Improvement in Field Scheduling

Aarushi Singhal

University of Cincinnati

Table of Contents

1. Introduction

Nowadays, we have observed our gadgets are going more and more compact and small. Now the mobile phones can do large amount of work which was earlier possible to be done only with the help of computers. With advancement in field of technology we have to make amendments in ways of our techniques to achieve optimized results. Optimization is the process of not only achieving efficiency to find out solution but also make process cost efficient. One such area where optimization is required is, organizing how the task will run in operating system. When user works on computer he usually open so many programs at same time such as listening to music, browsing on internet, chatting on Facebook, may be watching videos etc. so it becomes cumbersome for processor to organize all the tasks in efficient and optimized manner. Organizing all the processes in systematic manner and maintaining the priorities of all their execution is known as "Scheduling." There are different kind of scheduling algorithms which in different ways helps in execution of the processes. There are mainly 4 types of scheduling algorithms which are broadly used for scheduling the processes are "first come first served", "shortest job first", "priority scheduling", "round robin scheduling". These algorithms organize the processes in their own way and execute them, but these algorithms have certain limitations which affect their quality.

Before describing scheduling, there are two types in which process can execute

1. Preemptive – It is the method which moves a process from running to ready without the process requesting it i.e. it will see whether the process execution takes more time to complete and if it does then it can stop process in between to give other process a chance to execute.
2. Non-Preemptive – the system will execute the process till it gets over the process which is currently in execution (Silberschatz et al., 2005).

Explanations:

1. FCFS

First come, first served (FCFS) is an operating system process scheduling algorithm and a network routing management mechanism that automatically executes queued requests and processes by the order of their arrival. With first come, first served, what comes first is handled first ,the next request in line will be executed once the one before it is complete. This leads to problem of starvation which is also called "Convoy Effect."

2. Shortest Job First

Shortest job first (SJF is a scheduling policy that selects for execution the waiting process with the smallest execution time .

In this even if the process in the queue have high priority but more burst time will not be executed and will starve.

3. Priority scheduling

Priority Scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned a priority. Process with the highest priority is to be executed first and so on.

The limitation of this scheduling is that the process if having higher priority will execute but if it has high burst time then other processes with less burst time and low priority will suffer. It will lead to starvation.

4. Round Robin scheduling

Round-robin (RR) scheduling is one of the algorithms employed by process and network schedulers in computing. As the term is generally used, time slices (also known as time quanta) are assigned to each process in equal portions and in circular order, handling all processes without priority. The limitation of round robin is that it does not consider priority of the processes and execute processes in form of first come first served manner except the factor of time quanta which in the case of process having high burst time may lead to starvation ("Comparison of different scheduling algorithms in OS", 2013).

Scheduling algorithm is required to carry out the required number of processes in such a manner that their execution should bring out high throughput and focuses on priority of message also so that urgency can be answered. The above description shows that even if these algorithms are different from each other and have different ways to perform scheduling still they rest on same kind of demerit that is of starvation. There is a need to develop such kind of algorithm that focuses on bringing such kind of tactic that schedule the process as per their priority but at the same time no starvation should occur.

One of the efficient algorithm in artificial intelligence and machine learning which is used for optimizing different results and enhancement of output quality is "artificial bee colony" (Karaboga, 2005). It is one of the most searched upon domain nowadays and is depended upon bee behavior of foraging and selection of food sources. Algorithms that are based on animal behavior for search of food is called "Swarm Intelligence" (Goldberg, 1998). Artificial Bee colony or ABC algorithm is one of such algorithm which concentrate on bee techniques and methodologies for searching and remembering their food sources and determining best food source for further processing. This ABC algorithm has now been used in different areas and has derived wide range of applications.

Basically, it is being implemented in GPS (Yetkin and Berber, 2014), wireless routers (Okdem et.al, 2014), building reservoirs and dams (Hossain and Shafie, 2013) etc. one such new application where it has not yet properly implemented is scheduling.

This research proposal concentrates on proposing a modified version of algorithm which focuses on existing scheduling algorithm's demerits and tries to overcome their limitations. The study focusses on involving concepts of all different kinds of scheduling algorithm and building it in a new type of algorithm that could target on demerits of existing approaches.

2. Research Significance

This section shows how this research is significant in field of computer science and information technology. It focuses on showing academic and practical significance of this study.

2 .1 **Academic Significance**

This research proposal focuses on building an algorithm that concentrates on improving process of "scheduling" in operating system. The proposed algorithm not only focuses on targeting the existing algorithm's demerit but also works to improve throughput and execution. If all the process in the queue are being executed in optimized manner then the tasks will be performed in an efficient way with no loss of any process and enhances quality of operating system in terms of managing the resource allocation.

The research concentrates on dynamically assisting the bundle of processes in queue and tries to find out optimized solution out of the rest (implementing concept of artificial bee colony). It compares the performance of proposed algorithm with existing approach thus adds value.

2.**2 Practical significance**

This research proposal focuses on implementation of concepts of different scheduling along with the concepts of artificial bee colony to maximize the optimality (enhancing the efficiency in obtaining better solution of a problem accurately and in less cost). This proposal focusses on management of resources by effectively organizing process execution, by optimizing time complexity, waiting time and space complexity with dynamically arranging the incoming processes in the queue to generate solution.This concept can be implemented in different areas such as routing in network. The messages that are being transmitted from one computer to another, how to find out which message should be transmitted first. How to schedule the messages coming to router according to their urgency to be transmitted into network, there the concept of this algorithm can be implemented.

It can be implemented in ATM transactions. It can just make working efficient by allocating each process with proper resources.

It can be implemented in devices such as Database Management System where different tasks are needs to be managed simultaneously. It can efficiently distribute resources and help in proper execution of each process.

It can be implemented in devices such as Fitbit where different tasks are running at same time.

3. Research Questions:

1. Is the proposed algorithm better than the scheduling algorithm which are currently being used in operating system?
2. Do the proposed algorithm targets the condition of starvation?
3. Whether the scope of proposed algorithm is limited to scheduling or it can be used in other areas too?

4. Brief Literature Review

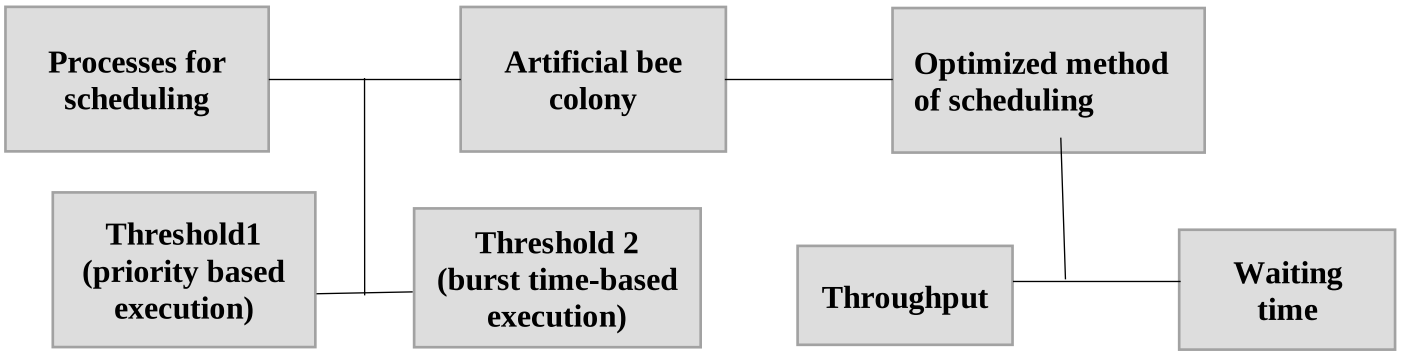
Through literature review it could be found that the different scheduling algorithms such as FCFS, shortest job first, priority scheduling, round robin scheduling, it can be concluded that different strategies are being employed but actually the condition of starvation is predominant. Whether it is convoy effect in FCFS or starvation in priority or cascading completion and unequal length of task problem in round robin all corresponds to proposing such an algorithm which define such strategy that could help in organizing processes in such a way that no starvation could occur. (Silberschatz et al., 2005). Artificial Bee Colony is one of the most flexible, robust, high performance, efficient and optimized searching algorithm for finding best solutions. It was initially designed to find solutions for numerical problems either constrained or unconstrained (Karaboga, 2005) and with more advancement it is being used in the areas such construction of dams (Hossain and Shafie, 2013) where ABC algorithm optimized the calculation of medium and low inflow category that can answer how much water need to be released for a month by observing the reservoir level, clustering in wireless network (Okdem et.al, 2012) where it optimized cluster formation to increase network life by making feature selection in optimized manner which was earlier not possible, making GPS robust (Yetkin and Berber, 2014) where the 2 parameters LTS (Least Trimmed Square) and LMS(Least Mean Square) calculation were optimized to reduce overall residual error etc. So, with all these researches it can be inferred that is one of the most adaptive and dynamic algorithm which optimizes different tasks. So, the concept of artificial bee colony is used in the research to optimize the process of scheduling After reviewing research paper related to priority based CPU scheduling algorithm in real time (Rajput and Gupta, 2012) it can be inferred that round robin and priority scheduling algorithm when they are merged together, the merged algorithm functioned better. So, this idea forms the basis of the proposed algorithm of this research proposal where two of the major algorithms such as priority and shortest job first is taken into consideration along with concepts of artificial bee colony algorithm to make it optimized. Other research paper focusing on method to increase efficiency of round robin scheduling by adjusting time quanta as per burst time (Noon et al., 2011). This concept is used to introduce threshold limit in the proposed algorithm in the research proposal.

5. Theoretical Basis and A Tentative Research Model

The following research proposal is based on programming approach in any high-level language like java, C/C++, Python etc. This model is based on the approach taken from, a priority based round robin CPU scheduling algorithm for real time systems (Rajput and Gupta, 2012) and time quanta being adjusted on the basis of burst time (dynamic round robin scheduling) (Noon et al., 2011). Applications of artificial bee colony in optimizing wireless sensor network by formation of clusters (Okdem et.al., 2011), estimating GPS by optimizing two parameters (Least Trimmed Square and Least Mean Square) by reducing the residual error in the computation of Euclidean distance between two points (Yetkin & Berber, 2012) and also in optimizing constraints of beam weighting (Karaboga, 2005) showed that artificial bee colony algorithm can be implemented in optimizing proposed research.

On the basis of literature review this research proposal proposes a tentative study model which focuses on actual working of the proposed algorithm. The concepts of artificial bee colony are used as initial seed of this algorithm and two thresholds are set for switching the execution of the processes in the queue. First threshold works on priority of processes for abut 35 units and after 35 units, threshold 2 works on burst time of processes for about 15 units, this alternation helps in controlling the condition of starvation by reducing waiting time in the queue and enhances performance by increasing overall throughput.

Tentative Research model


Processes for scheduling
Artificial bee colony
Optimized method of scheduling
Threshold1 (priority based execution)
Threshold 2 (burst time-based execution)
Throughput
Waiting time

6. Research Design

This section discusses how the proposed algorithm works and how it actually is better than the other algorithms used for scheduling. Research design comprises of experiment section validity and practical implementation.

6.1 Experiment

This section comprises of how the algorithm actually works and how it is efficient than other scheduling algorithms.

6.11 Algorithm

a. Compare the priorities of the processes in the pool (queue).

b. Size of pool = number of employed bees

c. Serve the process having highest priority.

d. Decrement wait time of the processes in h pool by the burst time of the process being executed.

e. If a new process having the priority greater than the priority of current process arrived, preempt the currently executing process and go to step 3.

f. Repeat steps 1-5 for threshold 1 (it is the value after which processes in the queue will be scheduled and executed according to their burst time, burst time will be seen as priority) number of times.

g. Wait time = new priority.

h. Repeat steps 1 to 5 for threshold 2 (randomized value of clock after which the process's actual priority becomes the priority for scheduling) number of times.

i. Revert priority to its original value.

j. Go to step 1.

6.12 Comparison among different scheduling algorithms

Parameters

On the basis of two measures the effectiveness of the proposed algorithm with other scheduling algorithms is being compared are as follows:

a. **Average Wait time** which is the total time for the process remains in the queue per unit time.

b. **Throughput** is the actual output computed in the end.

In this section, the proposed algorithm is being compared with all other scheduling algorithms with the help of example.

6.121 FCFS

Let us assume the queue size in which processes are arranged is 5 at a time and randomly the processes are coming with different burst time and priorities.

Processes	Priorities	Burst Time
P1	1	20
P2	4	12
P3	5	8
P4	10	10
P5	14	15
P6	2	2
P7	3	4
P8	1	17
P9	9	9
P10	6	8

As the algorithm is based on first come served basis the process which will come first in queue will execute first irrespective of the burst time it is going to take in.

Processes	Priorities	Burst Time	Wait Time
P1	1	20	
P2	4	12	+20
P3	5	8	+20 +12
P4	10	10	+20 +12+8
P5	14	15	+20+12+8+10
P6	2	2	+12+8+10+15
P7	3	4	+8+10+15+2
P8	1	17	+10+15+2+4
P9	9	9	+15+2+4+17
P10	6	8	+2+4+17+9

Working of the algorithm:

1. First processes 1-5 will be present in the queue. Then 1st process will be executed, making other processes in the queue waiting for the time equal to its burst time that is 20 units in this case.
2. Then new process P6 will come into the queue, even if it has high priority still P2 will be executed because it is there in queue first, so the processes have to wait for the time till P2 is executed completely and then P7 enters

3. The procedure continues till the processes are executed.
4. After the execution ends calculation of throughput and average wait time is done.

   Total Wait time = wait time (P1+P2+ P3+P4+P5+P6+P7+P8+P9+P10) = 333

   Average Wait time = Total wait time / total number of processes = 333/10= 33.3

   Throughput= #of processes per unit time= 10/341= 0.02935.

6.122 Shortest Job First Scheduling

Consider the same example and the algorithm will work by executing the process with the smallest burst time first in the queue.

5 processes are there at a time arranged in queue

Processes	Priorities	Burst Time
P1	1	20
P2	4	12
P3	5	8
P4	10	10
P5	14	15
P6	2	2
P7	3	4
P8	1	17
P9	9	9
P10	6	8

As the algorithm will work considering the process with smallest burst time in the queue.

Processes	Priorities	Burst Time	Wait Time
P1	1	20	+8+2+4+10+9+8+12+15+17
P2	4	12	+8+2+4+10+9+8
P3	5	8	
P4	10	10	8+2+4
P5	14	15	8+2+4+10+9+8+12
P6	2	2	
P7	3	4	

P8	1	17	+10+9+8+12+15
P9	9	9	
P10	6	8	

Working of algorithm:

1. First the algorithm picks P3 out of all the processes in the queue (P1 to P5) because it has the lowest burst time out of the rest and executes it. Other processes wait till the execution of P3 is executed completely.
2. With the execution of P3 new process P6 enters into the queue and its burst time is less than the rest of processes currently present in the queue so it will execute making other processes to wait for a time till it executes.
3. Now other process P7 enters the queue and again the same procedure repeats until all the processes are executed.
4. Once all the processes are completed then average wait and throughput is calculated.

   Total Wait time = wait time (P1 +P2+ P3+P4+P5+P6+P7+P8+P9+P10) = 235

   Average Wait time = Total wait time / total number of processes = 235/10= 23.5

   Throughput= #of processes per unit time = 10/255=0.039.

6.123 Priority Scheduling

Above example is taken with same assumption of queue taking same 5 processes at a time.

Processes	Priorities	Burst Time
P1	1	20
P2	4	12
P3	5	8
P4	10	10
P5	14	15
P6	2	2
P7	3	4
P8	1	17
P9	9	9
P10	6	8

Priority scheduling schedules the process execution on the basis of their priorities. The one with highest priority is executed first.

Processes	Priorities	Burst Time	Wait Time
P1	1	20	
P2	4	12	+20+2+4+17
P3	5	8	+20+2+4+17+12
P4	10	10	+20+2+4+17+12+8+8+9+10
P5	14	15	+20+2+4+17+12+8+8+9
P6	2	2	
P7	3	4	
P8	1	17	
P9	9	9	+12+8+8
P10	6	8	

Working of the algorithm:

1. The algorithm checks the process with highest priority among the processes that are present in queue and then evaluates the result. So, it will pick P1 an execute P1 and then P6 will enter into the queue. As the priority of P6 is 2 it will execute first and all other processes in the queue have to wait till its execution is over.
2. This procedure will be carried out until the all the processes are executed.
3. Once all the processes are completed then average wait and throughput is calculated.

   Total Wait time = wait time (P1 +P2+ P3+P4+P5+P6+P7+P8+P9+P10) = 324.

   Average Wait time = Total wait time / total number of processes = 324/10= 32.4.

   Throughput = 10/339= 0.029.

6.124 Round Robin Scheduling

Implementing round robin includes deciding threshold value for which it executes processes accordingly known as time slices.

Processes	Priorities	Burst Time
P1	1	20
P2	4	12
P3	5	8
P4	10	10
P5	14	15
P6	2	2
P7	3	4
P8	1	17
P9	9	9
P10	6	8

So here we choose 2 as the time slice:

Processes	Priorities	Burst Time	Execution Time 2 2 2 2 2 2 2 2 2 2 2 2 2
P1	1	20	18 16 14 12 10 8 6 4 2 0
P2	4	12	10 8 6 4 2 0 0
P3	5	8	6 4 2 0
P4	10	10	8 6 4 2 0
P5	14	15	13 11 9 7 5 3 1 0
P6	2	2	2 0
P7	3	4	4 2 0
P8	1	17	17 15 13 10 8 6 4 2 0
P9	9	9	9 7 5 3 1
P10	6	8	8 6 4 2 0

Algorithm Working:

1. The algorithm works by executing the process by repeating the cycles of execution for 2 units all the processes in the queue. Initially the P1 is executed for 2 units then resources are transferred to execute 2nd process followed by 3rd process and then 4th process and

then 5th process to again get back to first process and carry its further processing. In whole of this procedure the time for which first process has waited for its next execution after 2 units is 8 units (2+ 2+ 2+ 2).

2. The algorithm will continue till the all the processes in queue are completely executed.
3. Once all the processes are completed then average wait and throughput is calculated.

   Total Wait time = wait time (P1 +P2+ P3+P4+P5+P6+P7+P8+P9+P10) = 360

   Average Wait time = Total wait time / total number of processes = 360/10= 36.0

   Throughput = #of processes per unit time =10/360= 0.027.

6.125 Proposed Algorithm

This proposed algorithm is implemented in solving the above example.

Processes	Priorities	Burst Time
P1	1	20
P2	4	12
P3	5	8
P4	10	10
P5	14	15
P6	2	2
P7	3	4
P8	1	17
P9	9	9
P10	6	8

Before implementing above algorithm there are few assumptions that are to be made.

1. Size of queue = number of employed bees (So let us assume there are 5 bees bringing processes from different directions which according to them are the best option at that time at a same time)
2. Onlooker bees are actually computing the data. (Processor).
3. Threshold 1 is 35 units
4. Threshold 2 is 15 units.

5. If 2 or more processes at the time of threshold 1 becomes equal then the process with highest process will be picked by the processor as the means of optimized solution.

Processes	Priorities	Burst Time	Threshold time	Execution
P1	1	20	35	
P2	4	12	35 15 13 9 0	+20+2+4+9
P3	5	8 5	35 15 13 9 0 3	+20+2+4+9+12+8
P4	10	10 4	35 15 13 9 0 3 0 27 23 15 6	+20+2+4+9+12+3+8+5+8+9
P5	14	15	35 15 13 9 0 3 0 27 23 15 6	+20+2+4+9+12+3+8+5+8+9+6+4
P6	2	2		
P7	3	4		
P8	1	17 8	3 0	+12+3
P9	9	9	0 27 23 15	+3+8+5+8
P10	6	8	27 23	+5

How algorithm works:

1. The modified version of the algorithm focuses on 3 main things that are
   a. Priority
   b. Threshold time
   c. Burst time
2. First the 5 bees will take the 5 different processes and arrange them in queue. Then the threshold value of timer or clock is taken as 35 units. That means for 35 units the process will be executed according to their priority and after then the burst time will become priority. Then threshold 2 will come which will remain in algorithm for 15 units and again after then the priority will shift.
3. Initially out of P1, P2, P3, P4, P5 P1 will execute because it has the maximum priority. Other processes in the queue will wait for 20 units and threshold will reduce by 20 (35-20 =15), this shows 15 units are still remaining for threshold 1 to change to threshold 2. Then process P6 will come and it is having lesser priority than the others in the queue so it will execute for another 2 units which will bring threshold to 13 and similarly till Process P8 comes threshold is reduced to 9. After coming of P8 into

queue still it holds high priority so it will execute for 9 units and another 8 units will be left and clock changes to implement threshold 2 where burst time will become priority.

So, because of this now priority of P8 shifts from 1 to 8 and like this other process priority will also be changed.

Now the one with least burst time will hold higher priority so process P2, P3, P4, P5 all are having 0 as priority when the processes have same priority than the one with priority actually high as here its P2 so then P2 will execute for 12 seconds and all other processes will remain in queue for 12 seconds (15-12=3) so now 3 units for the threshold 2 remains. So, Process P3 with priority 5 will execute for 3 units and 5 units will be left.

Then again, the clock will transfer to threshold 1 again for another 35 units making priority of P8 as highest in the queue.

P8 will execute till 8 units (35-8= 27) other processes will wait for 8 units then P3 will execute (27-5=23), then P10 will execute (23-8=15) and so on till all the processes in the queue are finished.

4. Once all the processes are completed then average wait and throughput is calculated.

Total Wait time = wait time (P1 +P2+ P3+P4+P5+P6+P7+P8+P9+P10) = 294.

Average Wait time = Total wait time / total number of processes = 294/10= 29.4

Throughput= #of processes per unit time = 10/309=0.0323.

6.22 Validity

This experiment shows that this research study holds internal validity as it shows that the proposed algorithm is better than the previous algorithms on the basis of parameters we choose to compute the result that are average waiting time and throughput. After analyzing the result, it can be concluded that this algorithm can be used in all those areas where there is a need to properly distribute the data and optimize execution of multiple tasks. So, research study shows "external validity."

6.23 Practical Implementation

All the comparison above is in the basis of theoretical aspect but if proposed algorithm is actually implemented in any high-level language including C/C++, Java, Python etc. then the study will cheek the quality of research on the basis of time complexity, waiting time, throughput, accuracy, power consumption, space complexity, input size etc.

7. Additional Discussion and Future Scope

Proposed algorithm can also be tested on other dimensions of practical problems such as routing algorithms where messages are not arranged according to their priorities. The messages sent from sender and receiver can be sent in form of digital packets with certain period of urgency so that they don't have to suffer the condition of starvation.

This proposed algorithm can also be used in ATM transactions optimization. The ATM is performing one item at a time so with implementation of this proposed algorithm with proper implementation of artificial bee colony can be used to arrange the processes according to their priorities of usage and will evenly distribute the resources among the processes.

It can be implemented in Database Management System (DBMS) for managing different processes at same time.

This proposed algorithm can be used in the devices such as Fitbit which calculates different things at same time in order to give optimized result by organizing each process in better manner.

8. Research result to date

The research conducted above showed that as compared all the other scheduling algorithm including FCFS, Round Robin, priority scheduling etc. the modified algorithm has more throughput and less waiting time closer to shortest job first.

It also curbs the disadvantages of other scheduling algorithm as in FCFS, processes in the queue starves if any process which has come first takes lots of time to complete this is called "Convoy Effect" but the modifies algorithm processes in the queue does not starve for the whole time as with the use of threshold 1 and threshold 2 (2 timers) the priorities are usually being shuffled between actual priority and burst time which gives every process fair chance of being executed.

In SJS If any process even with high priority comes and has urgency of being executed but having more burst time it will not being executed and has to wait until the one with less burst time is scheduled so this demerit is being looked upon in the modified algorithm. As timers keeps the shuffling which maintains urgency and also does not let any process to starve for the whole time. In priority scheduling, there is demerit of starvation which is being countered with modified algorithm.

Round Robin also does not focus on urgency or the one with higher priority so, this is also being tackled by modified algorithm.

9. Conclusion

This shows that the conceptual result proves that with inclusion of two timers the priorities can be shuffled and with shuffling of priorities between 2 good scheduling algorithms we can come up with the algorithm that can overcome their disadvantages.

With above experiment we can also conclude that the average wait time is less and throughput is more in case of modified algorithm as compared to another algorithm except SJS. If proposed algorithm threshold 1 is set for large time for example 200 units then it will become priority scheduling or if the threshold 2 is set for large time it will become SJS.

References

1. Comparison of different scheduling algorithms in OS. (2013). Retrieved from www.studytonight.com/operating-system/comparision-scheduling-algorithms
2. Hossain, M. S., & Shafie, A. (2013). Application of artificial bee colony (ABC) algorithm in searchof optimal release of Aswan High Dam. In *Journal of Physics: Conference Series* (Vol. 423, No. 1, p. 012001). IOP Publishing.
3. Karaboga, D, (2005). An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
4. Okdem, S., Karaboga, D., & Ozturk, C. (2011, June). An application of wireless sensor network routing based on artificial bee colony algorithm. In *Evolutionary Computation (CEC), 2011 IEEE Congress on* (pp. 326-330). IEEE.
5. Noon, A., Kalakech, A., & Kadry, S. (2011). A new round robin based scheduling algorithm for operating systems: dynamic quantum using the mean average. *preprint arXiv:1111.5348*.
6. Rajput, I. S., & Gupta, D. (2012). A priority based round robin CPU scheduling algorithm for real time systems. *International Journal of Innovations in Engineering and Technology*, *1*(3), 1-11.
7. Silberschatz, A., Gagne, G., Galvin, P. (2005). CPU Scheduling, "Operating System Concepts, Eighth Edition. Chapter 5. Retrieved from https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/5_CPU_Scheduling.htm
8. Yetkin, M., & Berber, M. (2014). Implementation of robust estimation in GPS networks using the artificial bee colony algorithm. *Earth Science Informatics,*