Analysis of Process Scheduling Using Neural Network in Operating System

Harshit Agarwal and Gaurav Jariwala

Abstract Process scheduling plays a vital role in multitasking for any operating system. There are many factors involved during process scheduling like priorities, free memory, user demand and processor which if not handled properly can be very complex and time consuming. Neural network has adaptive nature which can be used to handle the complex part easily. The main aim of this paper is to review different types of scheduling algorithms working on the principle of neural network and offer constructive criticism to improve their efficiency.

Keywords Backpropagation · Bayesian system · Decision tree learning · Genetic algorithm · Neural network · Process scheduling · Rule-based system

1 Introduction

Since different users use their computer systems in a different ways, neural network in process scheduling will help to optimize the work. Neural network has the ability to learn and solve complex problems while finding the most optimal solutions.

Efficient process scheduling and context switching is the most important step of multitasking. This could be the stepping stone for the AI-based programs and neural network. Their applications are endless. Knowing what job is to be scheduled by studying the user behavior and doing that in an optimal way is a crucial step. It is necessary for the operating system to understand which processes are important and which are not. Researchers have focused on developing algorithms for learning the user behavior on system by recording the application's use and generating a predictive loop. The drawback of this system is that it requires a lot of time to process and is unpredictable.

H. Agarwal (✉) · G. Jariwala
Sarvajanik College of Engineering and Technology, Surat, India
e-mail: 9arshit@gmail.com

G. Jariwala
e-mail: gjariwala9@gmail.com

G. Ranganathan et al. (eds.), *Inventive Communication and Computational Technologies*, Lecture Notes in Networks and Systems 89,
https://doi.org/10.1007/978-981-15-0146-3_97

For process scheduling, there are many algorithms used. All techniques have some merits and demerits to them which we will discuss in this paper. Methods that will be discussed are job shop scheduling, preemptive scheduling, multithreading, genetic algorithm, decision tree learning, Bayesian system, rule-based system and neural network.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes process scheduling based algorithm in detail. Section 4 describes AI techniques based algorithm in detail and existing research. Section 5 shows comparison table between different algorithms. Finally, Sect. 6 draws conclusions and discusses future work.

2 Related Work

Ajmani and Sethi [1] proposed a fuzzy logic-based CPU scheduling algorithm to overcome drawbacks of conventional algorithms for efficient utilization of CPU. This paper has compared the proposed fuzzy CPU scheduling algorithm (PFCS) with priority algorithm; they found that the priority algorithm has more average waiting time and average turnaround time than PFCS. Rehaiem et al. [2] presented an approach for real-time scheduling of reconfigurable embedded systems using neural networks. Sharma et al. [3] have given a optimize credit based scheduling algorithm based on load balancing to improve the performance in cloud computing.

3 Process Scheduling Algorithms

Process scheduling is a procedure that arranges the order and time period of different processes during which they can use the CPU. If not for processing algorithms, a lot of CPU cycles would have been wasted ideally when the process is waiting for I/O or memory. Algorithms in use make our system efficient and fair.

3.1 Job Shop Scheduling

Job shop scheduling is also called linear workshop process scheduling [4]. It can be considered the simplest form of process scheduling. There are different job shops presently capable of handling the jobs, see [5]. They accept the contract and complete the work. In this case, the processes are jobs that are to be finished. The problem is to schedule these jobs in such a way that the jobs have to wait for minimum amount of time before they are processed, and the entire system works efficiently.

For example, before data analysis process is done, it is necessary to obtain data. Hence, I/O process cannot be scheduled after analysis process. Job shop scheduling requires details for the process, time of completion beforehand. We cannot add a new process at runtime which is unsuited for OS. Hence, this method is too static for process scheduling in OS [4].

3.2 Preemptive Scheduling

A scheduling discipline is categorized as preemptive if once the process has been given the CPU, it can be taken away. In preemptive scheduling, a process is given a quantum time during which it is allowed to run. After the quantum time is expired for a process, the CPU control is returned to kernel. This is done using a clock interrupts. The kernel then saves the state, and then, it decides which process will now be given control of the CPU.

If the process has completed before its quantum time expires, the process can forfeit its ownership of the CPU voluntarily to the kernel. This system is unlike non-preemptive scheduling where only when the process itself gives up the CPU kernel cannot access it. If the process is not completed, it goes back to the ready queue and waits for its turn again. The priority of the process in that queue is then depended on the algorithm the processor is running on.

Algorithm that is based on preemptive scheduling is Round Robin. The shortest job first (SJF) and priority scheduling under specific circumstances can be classified as preemptive scheduling. This kind of algorithm is better suited for background process than user process as shutting down the user process against their need will defeat the purpose of user-friendly OS.

3.3 Multithreading

Threads are called lightweight processes (LWP). Threads within a process share address space and other resources. The term multithreading is used to describe multiple threads in same process. When a multithreaded process is running on single-CPU system, the threads run one at a time just like processes in multiprogramming on single-CPU system. The CPU switches between threads so quickly that it gives an illusion of threads running in parallel. True parallelism of running threads can be achieved using multithreaded processors. The minimum requirement for multithreaded processor is the ability to pursue two or more threads in parallel within the processor pipeline—i.e., it must provide two or more independent program counters—and a mechanism that triggers a thread switch [6].

The best way to understand the usefulness of threads is through example. Word processors use threads to increase productivity and improve interactivity. When a user types a character at the keyboard, one thread is used to read that character. Word processor can execute other threads between keyboard interrupts. Like, storing the document in the disk to prevent data loss and checking for spelling mistakes. Each feature is implemented using different threads so even if a thread is blocked due to an I/O operation (storing the document), the word processor can response to keyboard interrupt.

Researchers have attempted to combine advantages of both user threads and kernel threads called scheduler activation (M:N model) [7]. The goals of the scheduler activation work are to mimic the functionality of kernel threads, but with the better performance and greater flexibility usually associated with threads packages implemented in user space [8]. A neural network scheduler can be designed for threads and process in similar way so in this paper we will concentrate on process scheduling rather than thread scheduling.

4 AI Technique Based Algorithms

Algorithms discussed above may not always work. They might be efficient, but they are too static. In order to create a perfect neural system-based operating system, it is necessary to have components that learn from the user over a period of time. AI techniques can be used for process scheduling. It learns user behavior and reprograms the system to predict user's need.

4.1 Genetic Algorithm

Genetic algorithm was inspired by the Darwin theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation [9]. The solutions are selected on the bases of their fitness and then promoted further. They are grouped in pairs and combined using genetic operator like mutation and crossover. The child solution created from the parent using these methods then acts like a parent in further iterations.

Figure 1 shows the flowchart of genetic algorithm where initial set of individual solutions are called population. An individual is characterized by a set of parameters (variables) known as Genes. Genes are joined into a string to form a chromosome (solution) as presented in Fig. 2. This process is repeated and evaluated until there is a solution in the set that exceeds a minimum fitness. Crossover operator is applied to the selected chromosomes during which better string is created, while the information in the parent string is optimally preserved. The crossover is done in hope of creating a better string, but the result obtain may or may not be the desired one depending on the use. After crossover, the strings are subjected to mutation [10]. Mutation causes bit-wise reversal. The bit's position is chosen with probability of Pm. The process of mutation is done to maintain diversity in population since selection process may

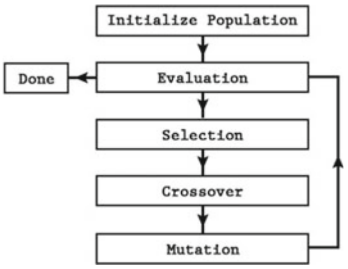

Fig. 1 Flowchart for genetic algorithm. *Source* [9]

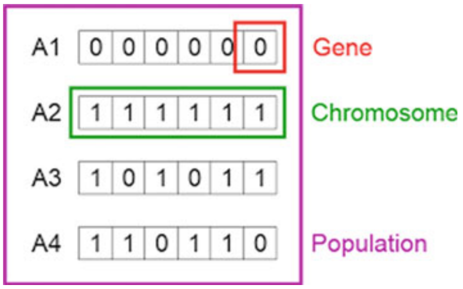

Fig. 2 Elements of genetic algorithm. *Source* [9]

promote same breed of strings which does not optimize search algorithm and helps to retrieve information that may be lost during crossover operator.

Limited research projects have been on implementing genetic algorithm-based schedulers [11, 12]. The process scheduler based on genetic algorithm shows similar results like shortest job first algorithm [13] in certain cases. But shortest job first algorithm cannot be implemented at the level of short-term CPU scheduling. There is no way to know the length of the next CPU burst. Here, performance can be improved by using genetic algorithm. The other studies involved did not use algorithm in the scheduler but rather in the application. The study showed that genetic algorithm had same result as the priority scheduler and the algorithm was time consuming. But the time consumption was mostly due to hardware. With even more simplified algorithm, this problem can be easily solved. The genetic algorithm evolving technique does provide more flexible mechanism than FIFO [8] scheduling and adapts itself to changing environment.

4.2 *Decision Tree Learning*

As the name says it is a tree-like model of decision. A decision tree contains non-leaf (internal nodes) nodes which represent input features and the leaf nodes are the possible solutions. The splitting at non-leaf nodes is done according to certain discrete function of input attribute values (as in our case) is called classification tree.

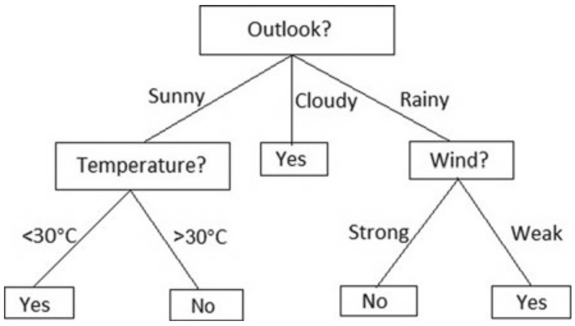

Fig. 3 Decision tree to play football

As shown in Fig. 3, an example decides whether to play football or not (in this case Yes or No). At the root node, an instance is classified by testing the attribute specified by this node, and then moving down tree branch according to the value of the attribute. This process is repeated at each node until leaf node is reached.

This algorithm is recursive in nature as same approach is used at each node to split the groups formed. As this algorithm never backtracks to reconsider earlier choices, it is also called greedy algorithm. The algorithm is very simple and easy to implement. The main disadvantage of using this algorithm is its very sensitive to noise meaning small variations in data might result in completely different tree being generated. Furthermore, due to the greedy characteristic of decision tree it cannot guarantee the globally optimal solution. See Doh et al. [14] for decision tree base scheduling for flexible job shops.

4.3 Bayesian System

Bayesian networks also known as belief networks are a graphical model that shows probabilistic relationships among set of variables [15]. It is a graph called a directed acyclic graph (DAG) which contains no directed cycles. Bayesian network requires probability distribution $P(Y|X)$ for each node Y. If node Y has no parents, then it is just $P(Y)$. $P(X|Y)$ is known as conditional probability. For example, to calculate the probability of computer failure, see [16].

Bayesian network can be used, even in the case of missing data, to learn the causal relationships and gain an understanding of the various problem domains and to predict future events [17]. As Bayesian network is a DAG, it cannot be used when the graphical model contains cycle. This approach is computationally expensive. In our case, there is no real structure that can easily be imposed on the data of the system.

4.4 Rule-Based System

Rule-based system uses a set of IF-THEN rules for classification. If the condition mention on the "if" side is satisfied, then the rule is said to be triggered, and the action corresponding in the "then" part executes. This system is the most accurate in comparison with other systems; this is because other system works on probability functions that may not give us always the desired or the most appropriate result. The rule-based system is only as accurate as the rules programmed in the system, but this turn creates problems such as, what if more than one rule is fired or none of them does. This issue can be solved with help of size ordering, rule ordering or class-based ordering [18], the programmer needs to cover the entire basis which makes this impractical for complex systems.

An example of a rule-based system is the domain-specific expert system that uses rules to make deductions or choices. For example, an expert system might help a doctor choose the correct diagnosis based on a cluster of symptoms or select tactical moves to play a game [19].

However, by creating a hybrid system using decision tree and rule-based system can have vast applications with the least amount of drawbacks. We can extract rule from the decision tree system where rules are created from the path of root to leaf node. Using this system helps to convert "IF-THEN" static system to dynamic system which increases its application arena.

4.5 Neural Network

Neural networks or artificial neural networks (ANNs) are analogy from biological neural networks in which neurons are used to transmit signals, while in ANNs nodes are used for the same. These nodes form layers, and each node is connected with every node of next layer with some weights.

There are three types of layers in ANNs, input layer, hidden layers and output layer. Any number of hidden layers can be there in a neural network. The input layer is used to feed the data to the neural network, while the output layer gives the result of the provided input. Each node has its activation upon which the output is decided. The activation of each node is calculated by summing the products of activation of every incoming node with their associated weight and adding it with the bias. Then, the resultant value is passed to a function called sigmoid to get the activation value of the node between 0 and 1.

Backpropagation algorithm is the most commonly used technique in ANNs. In this, each weight is adjusted from the output layer to input layer so that the network error can be reduced. There are other techniques that are used in ANNs which are out of scope for this paper.

Neural network is the most fitting technique to be used in process scheduling as it can adapt to different situations. Since the data of the process are continuous in

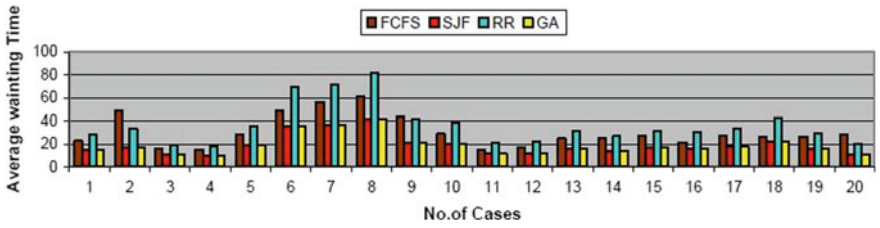

Fig. 4 Comparision of scheduling algorithm. *Source* [20]

nature so are input and output. The demerit of the neural networks is that it is hard to interpret the model as they are trained.

Many studies were done in which stimulations were run for using different algorithms on kernel level. Results obtained from these studies showed that neural network-based algorithms were faster than conventional system used currently.

As shown in Fig. 4, genetic algorithm had the least waiting time among the different algorithm used for job scheduling. Its use also provides more flexibility than other algorithms [20]. Results obtained from different studies [4, 14] were also in favor of neural network-based kernel. During initial phase of the use, system was slow which caused the running applications' performance in percentage of FPS drop and loading speed of the multimedia. But with time, the system was trained and performance showed significant improvement in waiting time for the jobs and percentage of FPS drop. The initial stage problem was reduced with help of pre-training but was not eliminated. Use of decision tree-based approach reduced flow time and increased tardiness. Stable performance was achieved with improvement in the method.

5 Comparison of Algorithms

See Table 1.

Table 1 Comparison between different scheduling algorithms

Name	Method	Merits	Demerits	Improvements
Job shop scheduling	– Queues the job in linear manner	– Simplest form of scheduling – Compatible with all the systems	– Creates a static system – Limits multitasking	– Using multilayer function for scheduling to increase productivity
Preemptive scheduling	– Assigns quantum time to process during which process takes the CPU. After which it is put back into queue	– Single process cannot monopolize the CPU – Priority-based quantum time can be assigned	– It is not suitable for user-based background processes	– User-based processes should have programming to override quantum time
Multithreading	– More than one thread work concurrently within a process	– Improves system's performance – Better use of CPU resources	– Debugging becomes complex – Increase occurrence of deadlock	– Banker's algorithm can be used to avoid deadlock but for using this algorithm maximum numbers of resources needed by each process should be known
Genetic algorithm	– Selection, crossover and mutation functions are used to select the most suitable process	– It overcomes the limitations of shortest job first algorithm	– Crossover may cause unwanted results to reoccur – Results are based on probability function's accuracy	– Creation of simpler program to avoid complexity – Using adaptive crossover function
Decision tree learning	– Non-leaf nodes are split, and the output is generated by the series of decisions	– It is easy to implement – Nonlinear parameters can be handled	– Overfitting is caused due to noise in the dataset – Does not produce globally optimal solution	– Using pruning size of the decision tree can be reduced to tackle overfitting

(continued)

Table 1 (continued)

Name	Method	Merits	Demerits	Improvements
Bayesian system	– It works on the bases of probabilities	– Can be used even in the case of missing data	– It cannot be used when the graph contains cycle – It is computationally expensive	– Its accuracy is very good on small datasets but it may asymptote to a high error rate, making it less useful as a classifier for very large databases [15]
Rule-based system	– Uses set of IF-THEN rules to trigger the rule	– Most accurate system – Can be integrated with any system	– All the rules need to be mentioned manually	– Hybrid system of rule based and decision tree can have more applications
Neural network	– It works on layered structure and output is generated using the activation	– Creates a user-specific system – Adapt to different situations	– Specialized hardware needs to be used to process data – Hard to interpret the structure of neural network	– Creating better compression system for fast computing

6 Conclusion and Future Work

In this paper, process scheduling algorithms and AI-based techniques were analyzed for their implementation in operating system. While in comparison study of some of these algorithms gave similar results, the more advance approach of neural network-based algorithms had the advantage of being more flexible. With recent increase in research in field of artificial intelligence and machine learning, these systems are more likely to use in the future. Job shop scheduling is obsolete and static to handle multitasking used in the current system, while Bayesian system implementation proves to be only theoretical in our case. Though initial uses of neural network systems were slow in comparison with regular process scheduling algorithms, with respect to time the training program and prediction loop provided more efficiency. Until training program reaches to optimal level use of generic algorithms will be more beneficial and then switching of algorithms could take place.

References

1. Ajmani P, Sethi M (2013) Proposed fuzzy CPU scheduling algorithm (PFCS) for real time operating systems. BIJIT - BVICAM's Int J Inf Technol 5(2) (New Delhi, India)
2. Rehaiem G, Gharsellaoui H, Ahmed SB (2016) A neural networks based approach for the real-time scheduling of reconfigurable embedded systems with minimization of power consumption. In: IEEE/ACIS 15th international conference on computer and information science (ICIS), Okayama, 2015, pp 1–6
3. Sharma A, Gupta AK, Goyal D (2018) An optimized task scheduling in cloud computing using priority. In: Proceedings 3rd international conference on internet of things and connected technologies (ICIoTCT), Jaipur, India, 26–27 Mar 2018
4. Bex P (2008) Implementing a process scheduler using neural network technology. Radbound University Nijmegen
5. Yoo BY (1977) Methods and techniques used for job shop scheduling. University of Central Florida
6. Ungerer T, Robic B, Silc J (2002) Multithreaded processors. Comput J 45(3):321–348
7. Anderson TE, Berchad BN, Lazowska ED, Levy HM (1992) Scheduler activations: effective kernel support for the user-level management of parallelism. ACM Trans Comput Syst 10(1):53–79
8. Tanenbuam AS (2009) Modern operating system (3rd edn). Upper Saddle River, NJ, 07458
9. Mallawaarachchi V, Introduction to genetic algorithms. https://towardsdatascience.com
10. Pai GAV, Rajasekaran S (2011) Neural networks, fuzzy logic and genetic algorithms - synthesis and applications, India, July 2011
11. Sharma M, Sindhwani P, Maheswari V (2013) Genetic algorithm optimal approach for scheduling processes in operating system. Int J Eng Res Technol 2. ISSN: 2278-0181
12. Elrad T, Jinlong L, Cork DJ (1998) Evolutionary computation for scheduling controls in concurrent object-oriented systems. Int J Comput Their Appl 5(3):11–20
13. Arpaci-Dusseau RH, Arpaci-Dusseau AC (2014) Operating systems: three easy pieces. Arpaci-Dusseau Books
14. Doh HH, Yu JM, Kwon YJ, Shin JH, Kim HW, Nam SH, Lee DH (2014) Decision tree based scheduling for flexible job shops with multiple process plan. World Acad Sci Eng Technol 8(3):621–627

15. Kohavi R, Becker B, Sommerfield D (2001) Improving Simple Bayes. In: Proceedings of the European conference on machine learning
16. Horný M (2014) Bayesian networks. Technical report no. 5, Department of Health Policy & Management, Boston University School of Public Health, Apr 2014
17. Ben-Gal I (2007) Bayesian networks. In: Ruggeri F, Faltin F, Kenett R (eds) Encyclopedia of statistics in quality & reliability. Wiley, London
18. Jiawei H, Kamber M, Jian P (2012) Data mining concepts and techniques (3rd edn)
19. Gupta A, Newell A, Wedig R, Forgy C (2005) Parallel algorithms and architectures for rule-based systems. IEEE Computer Society Press, Los Alamitos, CA. Tokyo, Japan
20. Dr. Kumar R, Er. Kumar R, Er. Kaushik A, Er. Gill S (2010) Genetic algorithm approach to operating system process scheduling problem. Int J Eng Sci Technol 2:4247–4252