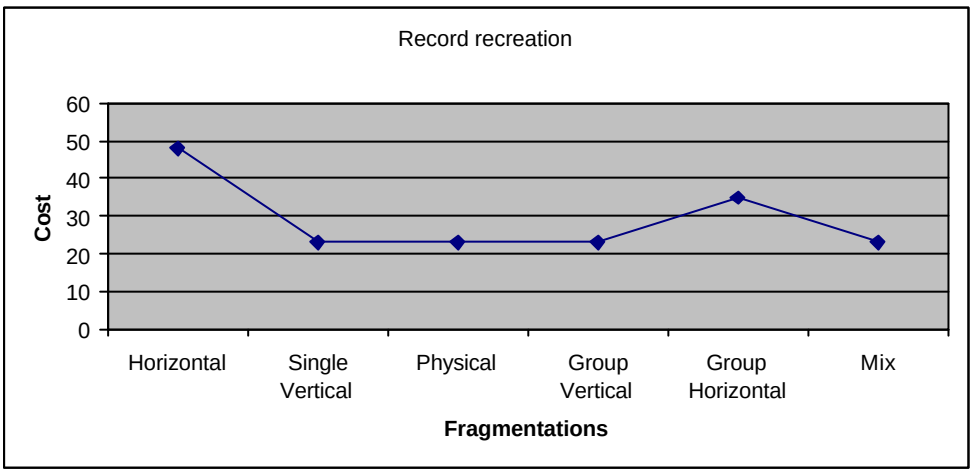

Figure 21-The best of the fragmentation techniques in terms of "record recreation" is the Horizontal

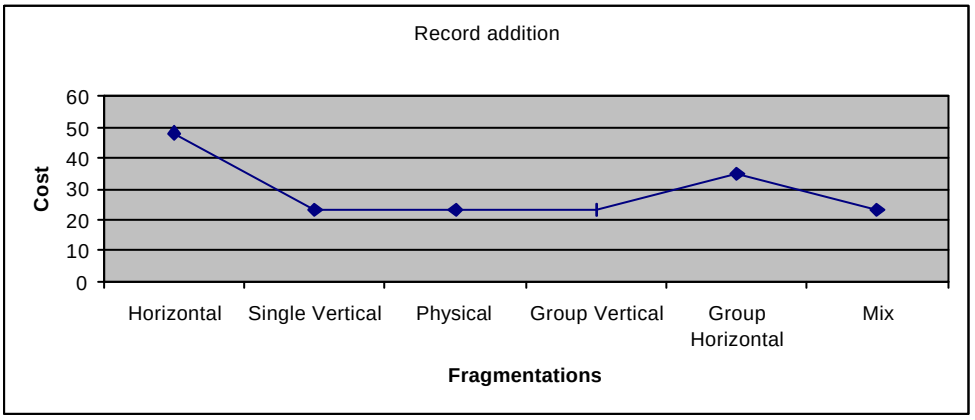

Figure 22-The best of the fragmentation techniques in terms of "record addition" is the Horizontal

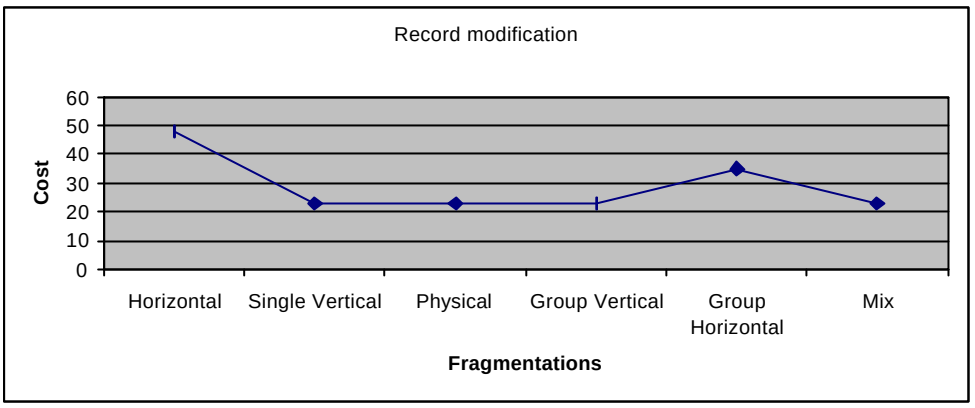

Figure 23-The best of the fragmentation techniques in terms of "column modification" is the Horizontal

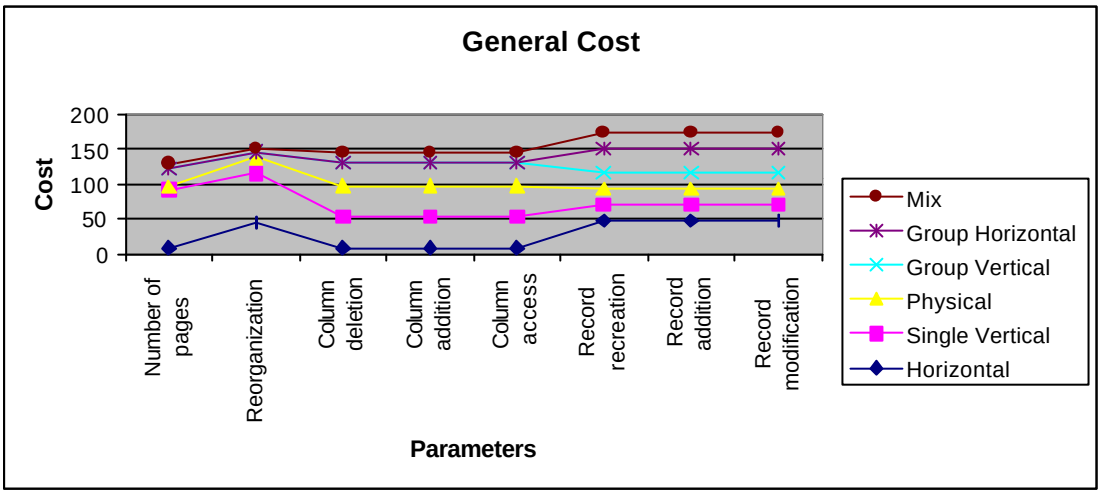

Figure 24-General cost function for radial basis functions neural networks

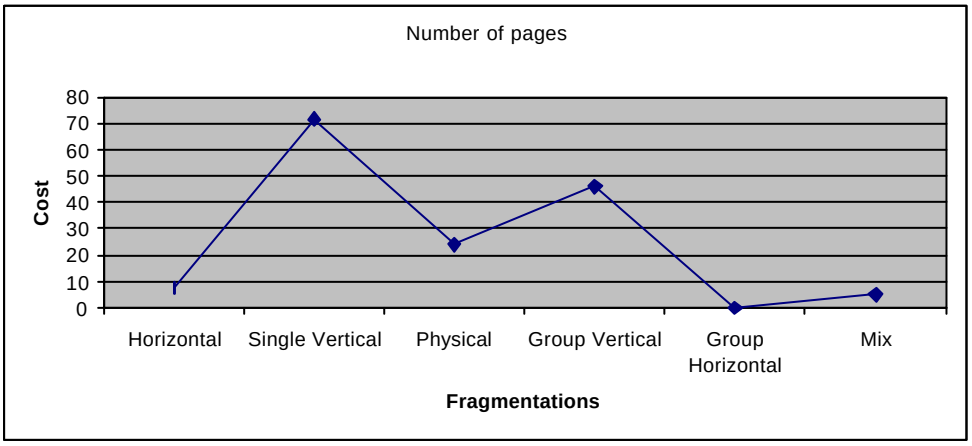

Figure 25-The best of the fragmentation techniques in terms of "the no. of pages" is the Single Vertical

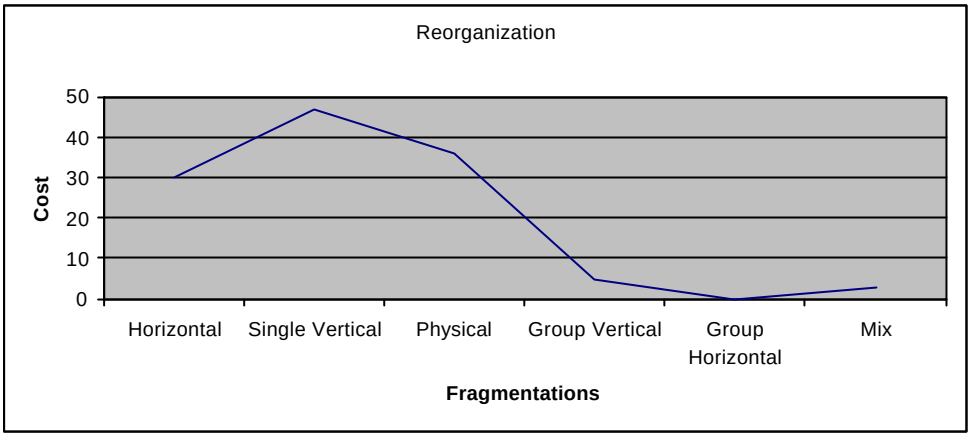

Figure 26-The best of the fragmentation techniques in terms of "reorganization" is the Single Vertical

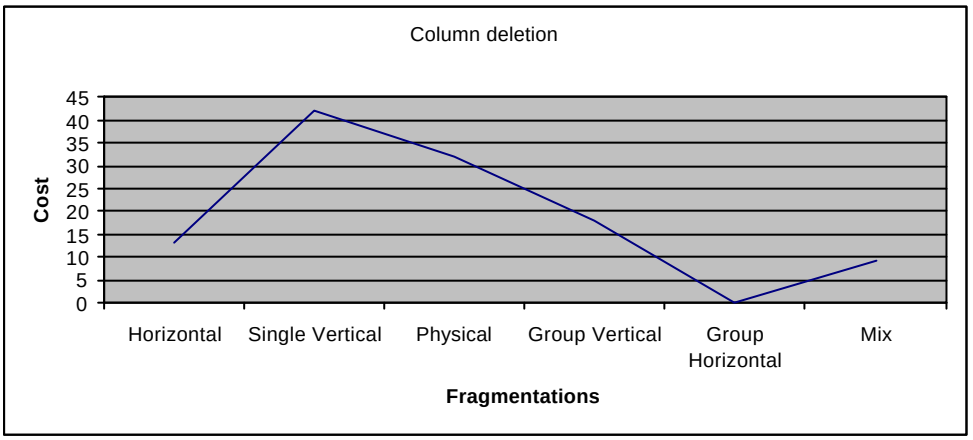

Figure 27-The best of the fragmentation techniques in terms of "column deletion" is the Single Vertical

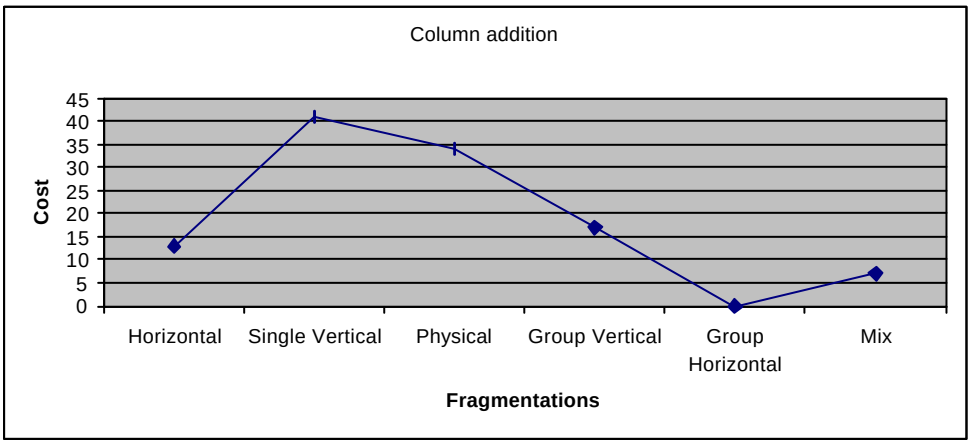

Figure 28-The best of the fragmentation techniques in terms of "column addition" is the Single Vertical

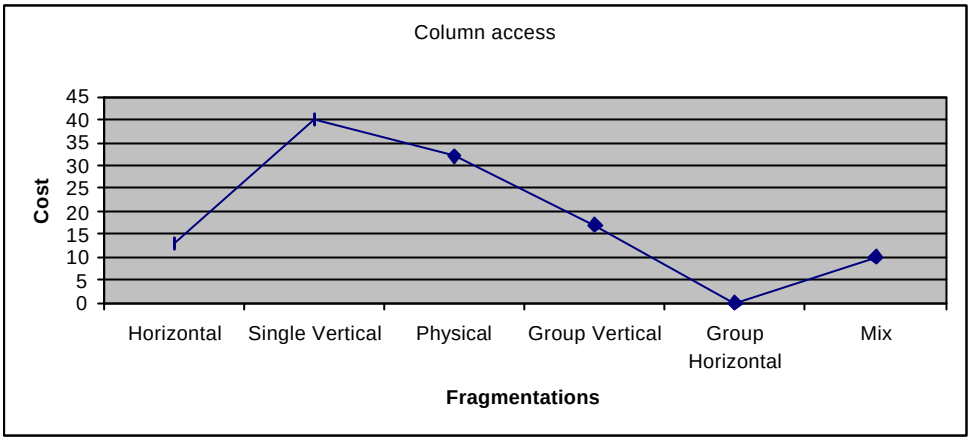

Figure 29-The best of the fragmentation techniques in terms of column access is the Single Vertical

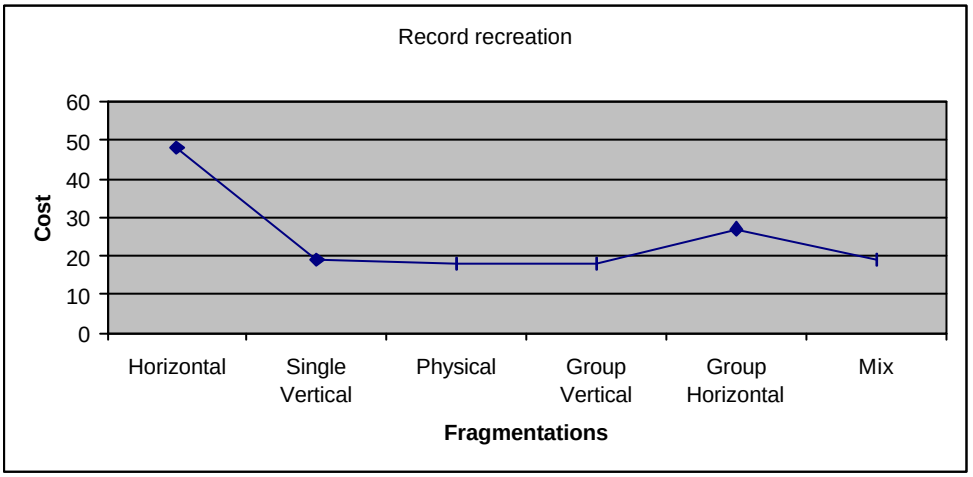

Figure 30-The best of the fragmentation techniques in terms of "record recreation" is the Horizontal

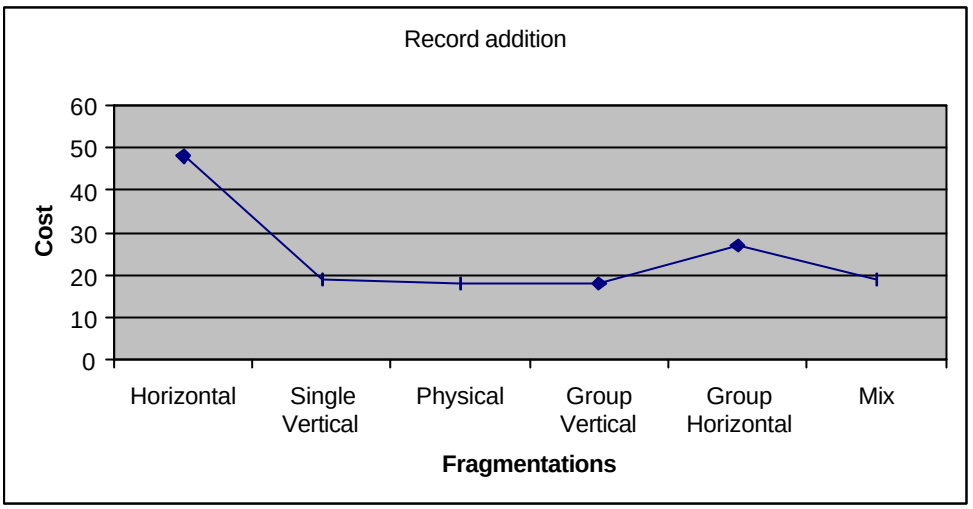

Figure 31-The best of the fragmentation techniques in terms of "record addition" is the Horizontal

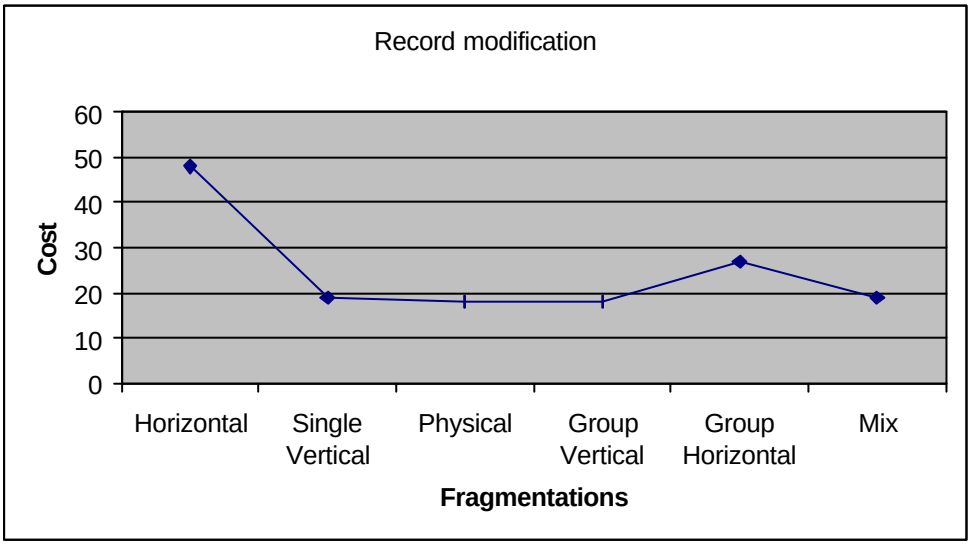

Figure 32-The best of the fragmentation techniques in terms of "column modification" is the Horizontal

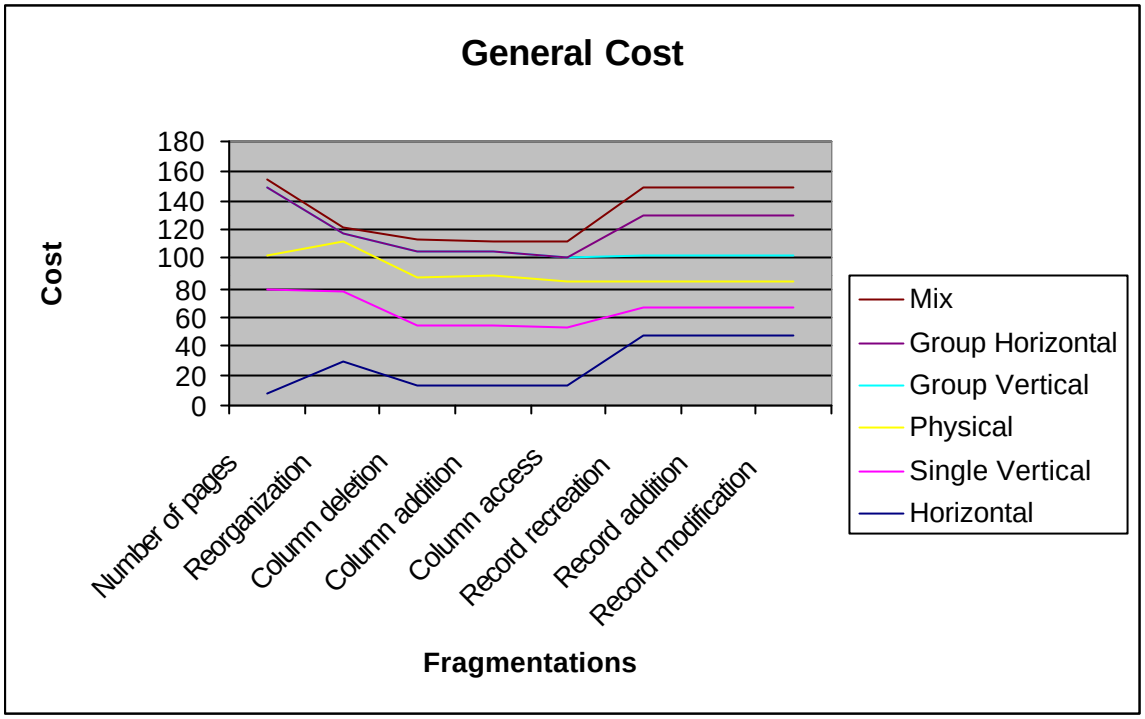

Figure 33-General cost function for random neural network

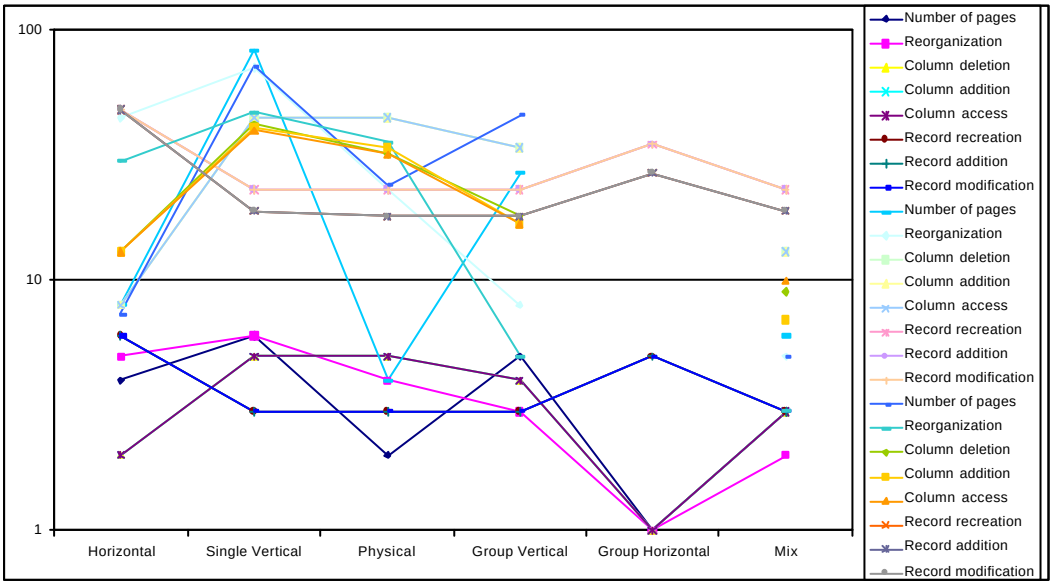

Figure 34-General cost comparsion within three approaches