# Empirical Framework to Mitigate Problems in Longer Job First Scheduling Algorithm LJF+CBT

Ibrahim Abdullahi
Department of Mathematics/Computer Science
Ibrahim Badamasi Babangida University, Lapai.
Niger State, Nigeria

S.B Junaidu
Iya Abubakar Computer Center,
Ahmadu Bello University, Zaria. Kaduna State,
Nigeria

## ABSTRACT
Process as an individualistic entity program and a program in execution requires good scheduling algorithm for its throughput and latency measures. This work made a study of existing process scheduling algorithms and carefully examines the Longest Job First (LJF) algorithm as a key to minimizing the overall *Average Waiting Time (AWT)* and the *Average Turn-Around Time (ATAT)* in multiprocessing systems to find ways of making the algorithm popularly usable in the field of computer application and life endeavors. A sample of generated process attributes of burst-time along each process were used to simulate scenario, by a new technique we referred to as Combinational Burst-Time (CBT) to curtail the major problems of starvation of the shorter jobs in queue. CBT as a framework minimized the large numbers of *context switching (CS)*, *starvation* and reduced convoy problems.

**Keywords:** Operating system, Process Scheduling, Average Waiting Time, Longest Job First, Average Turnaround Time, Combinational Burst Time CBT, Starvation.

## 1. INTRODUCTION
A process is submitted on creation from its initial state admitted to the first queue upon selection from the job pool and followed by the next phase of execution "ready" where the Schedulers take responsibility of using the designed form of scheduling following the specified algorithm. Running state as shown on the Process State diagram on Figure 1 depicts the most concern in the scheduling concept. Processes are then interrupted on users operations either through error or operational time sharing. From the 'ready', a scheduler takes processes to the 'running' phase by the mid-term, long-term and short-term schedulers. An algorithm usually determines the movements of the process into the CPU to grant execution, these algorithms are referred to as the process schedulers. A 'running' process can be moved to the 'waiting' phase as a result of an I/O request. The request is granted and the operating system re-schedules the process through I/O completion. The above scenario is then represented and the evaluation criteria are measured using *Average Waiting Time (AWT)*, *Average Turn-Around Time (ATAT)*, and the numbers of Context Switches *(CS)*. This paper work by comparing the popular schedule types: Longest Job First (LJF), First Come First Served (FCFS) and a new proposed concept called Longest Job First and Combinational Burst Time (LJF+CBT). Figure 1 illustrates a typical state running of all processes in the multiprocessor environment.

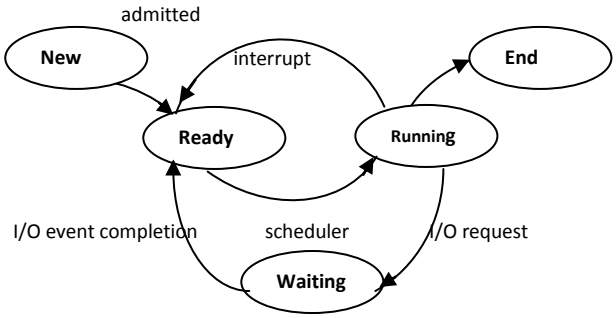


**Figure 1: Process State Diagram**

## 2. POPULAR SCHEDULING ALGORITHMS
Scheduling algorithms defines the way and sequence of execution of processes waiting in the ready queue until the process gains the Central Processing Unit (CPU) to finish execution [7]. Some common scheduling algorithms are: First Come First Serve (FCFS), Shortest Job First (SJF), Priority Scheduling, Round Robin (RR), and Shortest Remaining Time Next (SRTN) *etc*. In the FCFS practice, it schedules the processes in their orders of arrivals such that the first process that arrive the ready queue is firstly given the CPU to execute (Run). Typically, FCFS is referred as the lenient scheduling as its application is seen practically in ideal situations such as grocery shop, gas station services, and bookings in train stations and so on. SJF differs from the FCFS in the orders of burst time (required time to finish execution) and times of processing. SJF take into consideration the least processing times as the first to be scheduled. This Scheduling type has a priori knowledge of all processing times of the processes in the ready queue before the ordering is set. Priority Schedule on the other hand defines the order of the process execution based on the priority numbers. The priorities are usually set and defined by the user as the lower number, implies the highest priority and vice versa. Round Robin (RR) type moves processes in and out of the CPU using the order of processing defined a time quantum (q). RR works as a FCFS scheduling and cyclic concept but differ by the introduction of the quantum (q). SRTN is a type of algorithm that admits the shortest processing times of the processes in queue; it is also referred to as Shortest Remaining Processing Times.

# 3. LJF+CBT FRAMEWORK

## 3.1 Description

In the proposed framework, an experiment was carried out among different processes of distinct burst-times, a combinational burst time (CBT) is calculated following some laid down rules in our model, on evaluation of the model by algorithm steps, it is found that starvation problems on the queue is reduced as well as the overall waiting times, turnaround time and the total numbers of context switch was minimized as compared to the ordinary Longest Job First (LJF) scheduling type. Therefore, the frame work will be named Longest Job First Combinational Burst Time (LJF+CBT).

## 3.2 CBT Model

Every process has factors associated to its functions which include user priority, burst time and arrival time. [23]. Most scheduling concepts, one or two of these factors are used for designs and evaluation. The CBT model will focus on job identity, burst time, arrival time and the phase /stage of execution which will be referred to as a counter. The phase of execution stands the model out as compared to most existing types. For First Come First Served (FCFS), Shortest Job First (SJF) and Priority Scheduling only a factor is set where as the four factors listed in this paper shall reduce the *Average Waiting Time (AWT), Average Turn-Around Time (ATAT)* and the numbers of *Context switches (CS)*.

Description: Given $N$ jobs in a pool as a set of processes $X$, each with its distinct burst times $\delta$ and a phase of execution $\lambda$.

$N = Xi, Xj, \dots, Xn \dots (1)$. Where each process has its associated burst time δ and $i, j, k, \dots, n$ are sets of numbers. It results into $Xi\delta i$ for every $X$ to be assigned to the CPU at time $\lambda$. See Figure 1.

Another set of process $Yi\delta i = Xi\delta i + Xj\delta i \dots (2)$

define shorter jobs that will be merged to make a long job on satisfying the merging condition in the model.

If $Xi\delta i \leq n$, save as short job else schedule as long job where n = the threshold measure of job categorization.

Move all short jobs to an array to merge as $Yi\delta i$

Merge short jobs $Yi\delta i = Xi\delta i + Xj\delta j$ at phase $\lambda$ and provide a new order in the ready queue such that longer jobs are first considered and the new merged jobs shall be treated as thus longer jobs too.

New order shall be: $CPU = \{Xi\delta, Yi\delta i, Xj\delta j\} \dots (3)$ CBT defines the new order which the processes shall be admitted per phase into the CPU thereby reducing the delays of the shorter jobs after merging.

## 3.3 CBT Algorithm

1. Begin
2. Sort all processes in descending order of Longer Burst times.
3. If $Xi\delta i > n \; for \; each \; i, do$
4. Move $Xj\delta j$ to array if $Xj\delta j < n$
5. If $Xk\delta k$ exist and less than $n$; then merge $Xj\delta j + Xk\delta k = Yi\delta i$
6. Normalise until $Xi\delta i \leq n$ has no $Xj\delta j$ to merge.
7. Compare $Xi\delta i$ with $Yi\delta i$
8. If $Xi\delta \geq Yi\delta i$, place before $Yi\delta i$
9. Else place after $Yi\delta i$ in the queue
10. End

Note that n is set by the operating system designer as a number that categorizes jobs to be of type long and short.

## 3.4 CBT Test Case

Given a set of 10 processes each with its burst time, we use the CBT model and algorithm to test the performance of the LJF and the FCFS to investigate the total performance by setting the evaluation metrics of *Average Waiting Time (AWT), Average Turn-Around Time (ATAT)* and *Context Switch* (*CS)*. The processes have burst times between 1-50 and thus, processes with burst time less or equal to 25 are referred to as shorter jobs whereas processes with burst times above 25 are termed as longer jobs for this analysis. All processes will be treated using the CBT and without the CBT. Tables will be shown for clear understanding of the calculations.

# 4. EXPERIMENT CASE FOR LJF+CBT

Assumptions: In a multiprocessing system where processes of high numbers exist, starvation is the ultimate problem as shorter jobs wait endlessly in the continuous case of execution. For this experiment we shall consider a discrete case where we have only 10 processes in the system ready for processing.

*A. Data set*

Random generator in R Language was used to obtain the sets of burst times for the experiment, the ordering of the generated numbers were used as the process identifications for computational purpose.

*B. Performance Evaluation*

We shall be using the most common metrics for the analysis which are *Average Turn-Around Time (ATAT), Average Waiting Time (AWT)* and the total numbers of *Context Switch (CS).* For better performance, the numbers of *AWT, ATAT and CS* should be less.

*C. Experiment Nature*

We assumed all processes arrived at time $t = 0$ and all processes have same characteristics as we didn't distinguish between the I/O bound and the CPU bound processes. LJF and LJF+CBT shall be presented.

## 4.1. Experiment Process

For experimental case in this paper, a snap shot of processes shall be presented; each of the process was generated by statistical analysis. R language was used as a simulation tool to obtain the various burst times of the processes. User priority number column is also captured on the snap shot to enable further testing of the CBT model using the priority scheduling algorithm.

**Table 1: Process snap shot**

| Processes | Arrival Time | Burst Time | User Priority |
|---|---|---|---|
| P1 | 0 | 9 | 7 |
| P2 | 0 | 27 | 9 |
| P3 | 0 | 14 | 8 |
| P4 | 0 | 30 | 3 |
| P5 | 0 | 22 | 5 |
| P6 | 0 | 19 | 10 |
| P7 | 0 | 45 | 2 |
| P8 | 0 | 39 | 1 |
| P9 | 0 | 7 | 4 |
| P10 | 0 | 10 | 6 |

### *4.1.1 First Come First Served (FCFS)*

For the FCFS, the ordering of the Processes by the identification gives the form of the execution, since all processes are assumed to arrived at Time = 0, therefore, P1, P2, P3, P4, P5, P6, P7, P8, P9 and P10 is maintained.

See Table 1.

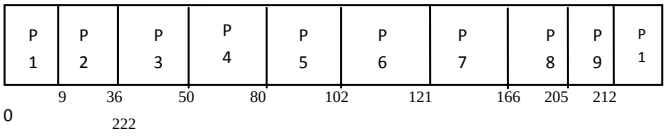


**Figure 2: Gantt chart FCFS**

*Note: the Gantt chart representation doesn't show the real scale of execution as P9 and P1 finished their executions at times 212 and 222 respectively.

*Waiting Time (WT) = Time first Scheduled – Arrival time*

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

*For each process,* WT= *Time first Scheduled – Arrival Time*

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

Thus *AWT* = (0 + 9 + 36 + 50 + 80 + 102 + 121 + 166 + 205 + 212) / 10

= 981/ 10

= 98.1

*Context Switch (CS) = 9*

*Turn-Around Time (TAT) = Overall Time a process spent in the system*

*= Time of Process completion - Arrival Time*

*Average Turn-Around Time = Sum of each Process / Total numbers of Processes.*

*Thus :TAT in the system will be thus:* P1: (9-0) = 9, P2: (36-0) =36, P3: (50-0) = 50, P4: (80-0) =80, P5: (102-0) = 102, P6: (121-0) = 121, P7: (166-0) = 166, P8: (205-0) = 205, P9: (212-0) = 212, P10: (222-0)= 222.

*ATAT = (9 + 36 + 50 + 80 + 102 + 121 + 166 + 205 + 212 + 222) / 10*

*= 1203/10*

*= 120.3*

### *4.1.2 Longest Job First*

The LJF algorithm is the most uncommon among the scheduling algorithm; it schedules the process with the highest burst time before shorter processes. It is seen as the reverse of the SJF; the major disadvantage includes *starvation* of the shorter processes [5] [16].

This form the bases for this research as the new CBT frame work aim at solving the problems of *starvation* and *convoy*.

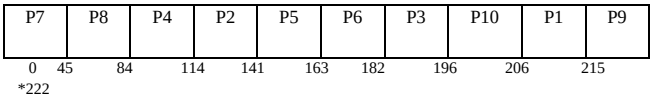


**Figure 2: Gantt chart LFJ**

*Note: Process P9 finishes its execution at time 222.

**Table 2: Re-arranging snap shot for LJF**

| Processes | Burst time |
|---|---|
| P7 | 45 |
| P8 | 39 |
| P4 | 30 |
| P2 | 27 |
| P5 | 22 |
| P6 | 19 |
| P3 | 14 |
| P10 | 10 |
| P1 | 9 |
| P9 | 7 |

*Waiting Time (WT) = Time first Scheduled – Arrival time*

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

*For each process,* WT= *Time first Scheduled – Arrival Time*

P1: (206-0) = 206, where P1 is the first Process that arrived at Time = 0 and thus had to wait for the longer processes to proceed.

P2: (114-0) = 114, where P2 arrived at Time = 0 and scheduled at 114.

P3: (182-0) = 182, P4: (84-0) = 84, P5: (141-0) = 141, P6: (163-0) = 40, P7: (0-0) = 0, P8: (45-0) = 45, P9: (215-0) = 215, and P10: (206-0) = 206.

From the formula

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

Thus *AWT* = (206 + 114 + 182 + 84 + 141 + 163 + 0 + 45 + 215 + 206) / 10

= 1356/ 10

= 135.6

Context Switch = 9

*Turn Around Time (TAT) = Overall Time a process spent in the system*

*= Time of Process completion - Arrival Time*

*Average Turn-Around Time = Sum of each Process / Total numbers of Processes.*

*Thus :TAT in the system will be thus:* P1: (215-0) = 215, P2: (141-0) =141, P3: (196-0) = 196, P4: (114-0) =114, P5: (163-0) = 163, P6: (182-0) = 182, P7: (45-0) = 45, P8: (84-0) = 84, P9: (222-0) = 222, P10: (206-0) = 206.

*ATAT = (215 + 141 + 196 + 114 +163 + 182 + 45 + 84 +222+ 206) / 10 = 1568/10*

*= 156.8*

### *4.1.2 Longest Job First + CBT*

This is the papers contribution to the field of operating system scheduling design, the CBT model is now evaluated and the results will be compared to the entire existing scheduling algorithm concept. The model designed in Section 3; is fully tested and the procedures are presented as follows:

Let all processes be sorted and merge as conditions in the model suggest.

From the CBT, Process P5 and P6 satisfy the condition of merging resulting a Longer Process *P5, 6 = 41, because the burst times are less than 25. Consequently, Processes P3 and P10 merge by the CBT model resulting into *P3, 10 = 24, Process P1 also generates to *P1, 9 = 16.

Sorting and presenting the result shows a new identification and burst times of the CBT.

**Table 3: Process snap shot after CBT**

| Processes | Burst time |
|---|---|
| P7 | 45 |
| *P5,6 | 41 |
| P4 | 30 |
| P2 | 27 |
| *P1,9 | 16 |

Results show that the total processes are reduced to 5.

The Gantt chart and evaluation calculations is shown on Fig 3: Gantt chart for the *LJF+CBT*

| P7 | *P5,6 | P4 | P2 | *P3,10 | *P1,9 |
|---|---|---|---|---|---|

**Figure 3: Gantt chart LJF+CBT**

LJF + CBT:

*Waiting Time (WT) = Time first Scheduled – Arrival time*

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

*For each process,* WT= *Time first Scheduled – Arrival Time*

*P19: (157-0) = 157, where *P1, 9 is the result of merging P1 and P9, at Time = 0 and thus had to wait for the longer processes to proceed.

P2: (116-0) = 116, where P2 arrived at Time = 0 and scheduled at 116.

*P3,10: (133-0) = 133, P4: (86-0) = 86, *P5,6: (45-0) = 45,

From the formula

*Average Waiting Time (AWT) = Sum of all Processes Waiting Time/ Total numbers of Processes.*

Thus AWT = (157 + 116 + 133 + 86 + 45) / 10

= 537/ 10

= 53.7

Context Switch = 5

In summary, the LJF+CBT gives an *Average Waiting Time (AWT)* far less than all scheduling algorithms presented.

The *Context Switch (CS) = 5.*

*Turn Around Time (TAT) = Overall Time a process spent in the system*

*Time of Process completion - Arrival Time*

*Average Turn Around Time = Sum of each Process TAT / Total numbers of Processes.*

*Thus: TAT in the system will be thus:* *P1, 9: (173-0) = 173, P2: (133-0) =133, *P3, 10: (157-0) =157, P4: (116-0) =116, *P5, 6: (86-0) = 86, P7: (45-0) = 45.

*ATAT = (173 + 133 +157 + 116 +86 + 45) / 10*

*= 710/10*

*= 71.0*

## 4.2 Comparisons and analysis

The obtained results from the metric evaluation calculations are presented as follows:

### *4.2.1 FCFS Vs LJF+CBT*

The First Come First Served (FCFS) scheduling algorithm produced the results of an *Average Waiting Time (AWT)* = 98.1 and an *Average Turn-Around Time (ATAT)* = 120.3 which resulted in the system having *Context Switch (CS)* = 9. Results of the comparisons presented the Gantt charts for each scheduling algorithm.

**Table 4: FCFS VS LJF+CBT**

| Algorithm | *Average Waiting Time (AWT)* | *Average Turn Around Time (ATAT)* | *Context Switch (CS)* |
|---|---|---|---|
| FCFS | 98.1 | 120.3 | 9 |
| LFJ+CBT | 53.7 | 71.0 | 5 |

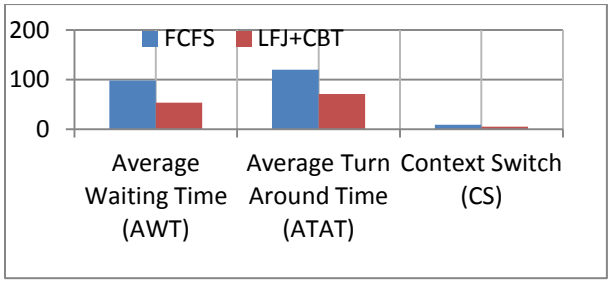


**Figure 4: Graph of FCFS VS LJF+CBT**

### *4.2.2 LJF Vs LJF+CBT*

Results of the LJF Scheduling algorithm obtained using Table 2 and 3; proves that the LJF+CBT performed better that the LJF. Gantt chart and numbers of context switches are presented thus:

**Table 5: LJF VS LJF+CBT**

| Algorithm | *Average Waiting Time (AWT)* | *Average Turn Around Time (ATAT)* | *Context Switch (CS)* |
|---|---|---|---|
| LJF | 135.6 | 156.8 | 9 |
| LFJ+CBT | 53.7 | 71.0 | 5 |

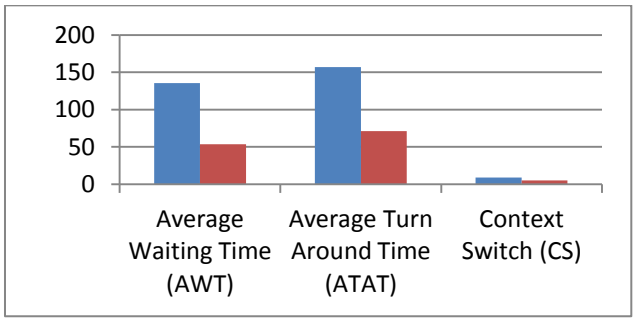


**Figure 5: Graphs of LJF Vs LJF+CBT**

## 5. CONCLUSION

In conclusion, it can be presented that an empirical data analysis and simulation was carried out and the LJF+CBT performs better than the FCFS and LJF scheduling algorithms as evaluation shows using the most popular metrics of *AWT*, *ATAT* and *CS*. The result shows that in a case where short jobs outnumber the long ones, LJF+CBT will be better. The Experiments used a threshold value of n to be 25 as upper bound 50/ 2 to categorize jobs as shorts ad long. The simulation performed better with a threshold of this nature using the numbers generated between 1 and 50. Since FCFS algorithm is applicable in the practice, we therefore present the case of genetic chromosome match as LJF+CBT. In manufacturing industries, LJF shall be applicable when the equipments are new and performing at optimal capacity in the field of mechanical engineering and locomotive equipments, it is therefore advisable to start the longer task before the machine tires around and loose some of its efficiency degree of performance. Future work shall suggest how LJF+CBT can

be incorporated into an operating system design in mobile phones (multi-tasking), video games and high performance computing projects to minimize the challenging problem of starving the shorter processes in the queue.

## 6. REFERENCES

[1] Achim, S. (2005). *On Performance Evaluation of Slackness option for the Self-turning dynP scheduler.* Central Institute for Applied Mathematics. Julich, Germany: RSJ.

[2] Baptiste, P. (1994). *Constraint- Based Scheduling : Two Extensions.* University of Strathclyde, Department of Manufacturing and Engineering . Scotland:

[3] Boleslaw, S. K. (1990). Mutual Exclusion Revisited. *Fifth Jerusalem Conference in Information Technology* (pp. 110-117). Israel, Jerusalem: IEEE Computer Society Press.

[4] Cheng, T., & Kahlbacher, H. (2006). *A proof for the Longest Job First Policy in one machine Scheduling.* CA: Naval Research Logistics.

[5] Dalibor, K., Rudova, H., Raneiri, B., Marco, P., & Capannini, G. (2005). Comparison of Multi-Criteria Scheduling. Springer. ISBN 978-0-387-09456-4

[6] Ernst, B. W., Schroeder, B., & Urvoykella, G. (2007). Scheduling in Practice. *Journal of Parallel Processing , 34* (4)

[7] Galvin, P. B., Abrahm, S., & Gagne, G. (2004). *Operating System Concepts* (7th Edition ed.). (B. Zobrist, K. Santor, & M. Lesure, Eds.) NJ, USA: John Wiley and Sons Inc.

[8] Gerald, S., Garima, K., & P, S. (2004). Job Fairness in Non-preepmtive Job Scheduling. *International Journal on Parallel Processing* , 186-194.

[9] Herve, M. (2007). On Scheduling fees to prevent merging, splitting and transfering Jobs. *Journal of Mathematics of Operational Research* , 266-283

[10] Idris, R. A., & Okopa, M. (2011). Modeling and Evaluation of SWAP scheduling Policy under varying Job size distribution. *Network Conference* (pp. 115-120). Uganda: IARIA.

[11] Javier, C., Fatos, X., & Ajith, A. (2007). Genetic Algorithm Based Schedulers for grid Computing Systems. *International Journal of Innovative Computing, Information and Control , 3* (6), 25-32.

[12] Ludmila, C., Lau, R., Burose, H., Subramanian, V., Bernhard, K., & Kuttiraja, V. (2011). Run-Time Performance Optimization and Job Management in Data Protection Solution. *IFIP/IEEE Symposium in Integrated Management* (pp. 20-27). Dublin: IEEE.

[13] Rakesh, M., Das, M., Lakshmi, M., & Sudhashree, P. (2011). Design and Performance Evaluation of a New Proposed Fittest Job First Dynamic Round Robin (FJFDRR) Scheduling Algorithm. *International Journal of Computer Information System , 2* (2), 23-93.

[14] Raksha, S., Vishnu, K., Mishra, M. K., & Bhuyan, P. (2010). A Survey of Job Scheduling and Resource Management in Grid Computing. *World Academy of Science Engineering and Technology* , 461-466.