Chapter 8
Bat Algorithm

8.1 Introduction

Bat algorithm is an innovative or population-based technique which belongs to the swarm intelligence. It is also referred to as a metaheuristic algorithm developed by Yang [1]. The bat algorithm as a unique algorithm provides a suitable solution technique than numerous and prevalent classical and heuristic algorithms. The algorithm is used for quick decision making and for solving complex problems in diverse fields of operations ranging from engineering, business, transportation, and other fields of human endeavor. It is important to understand the communication and navigational pattern of bats while defining the algorithm since the algorithm is based on the micro-bats echolocation (EL) [1]. EL is an enchanting and captivating sonar propensity produced bats. The appealing wave of sound made by the bats is a great strength they often exhibit while searching for prey. The wave of sound is a formula they adopt not only in food search but serves other purposes [2]. For instance, while searching for food in a completely dark environment, there might be obstacles or dangers on their way to the food source. This can easily be detected in some magical manner as they can sense and discern possible danger on their way to the food source as shown in Fig. 8.1 [3]. The structure and flying position of a bat is shown in Fig. 8.1. BA is therefore a very powerful algorithm as it uses the frequency-tuning methodology for range intensification of the solutions in the population. At the same time, the BA system implements the instinctive skyrocketing or zooming procedure to maintain stability during the food search or exploration process. In developing the BA, the pattern of direction finding, manipulation, and exploration during prey hunting is taken into consideration by mimicking the disparities in terms of emission rates of pulse and intensity of sound released by the bats while hunting or searching for prey. The algorithm demonstrates a high level of efficiency with a distinctive swift start process (Fig. 8.2).

M. O. Okwu and L. K. Tartibu, *Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications*, Studies in Computational Intelligence 927, https://doi.org/10.1007/978-3-030-61111-8_8

Fig. 8.1 Structure and flying position of a bat

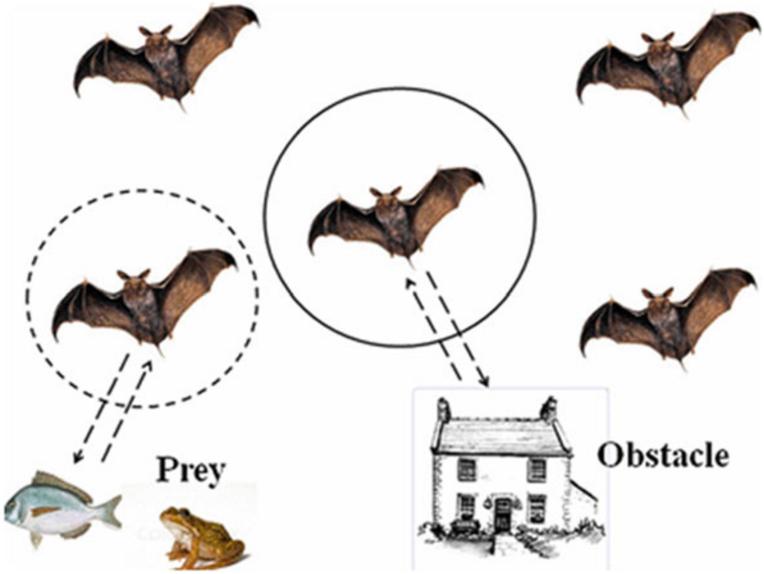


Fig. 8.2 Behavior and navigation pattern of bat

8.2 Idealized Rules of the Bat Algorithm

It is very important to note the rules associated with bats. Since bats emit loud ultrasonic sound waves and listen to the echo that reflects from the surrounding objects, the bat algorithm uses some idealized rules for uncomplicatedness. The following section points out some common rules of bat algorithm [1]:

- All bats employ the echolocation pattern to perceive sense or discern prey, barrier, distance, predator, or any form of obstacle during direction-finding, routing, or path navigation. This navigation pattern is performed in a very magical way, strange to man.
- Bats, in the course of direction-finding or path navigation fly haphazardly with a supposed velocity, v_i while maintaining position, x_i. They maintain a fixed frequency 'f' with varying wavelengths and loudness, A_i to reach their prey. They can adjust the frequency of pulse emission, r_i.
- As they get close to the prey, pulse increases, and loudness decreases.

The flow diagram describing the bat algorithm is shown in Fig. 8.3.

8.3 Example of Implementation of BAT Algorithm

Let's consider 3 bats, the Bat population = 3 ($i = 1, 2.3$), x_1, x_2 and x_3 represent respectively the position of the three bats.

Steps of the BAT algorithm implementation.

- Initialization of the frequency (f_i), the bat population (x_i), the loudness (A_i), the Pulse (r_i) and the velocity (v_i) with ($i = 1, 2, 3, \ldots n$).
  Initially ($t = 0$), we will assign random values to the frequency (f_i), position (x_i), loudness (A_i), Pulse (r_i) and velocity (v_i) as follows (Table 8.1):
- While t < Max number of iterations (t = 1, 2, 3, 4, 5, …), new solutions will be generated by adjusting frequencies, velocities, and locations.

BAT 1: Suppose that we consider 20 iterations for this example. Hence for the first iteration ($t = 1$), the condition is true. Therefore, new solutions will be generated by adjusting the frequency and updating velocity and position. The following equation will be used to adjust the frequency (NOTE: BAT can automatically adjust frequency).

$$f_i = f_{min} + \beta(f_{max} - f_{min}) \tag{8.1}$$

with the domain size of the problem $[f_{min}, f_{max}] = [0, 10]$ and $\beta \in [0, 1]$

$$v_i^t = v_i^{t-1} + \left(x_i^t - x_{gbest}^t\right) f_i \tag{8.2}$$

where x_{gbest}^t is the current best (nearest) position.

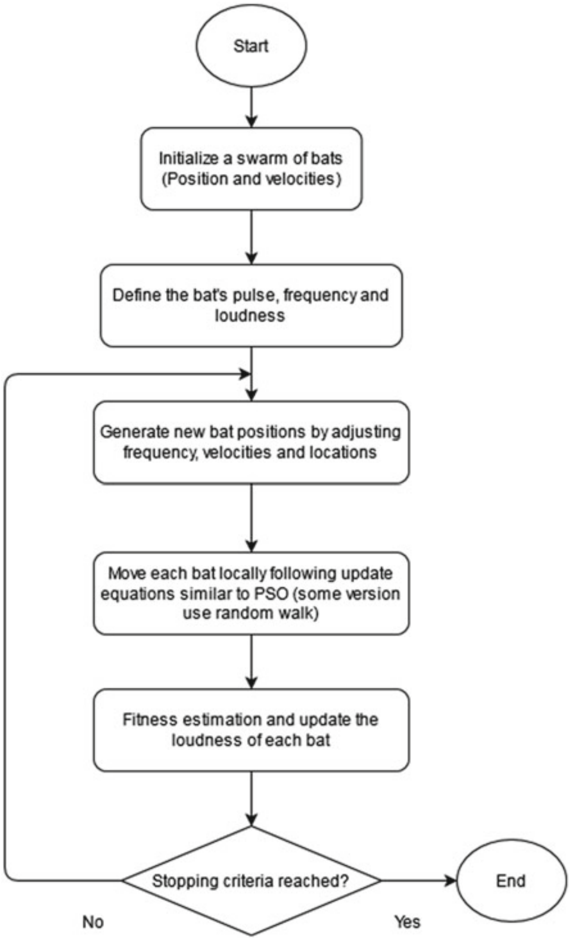


Fig. 8.3 Flow diagram of bat algorithm [1]

Table 8.1 Initial BAT parameters

BAT 1	BAT 2	BAT 3
$f_0 = 2$	$f_0 = 6$	$f_0 = 3$
$x_0 = 4$	$x_0 = 8$	$x_0 = 5$
$v_0 = 3$	$v_0 = 7$	$v_0 = 4$
$A_0 = 1$	$A_0 = 0.8$	$A_0 = 0.9$
$r_0 = 0$	$r_0 = 0.2$	$r_0 = 0.1$

$$x_i^t = x_i^{t-1} + v_i^t \tag{8.3}$$

- If the random value rand > r_i, selection of a solution among the best solutions can be done. Local solution around the best solution can be generated.

Table 8.2 Updated BAT parameters

BAT 1	BAT2	BAT3
$f_1 =$ $2 + 1(8 - 2) = 8$	$f_2 =$ $3 + 1(7 - 3) = 7$	$f_3 = 1+(5-1) = 5$
$x_1 = 4 + 35 = 39$	$x_2 = 8 + 63 = 71$	$x_3 = 5 + 29 = 34$
$v_1 =$ $3 + (4 - 0)8 = 35$	$v_2 =$ $7 + (8 - 0)7 = 63$	$v_3 =$ $4 + (5 - 0)5 = 29$
$A_1 = 1$	$A_2 = 0.8$	$A_3 = 0.9$
$r_1 = 0$	$r_2 = 0.2$	$r_3 = 0.1$

For the three bats, r_1, r_2 and r_3 are respectively equal to 0, 0.2 and 0.1. We will choose the random value between [0, 1], let's say 0.5. For BAT 1, 0.5 > 0: TRUE, for BAT 2, 0.5 > 0.2: TRUE and for BAT 3, 0.5 > 0.1: TRUE. If the condition is true, we will select a solution among the best solution.

How to select the best solution? According to the frequency: Best solution = if the prey is in the scope of frequency increase. The frequency of the waves increases if the BAT finds the prey (i.e. best solution). Referring to Table 8.2, the highest frequency corresponds to BAT 1, which is the best solution in this case.

We can generate a local solution around the best solution. We will update the position of the first BAT using the following equation:

$$x_{new} = x_{old} + \varepsilon A^t \tag{8.4}$$

with $\varepsilon \in [-1, 1]$

$$x_{new} = 35 + 1.1 = 36$$

- If rand < A_i and $f(x_i) < f(x_*)$ accept the solution and increase r_i and reduce A_i (as the BAT moves closer to the prey, the loudness decreases and pulse rate increases). $f(x_i)$ and $f(x_*)$ depend on the new BAT location and the best solution depending on the function. If the solution is TRUE, we will increase r_i and reduce A_i using the following equations:

$$r_i^{t+1} = r_i^0\left[1 - e^{\gamma t}\right] \tag{8.5}$$

$$A_i^{t+1} = \propto A_i^t \tag{8.6}$$

where the two constant $\propto$ and γ are in the range [0, 1].

Let's assume $\propto = \gamma = 0.9$, the updated values are:

BAT 1	BAT2	BAT3
$f_1 = 8$	$f_2 = 7$	$f_3 = 5$
$x_1 = 36$	$x_2 = 71$	$x_3 = 34$
$v_1 = 39$	$v_2 = 63$	$v_3 = 29$
$A_1 = 1$	$A_2 = 0.8$	$A_3 = 0.9$
$r_1 = 0.4$	$r_2 = 0.2$	$r_3 = 0.1$

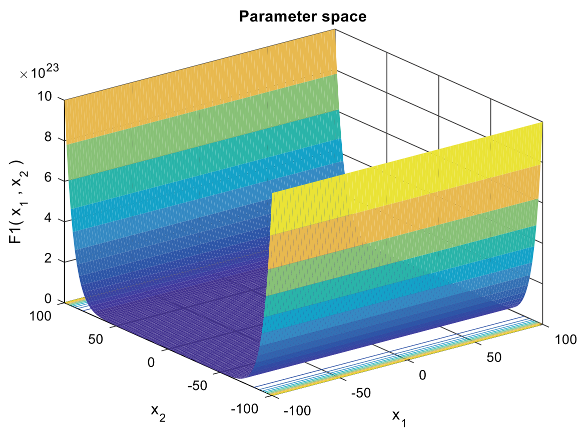
- We can now rank the BAT and find the current best. We will check the loudness first and the frequency. Based on that, the first BAT has the highest loudness and frequency. It is the current best solution.

8.4 Application of BAT Optimization Algorithm with a Numerical Example

A MATLAB code has been provided in the Appendix to illustrate the implementation of the BAT algorithm for a typical optimization problem considering a numerical equation $F(x) = ({x_1}^6 + {x_2}^4 - 17)^2 + (2x_1 + x_2 - 4)^2$ describing the Wayburen function.

The model considers 10 bats and the maximum number of iterations was 1000. The best solution obtained by BAT is: [1.3138 1.8528 0.261 0.83905 0.34859 1.1436 0.73859 1.7623 0.16537 0.28717]. The best optimal value of the objective function found by BAT is: 0.23604. The parameter space and objective space are shown respectively in Figs. 8.4 and 8.5. The MATLAB code is provided in Appendix F. A detailed description of the CODE is provided in Ref. [1].

Fig. 8.4 Parameter space



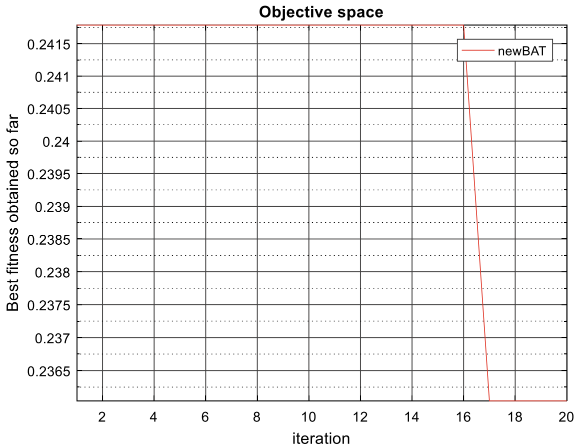


Fig. 8.5 Objective space

8.5 Application of BAT Optimization Algorithm for Traffic Congestion Problem

Case Scenario

A number of problems can be caused by congested traffic. Congestion is mainly caused by intensive use of vehicles. Traffic congestion in Delta State is a big challenge especially in the morning and night hours. Individuals who navigate to workplace in the morning hours (8 a.m.) encounter this recurring traffic congestion, leading to delay, high fuel consumption, environmental pollution and other operating costs. Also, returning home after work (4 p.m.) could be challenging too. This issue is common at certain locations in the state (PP junction in DSC and Mofor junction). Another location spotted with a high level of traffic congestion is the Udu road to Enerhen junction. From observation, the roads are very narrow to allow uninterrupted traffic flow.

Observation

It was observed that the transportation system in the state was poorly planned, as alternative routes were not mapped out for easy access, and to further congest the road, marketers and roadside sellers occupy the vast extension of the road. All these led to traffic congestion in a section of the state under un-signalized circumstances.

Network Architecture

To address the problem, road network can be taken as a directed graph G = (N, a), where 'N' is the set of nodes; i.e., the road junctions 'a' is the links connecting the junctions as shown in Fig. 8.6.

For each pair of origin (O) and destination (D), (O-D) signify a nonnegative travel demand, d_{rs}. The road network can represent a highly connected graph, where each node "i" is accessible by another node "j" following the directed path of the network

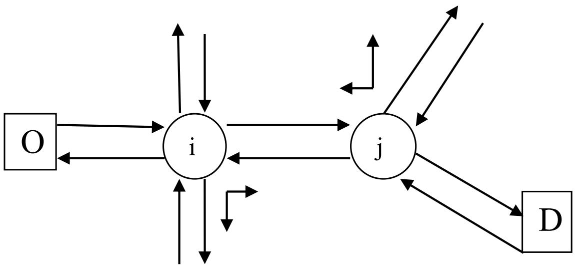


Fig. 8.6 An intersection of the network showing an O-D pair connected by a 3-way and a 4-way junction

'N'. It is assumed that the links connecting the nodes have a travel time function t_a, for the assigned rate of flow x_a.

This study aims to mimic the bat algorithm for solving traffic congestions issues in the state.

The major objectives include to:

(i) choose possible shortest route to travel from source to destination while maintaining uninterrupted traffic flow;
(ii) reduce the traffic delay at each junction under un-signalized circumstances.

The continuous network design model is chosen with budget constraints for the link capacity expansion. The objectives are interdependent and can be formulated as a bi-level problem. The upper level is responsible for reducing the travel time of the assigned road users or travellers. The lower level is the traffic assignment model which estimates the travelers flow. The model is presented clearly in Fig. 8.7.

This model can be formulated mathematically as shown below for both the layers.

Upper-Level Function

$$\min T(y) = \sum_{a \in A} \sum_{a \in A(y)} x_a t_a(x_a) + d_a y_a \tag{8.7}$$

$$\text{Such that, } \sum c_a y_a \leq B$$

where A is the set of all links 'a' in the network 'N'.

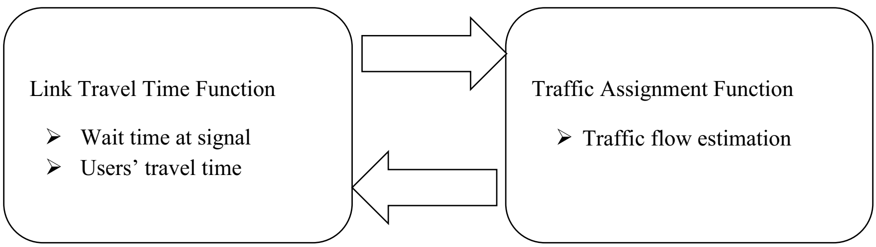


Fig. 8.7 Bi-level model for traffic network determination problem

X(a) gives the user equilibrium flow, which is estimated from the lower level of the model for the assigned value of link capacity ‘y’. c_a is construction cost for link ‘a’ and B is the budget.

Lower Level Function

$$\min \sum_{a \in A} \int_0^{x_a} t_a(u)du \tag{8.8}$$

$$\text{Such that, } \sum f_k = d_{ij} \quad f_k \geq 0$$

$$\sum_{rs} \sum f_k \delta_{ij} = x_a \forall \delta_{ij} = \{^{1;}_{2;} \text{ifa} \in k$$

where f is the flow on path k and r-s are the nodes on the path connecting O-D pair.

The flowchart for the traffic signal optimization from a BA perspective is shown in Fig. 8.8.

Solution Procedure

Using Delta State, Udu road as a case study from the Enerhen Junction to Orhuwhorun Junction. The nodes and links are presented in Fig. 8.9.

Model Notation and Formulations

- No of links, a = 2
- No of nodes/junctions, N = 3
- The possible number of networks in this case is given as 2^n
  $2^3 = 8$ possible number of networks
- Budget constraint, B = 500,000 NGN
- Upper-level function is considered
- First, according to the flow chart, the necessary parameters are considered
- Distance by car is 22 km
- Time taken is to navigate—33 min
- Assuming the following:
  - $X_a = 0.67$ km/min
  - $T_a = 33$ min
  - Y_a (link capacity) = 50
  - d_a (demand) = 300
  - $c_a = 50{,}000$ NGN per link

  Notice that:

$$\sum c_a y_a \leq B$$

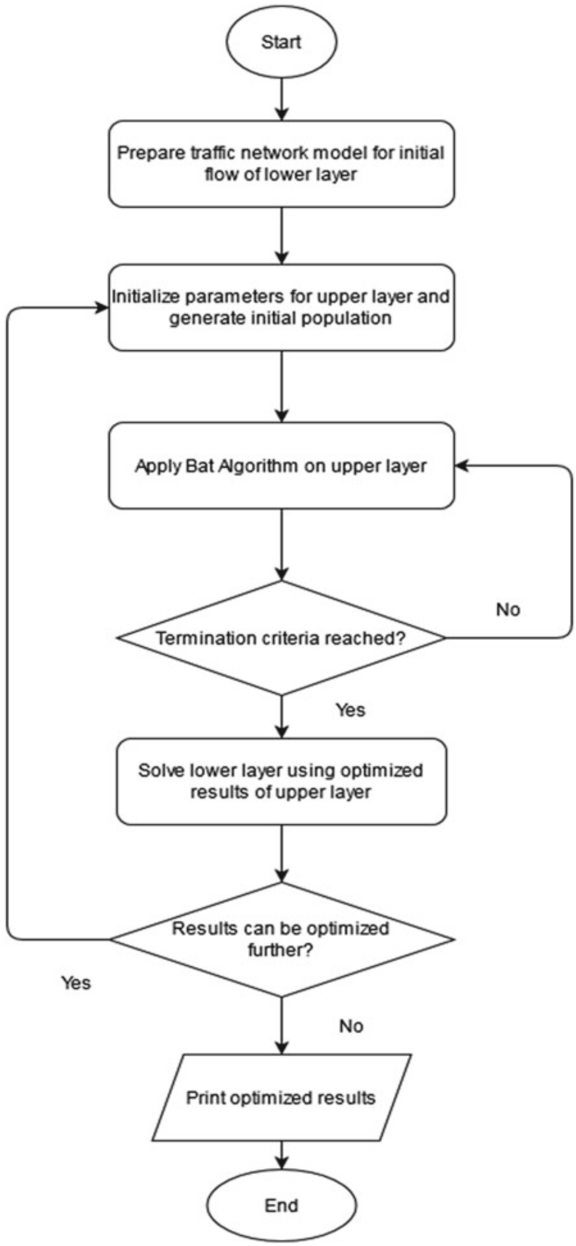
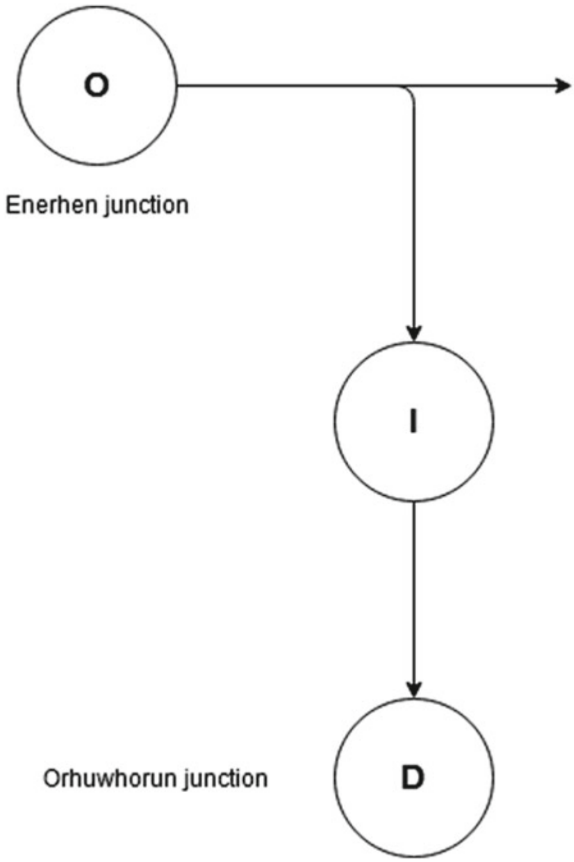


Fig. 8.8 Flow chart for the traffic signal optimization problem using bat algorithm

Fig. 8.9 Road network for three nodes and two links



$$\sum ₦50000 \times 50 = ₦2500000 \leq ₦4000000$$

Since the above condition is satisfied, we proceed;

$$\mathrm{minT(y)} = \sum_{a \in A} \sum_{a \in A(y)} x_a t_a(x_a) + d_a y_a$$

$$\sum_{a \in A} \sum_{a \in A(y)} 0.67 \times 33 \times 0.67 + (300 \times 50)$$

This gives min T(y) = 15,014.8137.
Bat algorithm is applied at the upper-level function.
Defining objective function f(x) and assigning the parameters.

- Bat population (demand), $x_i = 300$

- Velocity, $v_i = 0.67$ km/min
- Loudness, $a_i = 50{,}000$ (cost of link construction)
- Pulse rate, $r_i = 50$ (link capacity)
- Frequency, $f_i = 0.030$.

Solution Method

- Let the bats stationed for direction-finding represent the available vehicles in search space or road network. The map reading or course-plotting is made easy.
- Since the number of vehicles passing the road has been estimated to be roughly 300 periodically and the link capacity is 50 per link, then the total capacity for both links is $50 \times 2 = 100$.
- The roads mapped for the study is such that 100 vehicles can be allowed for easy navigation.
- 300 vehicles use the road frequently, a random value is assigned to the pulse rate or link capacity.
- For the first iteration, it is assumed that the link capacity or pulse rate is 70 per link. This makes a total of 140 vehicles for both links.
- According to the initially defined condition, Rand > r_i; 70 > 50 satisfies the condition.
- It is important to generate a new solution around the defined condition.
- To achieve this, assume the link capacity (pulse rate) to be 75, making the demand for both links equal to 150 vehicles.
- Since the link capacity has been increased, the time rate of flow will reduce. So that t_a equal 31 min; f_i equal 0.032 and velocity, v_i equal 0.71 km/min. The cost of construction (loudness) increases to 90,000 NGN.
- For the second iteration, assume the link capacity is 100, therefore, demand for both links equal 200 vehicles.
- Recall the established condition, Rand > r_i; 100 > 50 satisfies the condition, it is important to store and generate a new solution around the stated condition.
- Assume the link capacity to be 105, making the demand for both links equal to 210.
- Again, the time rate of flow will automatically increase as the link capacity has reduced.
- So, t_a equals 29 min; f_i equals 0.034 and v_i equals 0.76 km/min.
- The cost of construction (loudness) increases to 100,000 NGN.
- For the third iteration, let Rand equal 140, making the demand for both links equal to 280 vehicles.
- Following the established condition, Rand > r_i; 140 > 50 satisfies the stated condition. It is important to store and generate a new solution around the stated condition.
- Considering the link capacity of 145. The demand for both links becomes 290. With increase in the link capacity, the time rate of flow will increase to $t_a = 27$ min; $f_i = 0.037$ and $v_i = 0.81$ km/min. The construction cost (loudness) equals to 150,000 NGN.

- Finally, for the last iteration, Rand equals 155, and the demand for both links equals 310. Since Rand > r_i; 155 > 50. Since the condition is satisfied, a new solution is generated.
- With a pulse rate of 160, the demand for both links becomes 320 vehicles. The time flow rate increases to 25 min, at f_i of 0.040, and v_i equal 0.88 km/min. The construction cost (loudness) increases to 200,000 NGN.

Calculating the fitness of generated population, the table is presented thus:

Population/demand	Capacity links
100	50
150	75
210	105
290	145
320	155

Comparing obtained information using the condition established thus:

- Rand < a_i and $f(x_i) < f(x)$
- $a_i = 50{,}000$ and $f(x_i) = 0.030$.

For the first iteration, the random loudness was ₦90,000 which does not satisfy the condition since it is greater ₦50,000. But, $f(x_i)$, 0.030 is less than f(x) which are, 0.032, 0.034, 0.037 and 0.040, so the results are shown and ranked in the following table.

Population/demand	Capacity per link	Velocity (v_i) (km/min)	Time (t_a) (min)	Frequency (f_i)
100	50	0.67	33	0.030
150	75	0.71	31	0.032
210	105	0.76	29	0.034
290	145	0.81	27	0.037
320	155	0.88	25	0.040

At this stage, the termination criteria is reached. Optimal result is generated by creating a road with the link capacity of 155 vehicles so that a total of 320 vehicles can freely navigate through, from the origin to destination at a faster rate, 0.88 km/min and for a lesser time, 25 min. The construction of this road will only cost 200,000 NGN per link summing up to a total amount of ₦400,000 which is clearly within the assigned budget.

So having solved the problem of time, the lower-level function can be obtained by using optimized results

$$\sum_{a \in A} \int_0^{X_a} t_a(u)\,du$$

where, $t_a = 25$ min, $x_a = 0.88$ km/min.

Substituting in the above equation,

$$\sum \int_0^{0.88} 25u \cdot du$$

This results in total value of 9.68 which represent the traveller's flow. The result cannot be optimized further so that the construction budget is not exceeded.

By mimicking the bat algorithm and following the echolocation pattern of bats for easy location of prey and avoiding obstacles, a transportation network problem has been solved in Delta State, Udu road. The following parameters have been considered: population of bat (of the vehicles that passes the roads), velocity, loudness (construction cost for link), and pulse rate (link capacity), corresponding to the bat behavior. The problem of travel time has been optimized to allow uninterrupted traffic flow under un-signalized situation at Udu road. After four iterations, best solution was obtained with maximum link capacity of 160 to accommodate vehicles travelling that route on a daily basis.

References

1. Yang, X.S. 2010. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, 65–74. Berlin, Heidelberg: Springer.
2. Yang, X.S. 2011. Bat algorithm for multi-objective optimisation. *International Journal of Bio-Inspired Computation* 3 (5): 267–274.
3. Rizk-Allah, R.M., and A.E. Hassanien. 2018. New binary bat algorithm for solving 0–1 knapsack problem. *Complex and Intelligent Systems* 4 (1): 31–53.

Chapter 9
Ant Lion Optimization Algorithm

9.1 Introduction

Ant lion optimization (ALO) is an innovative optimization technique inspired by nature and was first proposed by [1]. The ALO algorithm as a biomimicry replicates the hunting pattern of the ant lions in a real-life scenario. Following the hunting nature of an ant lion to develop the algorithm is a five-step process. Ant lion has five main strategies of hunting prey. The first process often implemented by an ant lion is the random walk technique of the ants. The second has to do with the process of building traps for prey. The third process is the technique of stinging and entrapment of ants in traps. The fourth process has to do with holding down or catching prey and the last is the re-building of traps to hunt down prey. Strong and robust mathematical equations of the ALO has been developed by mimicking the behavior of the ALO in real life. This is very useful for solving problems in a diverse area of human life. The life cycle of the ant lion is shown in Fig. 9.1. Ant Lion Optimizer (ALO) can be used to solve NP-hard problems, combinatorial optimization problems like scheduling problems, transportation and transshipment problems, and other path planning problems. Ant lion algorithm is very useful in combinatorial and stochastic operations. Especially for: parallel machine scheduling, maintenance, cost-effective integrated maintenance scheduling in power systems, optimal sizing and siting of distributed generation, gear train design problem, ship propeller design, cantilever beam design problem, optimal scheduling of thermal units for small scale electrical economic power dispatch problem and many more areas of operations.

M. O. Okwu and L. K. Tartibu, *Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications*, Studies in Computational Intelligence 927, https://doi.org/10.1007/978-3-030-61111-8_9