International Journal of Soft Computing 7 (6): 283-293, 2012
ISSN: 1816-9503


# A Fuzzy Neural Network Based Compiler Directed Power Optimization for Disk Based Systems with Modified LM Training

G. Ravikumar and N. Nagarajan
Coimbatore Institute of Engineering and Technology, Coimbatore, India

**Abstract:** Power consumption of large servers and disks has become a popular research topic as this issue is important from both technical and environmental perspectives. The performance of the disk systems are greatly affected by extreme power consumption. A majority of the research in disk power management has concentrated on the behavior of the disk during periods of idleness. The main focus is on when the disk should be put to idleness to reduce power consumption without affecting the performance. Due to the increasing requirements of current and forthcoming data-intensive computer applications, there has been a chief alteration in the disk subsystem which now comprises of more disks with higher storage capacities and higher rotational speeds. Thus, disk power management has become a vital issue as it consumes very high power. This study proposes and evaluates an efficient compiler-directed disk power management technique which utilizes disk access schemes for reducing energy consumption. This study uses a novel approach called LMFNN in which the Fuzzy Neural Network is trained using Modified Levenberg Marquardt Learning algorithm. The proposed scheme analyzes the various disk access techniques and selects the suitable algorithm which would provide better overall performance of the disks. The experimental evaluation using a diverse set of workloads indicates that the proposed LMFNN approach provides better power consumption than the conventional approaches.

**Key words:** Fuzzy Neural Network (FNN), power optimization, disk idle period, subsystem, intensive

## INTRODUCTION

Disk Power Management (DPM) is a vital issue in modern computer systems. For several Battery-Oriented Mobile Systems, hard drives utilize a considerable segment of energy. It is the fact that disk systems make up almost 27% of the total energy consumption in a data center (Garrett, 2007). Thus, reducing disk power consumption is vital for both mobile computers and enterprise servers.

Large scale datacenters have to maintain thousands of rotating disks for fast online access of stored datasets (Gurumurthi *et al*., 2003). As the cost of the disks are getting reduced in terms of dollars per gigabyte, the forecast is that the energy costs for operating and cooling these rotating disks will ultimately outstrip the cost of the disks and the connected hardware required to control them.

Since, data intensive applications require fast and reliable access to online data resources, the percentage of power consumption by disk storage systems will continue to increase. This in turn necessitates the exploitation of power hungry faster (high RPM) and larger capacity disks. Thus, reducing power consumption is becoming a vital issue for high-end cluster/server based systems.

Especially, recent researches Bohrer *et al*. (2002) and Elnozahy *et al*. (2003) shows a huge segment of the system maintenance budget is spent in cooling because of the excessive power consumption of such systems. Moreover, high power consumption which requires sophisticated power generation and transmission technologies is known to be harmful to the environment. Therefore recent years have observed several efforts on minimizing energy consumption of disks (Chase *et al*., 2001; Chase and Doyle, 2001).

Numerous energy saving techniques for Disk Based Storage Systems have been proposed by various researchers (Irani *et al*., 2005; Weddle *et al*., 2007). A majority of these approaches revolve around the concept of spinning down the disks from their usual high energy mode (idle mode) into a lower energy mode (standby mode) after they experience a period of inactivity whose length exceeds a certain breakeven time (also called idleness threshold). The reason for this is that typical disks consume about one tenth of the power in standby mode as compared with their power consumption in idle model.

Majority of the present disks provide diverse power modes of operation such as active and idle. Active mode occurs when the disk is servicing a request and idle mode

**Corresponding Author:** G. Ravikumar, Coimbatore Institute of Engineering and Technology, Coimbatore, India

when it is spinning and not serving a request and one or more low power modes that use less energy than idle. Various researchers have taken up the chance of spinning down the disk during periods of idleness or serving the requests at lower rotational speeds when performance is not a problem. Disk idle periods are considered as a vital factor in those power consumption techniques (Douglis *et al*., 1994, 1995).

Most of the traditional techniques are reactive in manner and thus cannot provide efficient results in all the scenarios. The conventional disk power management techniques fail to estimate the disk idleness accurately (Elnozahy *et al*., 2002; Li *et al*., 1994). The selection of the appropriate disk access scheduling algorithm has not been effectively taken into consideration for the disk power management.

Disk access techniques greatly influence the power consumption of the disks. Focusing on large disk-intensive scientific applications with regular data access patterns, this study proposes and experimentally evaluates a compiler-based approach to disk power management. The proposed approach exposes disk layout information to the compiler, allowing it to derive the disk access pattern, i.e., the order in which parallel disks are accessed. The idea is to restructure an application code such that disk reuse is maximized, i.e., the application accesses as many data elements as possible from a set of disks before moving to the next set.

This approach uses the Fuzzy Neural Networks (FNNs) for selecting the suitable disk access scheduling algorithm. Neural fuzzy networks are the realizations of the functionality of fuzzy systems using neural networks (Nauck *et al*., 1997). Fuzzy Neural Networks (FNNs) have been extensively used to deal with the problems such as classification, identification, control, pattern recognition, etc. (Wang *et al*., 1997). A fuzzy system comprises of a group of fuzzy if-then rules. The main goal of a neural fuzzy network is its capability to model a problem domain through a Linguistic Model rather than Complex Mathematical Models. The Linguistic Model is basically a fuzzy rule base comprising of a group of IF-THEN fuzzy rules that are highly instinctive and easily understandable by the human users. Moreover, the black-box nature of the neural network paradigm is resolved as the connectionist framework of a neural fuzzy network fundamentally defines the IF-THEN fuzzy rules. Moreover, a neural fuzzy network can self alter the parameter of the fuzzy rules through neural network based learning approaches.

This study proposes compiler-directed disk power management technique which can accurately selects the appropriate disk access technique which in turn provides better power consumption. This technique focuses on developing an accurate and computationally efficient power consumption models for disk-based systems using the compiler directed energy optimization technique which saves disk power by using the Fuzzy Neural Network (FNN) to select the most appropriate disk scheduling algorithm and provides effective disk power management.

## LITERATURE SURVEY

Enhancing security and reducing power utilization are essential for large-scale data storage organizations. Even though numerous studies have been concentrated on data protection and energy efficiency, the majority of the available techniques have concentrated on just one of these two metrics. Yin *et al*. (2010) presented a novel technique to incorporate power optimization with security services to boost the security of energy efficient large scale storage organizations. In this approach, the dynamic speed control for power management procedure is used or DRPM to preserve energy in protected storage systems. The researcher had given two manners of incorporating privacy services with the dynamic disk speed control method. The first method-security aggressive in nature is mostly concentrated on the enhancement of storage system security with less importance on energy preservation. The second method provides advanced precedence to energy preservation as different to the security optimization. The experimental outcome shows that the energy-aggressive method offers better energy savings than the Security-Aggressive Method. On the other hand, the superiority of security realized by the Security-Aggressive Method is advanced than that of the Energy-Aggressive Method. Furthermore, the observed result shows that energy savings yielded by the two methods turn out to be more distinct when the data size is larger. The result demonstrates that the response time of the Security-Aggressive Method is more responsive to data size than that of the Energy-Aggressive Method.

Significant performance, high reliability and energy-proficient storage systems are very vital for mobile data-intensive applications such as remote surgery and mobile data center. Mobile disk-array-based storage systems are more liable to disk malfunctions than with traditional stationary storage systems. This is mainly because of their complicated application environments. Moreover, mobile disk-array-based storage has very inadequate power supply. Hence, data reconstruction techniques which are carried out in the existence of disk malfunctions for mobile storage systems must be performance-driven, reliability-aware and energy-efficient. Existing reconstruction approaches cannot accomplish

the three objectives concurrently as they mostly overlooked the information that mobile disks have much superior failure rates than stationary disks. In addition, they generally disregard energy-saving. In this study, Xie and Wang (2008) proposed a novel reconstruction approach, called Multi-level Caching-based Reconstruction Optimization (MICRO) which can be used to RAID-structured mobile storage systems to obviously cut down reconstruction times and user response times while saving energy. MICRO collaboratively uses storage cache and disk array controller cache to lessen the number of physical disk accesses produced by reconstruction. The simulation results reveal that MICRO technique lessens reconstruction times on average 20.22 and 9.34% when compared with the approaches like DOR and PRO. Moreover, it saves energy no <30.4 and 13%, respectively.

To maintain the huge storage necessities, consumer electronics for video playback are progressively more being outfitted with Hard Disk Drives (HDD) that use a considerable amount of energy. A video player possibly will prefetch several frames to provide a chance to disk to go to standby mode however, this might cause playback to be unclear or blocked if appropriate power mode transitions are not built-in. Go and Song (2008) provided the design, implementation and estimation of a data prefetching method for energy-aware video data retrieval for Portable Media Players (PMP). A difficulty is formulated when the prefetching is used for Variable Bit Rate (VBR) streams to diminish disk energy utilization and then developed a novel energy-aware data retrieval scheme that prefetches video data in a quick way in order to raise the period in which disk reside in standby mode while promising the real-time service. This method is implemented in the legacy video player known as Mplayer that is characteristically used for Linux-based consumer machines. Experimental observation shows that it saves energy to the extent that 51% compared with traditional methods.

## DISK ACCESS PATTERN EXTRACTION

The proposed approach uses compiler driven approach for power optimization. The basic necessity for utilizing a compiler in reducing disk power consumption is to identify the way in which the parallel disks are accessed at a high level. The compiler needs the data access pattern of the application code being optimized and disk layout information for the array data as shown in Fig. 1a and b.

The first of these can be attained by examining the application source code. For the second parameter required, the disk layout information is exposed to the compiler. Thus, the compiler will be aware of how array data are striped across the parallel disks and thus can optimize the source code.

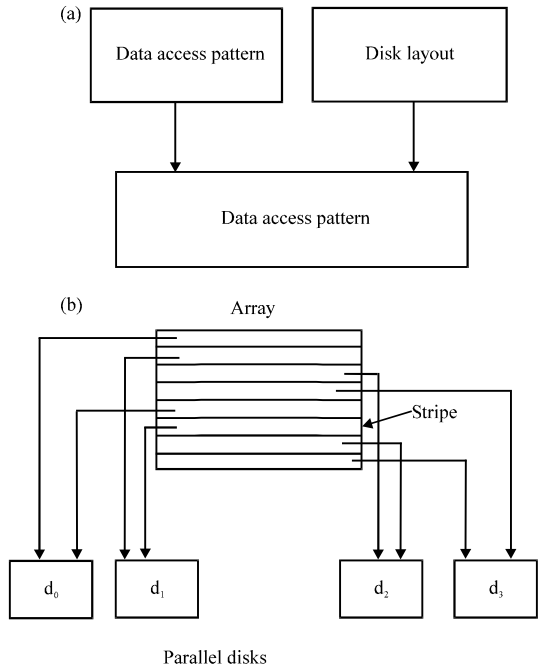


Fig. 1: a) Determining disk access pattern. b) Striping an array over four disks

File striping is a method that partitions a large data into small segments and stores these segments on separate disks in a round-robin fashion. This facilitates multiple processes to access various segments of the data concurrently without much disk disputation. In this research, a disk layout of an array is represented using a triplet of the form:

(starting_disk, stripe_factor, stripe_size)

It is observed that various existing file systems and I/O libraries for high-performance computing offer APIs to deliver the disk layout information when the file is created. For instance, in the Parallel Virtual File System (PVFS) (Ross *et al*., 2002), the default striping parameters can be altered by setting base (the first I/O node to be used), pcount (stripe factor) and ssize (stripe size) fields of the pvfs_filestat structure.

Then, the striping information given by the user through this pvfs_filestat structure is passed to the pvfs_open() call's parameter. When building a file from within the application, this layout data can be made accessible to the compiler and moreover the compiler uses this information in conjunction with the data access pattern it extracts to determine the disk access pattern (Son *et al*., 2007).

It is to be noted that each data array operated by the application is stored in a separate file in the I/O System. As each file can have a distinct triplet, each array can have a distinct disk layout than the others. Thus, the disk access techniques become very vital in optimizing the disk access. Therefore, a novel technique that chooses the best disk access technique through neural networks is proposed in this approach.

**Proposed approach:** The patterns in which the disk is accessed play a vital role in the energy optimization. Scheduling algorithms help in identifying the way in which the disk is accessed. The proposed approach focuses on selecting the most suitable scheduling algorithm in relation to the system's load because operating systems usually use the same scheduling algorithm irrespective of this. To carry out this selection, the approach proposes the use of artificial neural networks.

**Disk seek algorithms:** The most commonly used strategies for disk scheduling for operating systems are the following.

**FCFS planning (First Come, First Served):** In this type of algorithm the first to arrive is the first to be served, so a request cannot be displaced by the arrival of a new one. There is no reordering of the queue of requests, the positional relationship between the waiting requests is ignored and although it offers a small variance, it is detrimental to requests situated at the end of the queue.

**SSF planning (Shortest Seek First):** In order to lessen the movements of the access method, the disk arm moves to the request nearest to its actual position. It does not consider the order in which requests turn up in the queue, facilitating the central tracks of the disk. This causes the average response time for moderate loads of this approach lower than that of the FCF approach.

Even though this approach carry out well in most cases (it reduces arm movement), it can construct the inanition of certain requests as they are delayed by the advent of new ones that need less arm movement. In order to eradicate the issue of inanition and to make the approach more deterministic a modification of the SSF algorithm is proposed in which requests are dealt in closed batches. Therefore, a set of requests served in its entirety before another is concentrated and the predictability of the approach is increased considering the average response time of the diverse requests.

**SCAN planning:** The Scan or Elavator approach has been the fundamental of most implemented planning approaches. The main approach of this technique is displacing the disk arm to serve all the requests it identifies in its path, only changing direction when there are no requests in the prevailing direction. Thus, the outer tracks are more visited, so reducing one of the main issues of the (original) SSF algorithm.

This technique has given rise to different modifications among which the n-steps scan stands out. In this scenario, only the requests which are identified when starting a run are served. The requests which arrive during a run are grouped together and ordered to be served on the return run.

A second modification has been the C-Scan (Circular Search) algorithm. This approach moves the disk arm from the outer cylinder to the interior. On finalizing the run, it leaps to the request nearest the outer cylinder and initiates again. The third modification has been the Eschenbach approach. This algorithm moves the disk arm like in the C-Scan but the requests are served according to the arrival time of the sector to the head, i.e., it takes into account rotational delay or latency. In the case of two requests transferring sector positions inside a cylinder, it only serves one in the prevailing arm movement.

**Selection of disk seek algorithms using fuzzy neural networks trained using Modified Levenberg Marquardt algorithm:** Each disk scheduling approach generates a various number of jumps of the access mechanism, their appropriateness is based on the characteristics of the input/output requests (load) submitted to the system. The jumps show the number of cylinders through which the disk arm passes without stopping to serve an input/output request. These access approach jumps creates delays in the transfer of information, taking up most of the time of an input/output operation. As a result, the selection of the most suitable seek algorithm for each array of input/output requests is essential to globally enhance the performance of the system.

Thus, the performance of the system can be enhanced by using the Fuzzy Neural Network (FNN) to select the most suitable seek algorithm for the load of the system.

In this study, novel Fuzzy Neural Networks (FNNs) combining with Modified Levenberg Marquardt learning mechanism called Modified LM based Fuzzy Neural Networks (LMFNN) are proposed for effective selection of the most appropriate seek algorithm.

The LMFNN integrate the ability of reducing the error oscillations and converging to desired outputs of Modified Levenberg Marquardt algorithm in high dimensional data spaces and the efficient human-like reasoning of FNN in handling uncertainty information.

**FNN architecture:** The proposed SVFNN is a four-layered FNN that is comprised of the input, membership function, fuzzy rules and output layers as shown in Fig. 2. Layer 1 accepts input variables whose nodes represent input linguistic variables.

Layer 2 is to compute the membership values whose nodes denote the terms of the relevant linguistic variables. Fuzzy rules are denoted by the nodes at Layer 3. The links before Layer 3 denote the preconditions of fuzzy rules and the links after Layer 3 denote the effects of fuzzy rules.

Layer 4 is the output layer. This four-layered network recognizes the following form of fuzzy rules:

- Rule $R_j$: If $x_1$ is $A_{1j}$ and ... $x_i$ is $A_{ij}$... and $x_M$ is $A_{mj}$, THEN y is $d_j$, j = 1, 2, ..., N. Where, $A_{ij}$ represent the fuzzy sets of the input variables $x_i$, i = 1, 2, ..., M and $d_j$ are the consequent parameter of y

For the simplicity of analysis, a fuzzy rule 0 is added as:

- Rule 0: IF $x_1$ is $A_{10}$ and $x_M$ is $A_{M0}$, THEN y is $d_0$. Where, $A_{k0}$ denotes a universal fuzzy set whose fuzzy degree is 1 for any input value $x_i$, i = 1, 2, ..., M and $d_0$ represents the resulting parameter of y in the fuzzy rule 0. $O^{(P)}$ and $a^{(P)}$ are defined as the output and input variables of a node in layer P, respectively

The signal propagation and the fundamental functions in each layer are illustrated as follows.

**Layer 1 (Input layer):** No calculation is done in this layer. Each node in this layer which is equivalent to one input variable, only transmits input values to the next layer directly. That is:

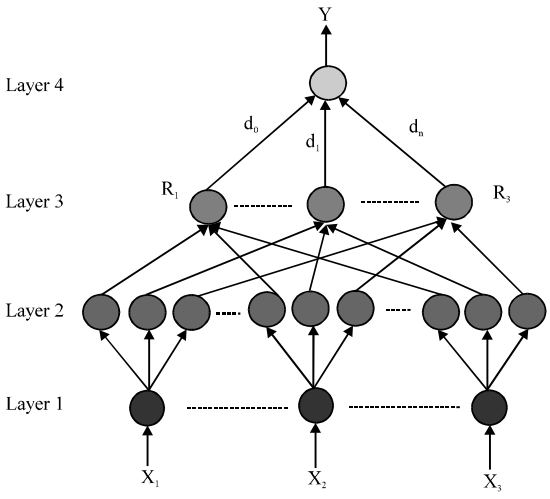


Fig. 2: Structure of the four-layered Fuzzy Neural Network

$$O^{(1)} = a_i^{(1)} = x_i \tag{1}$$

where, $x_i$, i = 1, 2, ...., M are the input variables of the FNN.

**Layer 2 (Membership function layer):** Each node in this layer is a membership function that is equivalent to one linguistic label of one of the input variables in Layer 1. On the other hand, the membership value which indicates the degree to which an input value belongs to a fuzzy set is computed in Layer 2:

$$O^{(2)} = u_i^{(j)}(a_i^{(2)}) \tag{2}$$

where, $u_i^j = (.)$ is a membership function $u_i^j = (.)$: R→[0, 1], i = 1, 2, ..., M, j = 1, 2, ..., N. With the use of Gaussian membership function, the operation performed in this layer is:

$$O^{(2)} = e^{-\frac{(a_i^{(2)} - m_{ij})^2}{\sigma_{ij}^2}} \tag{3}$$

Where:

$m_{ij}$ = The center (or mean)

$\sigma_{ij}$ = The width (or variance) of the Gaussian membership function of the jth term of the ith input variable $x_i$

**Layer 3 (Rule layer):** A node in this layer denotes one fuzzy logic rule and carry out precondition matching of a rule. AND operation is used for each Layer 2 node:

$$O^{(3)} = \prod_{i=1}^{M} a_i^{(3)} = e^{-[D_j(x-m_j)]^T[D_j(x-m_j)]} \tag{4}$$

where, $D_j = \text{diag}[1/\sigma_{1j}, ..., 1/\sigma_{Mj}]$, $m_j = [m_{1j}, m_{2j}, ..., m_{Mj}]^T$, $x = [x_1, x_2, x_3, ..., x_M]^T$ is the FNN input vector of FNN. The output of a Layer 3 node denotes the firing strength of the equivalent fuzzy rule.

**Layer 4 (Output layer):** The single node $O^{(4)}$ in this layer is labeled with Σ which calculates the overall output and can be computed as:

$$O^{(4)} = \sum_{j=i}^{N} d_j \times a_j^{(4)} + d_0 \tag{5}$$

where, the connecting weight $d_j$ is the output action strength of the Layer 4 output associated with the Layer 3 rule and the scalar $d_0$ is a bias. Thus the fuzzy neural network mapping can be rewritten in the following input-output form:

$$O^{(4)} = \sum_{j=i}^{N} d_j \times a_j^{(4)} + d_0 = \sum_{j=i}^{N} d_j \prod_{i=1}^{M} u_i^{(j)} + d_0 \quad (6)$$

**Construction of fuzzy rules:** Suppose that there are n requests: $R_1,..., R_n$. The processing time $p_i$ are deterministic and the deadlines are $d_i$. But a certain delay is tolerable up to a Late date $d_i^*$ beyond which the request will be canceled because the customer will resort to other services. As completion time of job $J_i$ passes between due date $d_i$ and late date $d_i^*$, the satisfaction of request decreases until it vanishes at the latter. Figure 3 shows a fuzzy deadline and its associated membership function.

The greater the delay, the lower the satisfaction. In this research, preemptions for the requests aren't admitted and the object is to find optimal assignment of the priorities using classical ordering of fuzzy sets.

Let $C_i$ denote completion time of job $J_i$ and the membership function $S_i(C_i)$ denoted degree of satisfaction is associated with each request and its deadline which is defined as follows:

$$S_i(C_i) = \begin{cases} 1 & C_i \le d_i \\ \dfrac{1-(C_i-d_i)}{(d_i^*-d_i)} & d_i < C_i < d_i^* \\ 0 & d_i \le C_i \end{cases} \quad (7)$$

If complete time $C_i$ is before due date $d_i$, there is maximum degree of satisfaction that is one. When complete time delays beyond $d_i^*$, level of satisfaction will decrease to zero. Generally, $C_i$ is in the interval $(d_i, d_i^*)$, and $u_i(C_i)$, degree of satisfaction with respect to $C_i$ belongs to (0, 1). Let the feasible schedule denoted by $\prod$. Then, minimum degree of satisfaction with schedule $\prod$ is:

$$S_{min} = \min\{S_i(C_i(\prod)) \mid i = 1, 2, ..., n\} \quad (8)$$

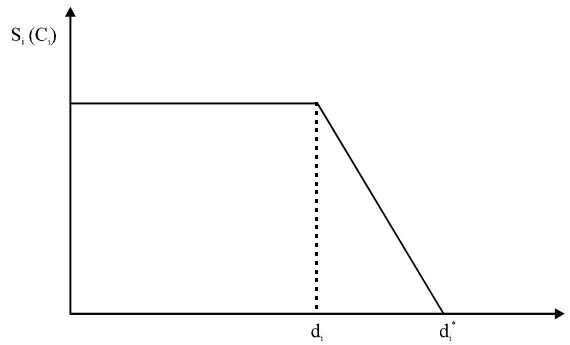


Fig. 3: Membership function of fuzzy deadline

Under the above setting, consider the following problem P:

$$\text{Maximize } S_{min} = \min\{S_i(C_i(\prod)) \mid i = 1, 2, ..., n\}$$
$$\text{Subject to } S_i(C_i(\prod)) > 0 \quad (9)$$

That is to get an optimal schedule $\prod$ which maximizes the minimum satisfaction with respect to fuzzy deadlines of jobs. Now, let $t = S_{min}$. Obviously $0 \le t \le 1$. Then, the inequalities:

$$C_i \le d_i + (1-t)\, e_i \text{ and } e_i = d_i^* - d_i, \text{ for } 1 \le i \le n \quad (10)$$

must hold. Next, employ the five rules to solve above problem P.

**Rule 1:** If $t>0$ then each optimal disk schedule satisfies inequality.

**Rule 2:** If $t = 0$ then for each disk schedule, there exists request $R_k$ such that:

$$d_i + (1-t)e_k < C_k \quad (11)$$

For each $0<t<1$, let $D_i(t)$ be modified deadlines for completion of requests:

$$D_i(t) = d_i + (1-t)e_i, \text{ for } 1 \le i \le n \quad (12)$$

**Rule 3:** Only those disk schedules are feasible whose modified deadlines $D_i(t)$ $(1<i\le n)$ satisfy following inequality for some t $(0<t\le 1)$:

$$C_i \le D_i(t) \quad (13)$$

If no such feasible schedule exits then every disk schedule is optimal and the maximum of the minimum degree of satisfaction is zero.

**Rule 4:** For each fixed value of t, the optimal assignment of priorities exists if and only if the priorities induced by ordering the requests in the non-decreasing order of $D_i(t)$ are feasible. Since, the modified deadlines change accordingly with t, researchers have to determine the values:

$$t_{ij} = 1 + \frac{d_i - d_j}{e_i - e_j}, \text{ for some } i, j, i \ne j \quad (14)$$

Rank all the quantities $t_{ij}(0<t_{ij}<1)$ in increasing order and create the interval $[t_k, t_{k+1}]$ $(k = 0,..., k_{max})$. All of these intervals follow such rule.

**Rule 5:** For any t in the interval $[t_k, t_{k+1}]$, the requests must have same priorities. If t passes from interval $[t_k, t_{k+1}]$ to interval $[t_{k+1}, t_{k+2}]$, the requests must change the priorities.

Above rules suggest the following procedure for finding an optimal disk schedule. First, find all points of intersection $t_{ij}$ of $D_i$ and $D_j$ satisfying $0<t_{ij}<1$, then sort them and employ a binary search for finding the maximum among them for which a feasible schedule exits according Rule 4. If no feasible schedule exits then every disk schedule is optimal. The algorithm returns the resulting schedule $\prod$ as a permutation, i.e., $\prod$ returns the index. $\prod(i)$ of the request in the ith for each $1 \le i \le n$.

**Algorithm**

**Step 1:** Find all $0 \le t \le 1$ such that $t = (e_i - e_j + d_i - d_j)/(e_i - e_j)$, for some $i \ne j$. Let $\pi$ be an optimal schedule.

**Step 2:** Let T be a set of such t. Add 1 to T. Sort all items in T in decreasing order such that t = 1 is first one. Choose the initial t from T.

**Step 3:** Find schedule S according to the non-decreasing order of $D_i(t)$. If $C_i(s) \le D_i(t)$ for all $1 \le i \le n$, then $\pi \leftarrow S$, stop; else go to Step 4.

**Step 4:** Select next t from T. Go to Step 3.

**Learning algorithm:** FNN is trained using the Modified Levenberg Marquardt Method algorithm (Suratgar *et al.*, 2005). In the FNN algorithm, the performance index F (w) to be minimized is defined as the sum of squared errors between the target outputs and the network's simulated outputs:

$$F(w) = e^T e \tag{15}$$

Where:

w = [w1, w2, ...,wN] consists of all weights of the network

e = The error vector comprising the error for all the training examples

When training with the LM Method, the increment of weights $\Delta w$ can be obtained as follows:

$$\Delta w = [J^T J + \mu I]^{-1} J^T e \tag{16}$$

Where:

J = The Jacobian matrix

$\mu$ = The learning rate which is to be updated using the $\beta$ depending on the outcome

In particular, $\mu$ is multiplied by decay rate $\beta$ $(0<\beta<1)$ whenever F (w) decreases whereas $\mu$ is divided by $\beta$ whenever F (w) increases in a new step. The standard LM training process can be illustrated in the following pseudo-codes:

- Initialize the weights and parameter $\mu$ ($\mu$ = 0.01 is appropriate)
- Compute the sum of the squared errors over all inputs F (w)
- Solve (2) to obtain the increment of weights $\Delta w$
- Recomputed the sum of squared errors F (w)

Using $w + \Delta w$ as the trial w and judge. If trial F (w)<F (w) in Step 2 THEN:

$$w = w + \Delta w \tag{17}$$

$$\mu = \mu . \beta \, (\beta = 0.1) \tag{18}$$

Go back to Step 2

ELSE:

$$\mu = \frac{\mu}{\beta} \tag{19}$$

Go back to step 4

END IF

**Modification of the LM Method:** Considering performance index is $F(w) = e^T e$ using the Newton Method researchers have as:

$$W_{k+1} = W_k - A_k^{-1} . g_k \tag{20}$$

$$A_k = \nabla^2 F(w)\Big|_{w=w_k} \tag{21}$$

$$\mu = \frac{\mu}{\beta} \tag{22}$$

$$g_k = \nabla F(w)\Big|_{w=w_k} \tag{23}$$

$$[\nabla F(w)]_j = \frac{\partial F(w)}{\partial w_j} = 2\sum_{i=1}^{N} e_i(w) . \frac{\partial e_i(w)}{\partial w_j} \tag{24}$$

The gradient can write as:

$$\nabla F(x) = 2J^T e(w) \tag{25}$$

Where:

$$J(w)=\begin{pmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{21}}{\partial w_1} & \frac{\partial e_{21}}{\partial w_2} & \cdots & \frac{\partial e_{21}}{\partial w_N} \\ \vdots & & & \\ \frac{\partial e_{KP}}{\partial w_1} & \frac{\partial e_{21}}{\partial w_2} & \cdots & \frac{\partial e_{21}}{\partial w_N} \end{pmatrix} \quad (26)$$

J (w) is called the Jacobian matrix. Next, researchers want to find the Hessian matrix. The k, j elements of the Hessian matrix yields as:

$$[\nabla^2 F(w)]_{k,j} = \frac{\partial F(w)}{\partial w_k \partial w_j} \quad (27)$$

$$[\nabla^2 F(w)]_{k,j} = 2\sum_{i=1}^{N}\left\{\frac{\partial e_i(w)}{\partial w_k}\frac{\partial e_i(w)}{\partial w_j} + e_i(w)\frac{\partial^2 e_i(w)}{\partial w_k \partial w_j}\right\} \quad (28)$$

The Hessian matrix can then be expressed as follows:

$$\nabla^2 F(w) = 2J^T(W).\, J(W) + S(W) \quad (29)$$

Where:

$$S(W) = \sum_{i=1}^{N} e_i(w).\nabla^2 e_i(w) \quad (30)$$

If S(w) is small, Hessian matrix is approximated as:

$$\nabla^2 F(w) = 2J^T(W).\, J(W) \quad (31)$$

Using Eq. 12 and Eq. 4 Gauss-Newton Method is obtained as:

$$\begin{aligned} W_{k+1} &= W_k - [2J^T(w_k).\, J(w_k)]^{-1} 2J^T(w_k) e(w_k)] \\ &\cong W_k [J^T(w_k).\, J(w_k)]^{-1} J^T(w_k) e(w_k)] \end{aligned} \quad (32)$$

The advantage of Gauss-Newton is that it does not require calculation of second derivatives. There is a problem the Gauss-Newton Method is the matrix H = $J^TJ$ may not be invertible. This can be overcome by using the following modification. Hessian matrix can be written as:

$$G = H + \mu I \quad (33)$$

Suppose that the eigen values and eigenvectors of H are $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ and $\{z_1, z_2, \ldots, z_n\}$. Then:

$$\begin{aligned} Gz_i &= [H + \mu I]_{zi} \\ &= Hz_i + \mu z_i \\ &= \lambda_i z_i + \mu z_i \\ &= (\lambda_i + \mu) z_i \end{aligned} \quad (34)$$

Therefore, the eigenvectors of G are the same as the eigenvectors of H and the eigenvalues of G are $(\lambda_i+\mu)$. The matrix G is positive definite by increasing $\mu$ until $(\lambda_i+\mu)>0$ for all i therefore the matrix will be invertible. This leads to Levenberg-Marquardt algorithm:

$$w_{k+1} = w_k [J^T(w_k) J(w_k) + \mu I]^{-1} J^T(w_k) e(w_k)] \quad (35)$$

$$\Delta w_k = w_k [J^T(w_k) J(w_k) + \mu I]^{-1} J^T(w_k) e(w_k)] \quad (36)$$

As known, learning parameter, $\mu$ is illustrator of steps of actual output movement to desired output. In the standard LM Method, $\mu$ is a constant number. This study modifies LM Method using $\mu$ as:

$$\mu = 0.01 e^T e \quad (37)$$

where, e is a k×1 matrix therefore $e^Te$ is a 1×1, therefore $[J^TJ+\mu I]$ is invertible. Therefore if actual output is far than desired output or similarly, errors are large so, it converges to desired output with large steps. Likewise, when measurement of error is small then, actual output approaches to desired output with soft steps. Therefore error oscillation reduces greatly.

Thus, the most suitable disk access technique is obtained by using FNN. And moreover, the disk access patterns are also obtained.

## PROACTIVE DISK POWER MANAGEMENT

Thus, disk access patterns are obtained from the above mentioned technique. Then, the compiler can insert clear disk power management calls (instructions) in suitable places in the source code. The principle of these instructionsdiffers depending on the original disk capabilities (for instance, TPM versus DRPM). For TPM disks, spin_up() and spin_down() calls are used. The format of the spin_down() call is given as follows:

Spin_down ($d_i$)

where, $d_i$ is the disk ID. As a disk access pattern predictsboth idle and active times, this datacan be utilized topreactivate disks that have been spun down by a spin_down() call. In order toobtain the suitable point in the code to start spinning up the disk, spin-up time

(delay) of the disk is considered (that is the time taken for the disk to reach its full speed where it can perform read/write activity). Especially, the number of loop iterations before which the spin-up to be inserted is computed as:

$$Q_{su} = \left\lceil \frac{T_{su}}{s + T_m} \right\rceil \quad (38)$$

where, $Q_{su}$ denotes the preactivation distance (in terms of loop iterations), $T_{su}$ represents the expected spin-up time (in cycles), $T_m$ denotes the overhead incurred by a spin_up call and s denotes the number of cycles in the shortest path through the loop body. It is to be observed that $T_{su}$ is typically much larger than s. The format of the call that is used to preactivate (spin up) a disk is given as follows:

Spin_up ($d_i$)

where as before, $d_i$ is the disk ID. It is to be observed that if preactivation is not used, a TPM disk is automatically spun up when an access (request) comes but in this case, the associated spin-up delay is fully incurred. The purpose of disk preactivation is to eliminate this performance penalty.

## EXPERIMENTAL RESULTS

**Setup:** This disk power simulator similar to DiskSim (Bucy and Ganger, 2003) is determined by externally provided disk I/O request traces which are constructed by the trace generator. The disk energy consumption comprises of all the energy consumptions in both active and idle periods, considering all the states of disks during the whole execution. Also, the performance numbers include all conflicts in accessing the parallel disk system.

The performance of the proposed approach is compared with the approaches such as TPM, DRPM and BPN with modified LM (BPN with MLM) based Power Management System for each benchmark code in this experimental suite (Table 1).

The performance evaluation of the proposed approach is compared with the techniques like Traditional Power Management Systems, DRPM and BPN with modified LM based Power Management System. The energy saving comparison for various power loads are evaluated and compared.

Figure 4 shows the comparison of the proposed approach with the existing approaches such as TPM, DRPM and BPN with MLM. It is observed from the graph that for different work load applied to the power

Table 1: Benchmarks and their characteristics

| Benchmarks | Data size (GB) |
|---|---|
| 168.wupwise | 88.6 |
| 171.swim | 64.7 |
| 172.mgrid | 75.5 |
| 173.applu | 100.6 |

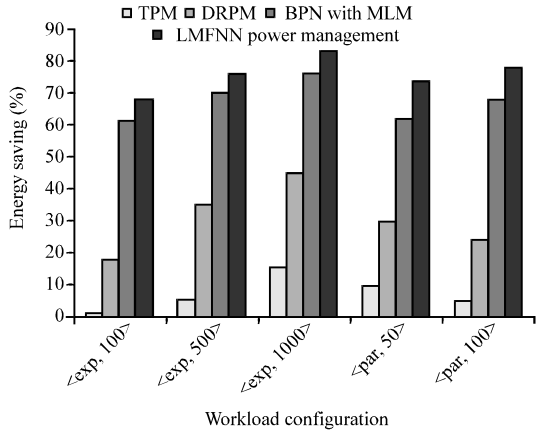


Fig. 4: Comparison of energy savings for exponential and Pareto Methods

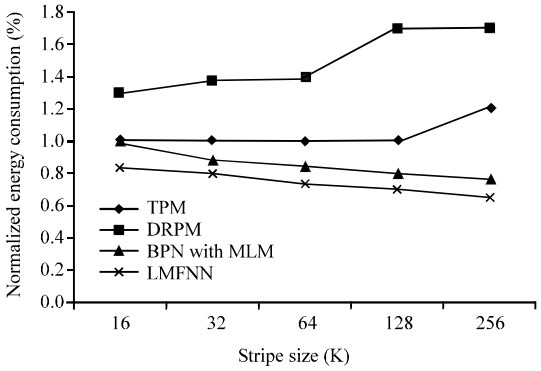


Fig. 5: Impact of stripe size on energy consumption

management techniques, the energy saving obtained by the proposed approach is very high when compared with TPM and DRPM.

The normalized energy consumptions with the different stripe sizes are shown in Fig. 5. It is obvious that the energy savings brought by the approaches such as TPM, DRPM and BPN with MLM increase as the stripe size increases. There is not much of idleness when the stripe size is very small (16 kbyte). Actually, the disk idleness in this case becomes so little that even code restructuring cannot take much gain of it. Additional disk requests can be serviced by a single stripe. When the stripe size is increased which means that the stripe-level data reuse also improves. It is observed that the proposed LMFNN disk management scheme generates the best savings of about 0.69% with 256 kbyte stripe size.

The normalized energy values under various stripe factors are shown in Fig. 6. Figure 6 clearly shows that when the number of disks is increased, disk idleness increases and as a result, the compiler approaches show a better behavior. When the number of disks is very high for instance 32, the disk idleness attains a very high level. From the Fig. 7, proposed LMFNN disk management scheme is observed to obtain less energy consumption for all the stripe factors.

Figure 7 indicates the impact of starting disk on the energy consumption. To perform this set of experiments, a random integer number is constructed (for each array in the mgrid benchmark) between 1 and 8 to choose the disk from which the array is striped. Figure 7 shows the results for five such experiments. It is observed from the graph that the proposed LMFNN disk management scheme consumes less energy when compared with the approaches like TPM, DRPM and BPN with MLM power management approaches.

Figure 5-7 shows the impact of the stripe size, stripe factor and starting disk on energy consumption. It is

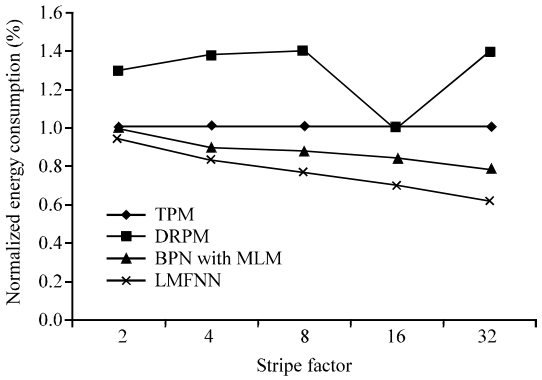


Fig. 6: Impact of stripe factor on energy consumption

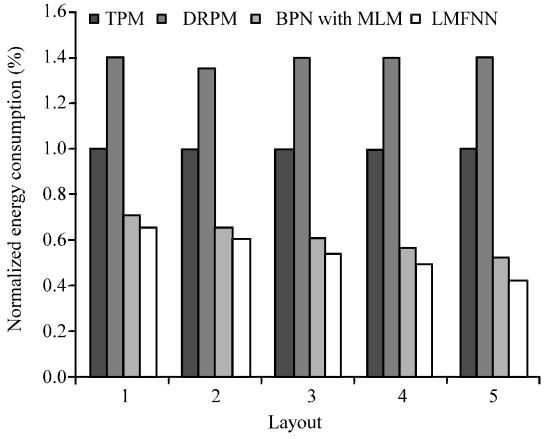


Fig. 7: Impact of starting disk on energy consumption

observed that the proposed LMFNN disk management approach has better energy consumption in all the three cases. Thus, the proposed LMFNN disk management approach out performs the other three approaches in terms of energy consumption.

## CONCLUSION

Disk power management has been considered as one of the key ways of saving energy in disks that process data-intensive applications. The earlier researches mainly focused approaches such as spinning-down a disk or rotating disks at a lower speed to save disk energy. A majority of these approaches do not focus on idle periods and the selection of appropriate disk access scheduling algorithm.

This study presents a profile-driven technique to diminish the energy consumption. This technique uses an efficient neural network approach for selecting suitable disk access techniques. Fuzzy Neural Network (FNN) approach is used in this approach for choosing the appropriate disk access techniques. FNN is trained using the modified Levenberg-Marquardt learning algorithm. This novel approach is called LMFNN as FNN is trained using Modified Levenberg-Marquardt. The experiment results reveal with this approach greatly reduces energy consumption of original applications significantly. The best energy savings are obtained by LMFNN scheme that selects a suitable disk access technique. The proposed LMFNN approach outperforms the disk power management techniques such as TPM, DRPM and BPN with modified LM.

## REFERENCES

Bohrer, P., E. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell and R. Rajamony, 2002. The Case for Power Management in Web Servers. In: Power-Aware Computing, Graybill, R. and R. Melhem (Eds.). Kluwer Academic Publisher, Norwell, New York, USA., ISBN: 9780306467868, Pages: 376.

Bucy, J.S. and G.R. Ganger, 2003. The DiskSim Simulation Environment Version 3.0 Reference Manual. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA., USA., Pages: 61.

Chase, J. and R. Doyle, 2001. Balance of power: Energy management for server clusters. Proceedings of the 8th Workshop on Hot Topics in Operating Systems, May 20-22, 2001, Elmau, Germany, pp: 165-165.

Chase, J., D. Anderson, P. Thackar, A. Vahdat and R. Boyle, 2001. Managing energy and server resources in hosting centers. Proceedings of the 18th ACM Symposium on Operating Systems Principles, October 21-24, 2001, Banff, Canada, pp: 103-116.

Douglis, F., P. Krishnan and B. Bershad, 1995. Adaptive disk spin-down policies for mobile computers. Proceedings of the 2nd Symposium Mobile and Location-Independent Computing, April 10-11, 1995, Ann Arbor, MI., USA., pp: 121-137.

Douglis, F., P. Krishnan and B. Marsh, 1994. Thwarting the power- hungry disk. Proceedings of the Conference on Winter USENIX, January 17-21, 1994, San Francisco, California, pp: 23-23.

Elnozahy, E.N.M., M. Kistler and R. Rajamony, 2002. Energy-efficient server clusters. Proceegings of the 2nd International Workshop Power Aware Computing Systems, February 2, 2002, Cambridge, MA, USA.

Elnozahy, M., M. Kistler and R. Rajamony, 2003. Energy conservation policies for web servers. Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems, March 26-28, 2003, Seattle, WA., USA., pp: 8-8.

Garrett, M., 2007. Powering down. ACM Queue, 5: 16-21.

Go, J. and M. Song, 2008. Adaptive disk power management for portable media players. IEEE Trans. Cons. Electron., 54: 1755-1760.

Gurumurthi, S., A. Sivasubramaniam, M. Kandemir and H. Franke, 2003. Reducing disk power consumption in servers with DRPM. Computer, 36: 59-66.

Irani, S., G. Singh, S.K. Shukla and R.K. Gupta, 2005. An overview of the competitive and adversarial approaches to designing dynamic power management strategies. IEEE Trans. VLSI Syst., 13: 1349-1361.

Li, K., R. Kumpf, P. Horton and T. Anderson, 1994. A quantitative analysis of disk drive power management in portable computers. Proceedings of the Conference on USENIX Winter, January 17-21, 1994, San Francisco, California, pp: 279-292.

Nauck, D., F. Klawonn and R. Kruse, 1997. Foundations of Neuro-Fuzzy Systems. John Wiley, New York, ISBN: 9780471971511, Pages: 305.

Ross, R.B., P.H. Carns, W.B. Ligon III and R. Latham, 2002. Using the parallel virtual file system. ftp://mirror.anl.gov/pub/pub/pvfs/user-guide.pdf.

Son, S.W., G. Chen, O. Ozturk, M. Kandemir and A. Choudhary, 2007. Compiler-directed energy optimization for parallel-disk-based systems. IEEE Trans. Parallel Distribut. Syst., 18: 1241-1257.

Suratgar, A.A., M.B. Tavakoli and A. Hoseinabadi, 2005. Modified levenberg-marquardt method for neural networks training. World Academy of Science, Engineering and Technology.

Wang, W.Y., T.T. Lee, C.L. Liu and C.H. Wang, 1997. Function approximation using fuzzy neural networks with robust learning algorithm. IEEE Trans. Syst., Man Cybern. Part B, 27: 740-747.

Weddle, C., M. Oldham, J. Qian, A.I.A. Wang, P. Reiher and G. Kuenning, 2007. PARAID: A gear-shifting power-aware RAID. Trans. Storage, 3: 1-16.

Xie, T. and H. Wang, 2008. MICRO: A multilevel caching-based reconstruction optimization for mobile storage systems. IEEE Trans. Comput., 57: 1386-1398.

Yin, S., M.I. Alghamdi, X. Ruan, M. Nijim and A. Tamilarasan *et al*., 2010. Improving energy efficiency and security for disk systems. Proceedings of the 12th IEEE International Conference on High Performance Computing and Communications, September 1-3, 2010, Melbourne VIC, Australia, pp: 442-449.