Research and Reviews: Advancement in Robotics
Volume 2 Issue 3

Branch-and-Bound Bat Algorithm for Solving Complex Optimization Problems

Adeyemo Temitope T.[1]***, Sanusi Bashir A.***[2]***, Olowoye Adebola O.***[3]***, Olabiyisi Stephen O.***[4]***, Omidiora Elijah O.***[5]

[1]*Department of Information and Communication Tech., Faculty of Engineering, Elizade University, Ilara-Mokin Ondo State, Nigeria*

[2,3,4]*Department of Computer Science, Faculty of Computing and Informatics, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria*

[5]*Department of Computer Engineering, Faculty of Engineering, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria*

Corresponding Author

E-Mail Id: [1]*temitope.adeyemo@elizadeuniversity.edu.ng*
[2]*sanusibashiradewale90@gmail.com*
[3] *aoolowoye@pgschool.lautech.edu.ng*
[4]*soolabiyisi@lautech.edu.ng*
[5]*eoomidiora@lautech.edu.ng*

ABSTRACT

Most of the real-world problems are concerned with minimizing or maximizing some quantity so as to optimize some outcome. Optimization techniques are used to generate a desired solution to real-life problems. Bat Algorithm (BA) is an evolutionary optimization technique and it is one of the recent metaheuristic algorithms for solving optimization problems. It was inspired by the echolocation behavior of micro-bats, with varying pulse rates of emission and loudness. The major limitation of BA is that it will converge very quickly at an early stage and then convergence rate slow down thereby giving a local optimal solution but not the global optimal solution. Branch-and-bound (BnB) is a common method for improving the searching process by systematically enumerating all candidate solutions and disposing of obviously impossible solutions. This advantage of BnB algorithm will be used to improve BA. BnB was used to improve the best-found solution of BA.

Keywords: *Bat algorithm, branch-and-bound, optimization techniques*

INTRODUCTION

Optimization is the modification of a structure for a more efficient result. It involves maximizing or minimizing some function relative to some set, often representing a range of choices available in a certain situation. The function allows a comparison of different available choices for determining which might be "best" [1]. Optimization techniques are used to generate the "most desired", optima or satisfactory solution. These techniques, in combination with more specific and efficient simulation methods, are the key doors for finding the optimum (or unconstrained maxima or minima) solution of continuous or differential problems [2].

With the advent of computers, optimization has become a part of computer-aided design activities. Due to this, many approaches have been developed for the optimization of different problems. Evolutionary programming is a meta-heuristic introduced as a useful optimization tool for handling non-linear programming problems. Various modifications to this method have been

proposed with a view of enhancing convergence speed. These have been applied successfully on some benchmark mathematical problems [3]. To solve complex optimization problems (NP-hard), integration of two or more optimization techniques proffers a better solution. An even more efficient behavior and higher flexibility when dealing with real-world and large-scale problems, can be achieved through a combination of a meta-heuristic with other optimization techniques. It has been demonstrated that exact optimization techniques and meta-heuristics both have specific advantages, which complement each other. Suitable combinations of exact algorithms and meta-heuristics can benefit much from synergy and often exhibit significantly higher performance with respect to solution quality and time [4].
Branch-and-bound method can simply be defined as an exact algorithm that is used for optimization problem to get an optimal solution to the problem. This looks for the outstanding solution for a given problem in the entire space of the solution. Furthermore, the bounds in the function to be optimized are merged with the value of the latest best solution. It allows the algorithm to find parts of the solution space completely [5]. However, the purpose of a branch and bound search is to maintain the lowest-cost path to a target. Once a solution is found, it can keep improving the solution. The Bat Algorithm (BA), also known as an evolutionary optimization method is an attempt to collect positive side of existing algorithms. It provides better formulation of the variables that also gives a control on convergence rate. The Bat Algorithm has a union of major advantages of existing successful algorithms, such as Particle Swarm Optimization (PSO), Harmony Search (HS), and Simulated Annealing (SA). BA has a good capability to balance the global exploration and the local exploitation during a search process.

Combining Branch-and-Bound algorithm with Bat Algorithm will give rapid convergence to global optima solutions. BnB algorithm will be used to reduce the search space of BA by redefining the best-found solution of BA. BnB improves the best-known solution of BA thereby helping it reach global minima at a high convergence rate. Hence, this paper presents the combination of both methods by collaborating them together during implementation. The rest of this paper is organized as follow. In Section 2, we present review of related work for the proposed algorithm. Section 3 describes the problem definition while section 4 proposed the literature review. Section 4 discusses the proposed algorithm and the conclusion makes up Section 5.

REVIEW OF RELATED WORKS

It is natural hybridized Branch-and-Bound with other meta-heuristics to find a good upper bound of the optimal solution quickly either on the root node or regularly during the execution of a Branch-and-Bound algorithm. Portman *et al*. [6] use a genetic algorithm on the root node of a Branch-and Bound in order to provide a good initial upper bound to the Branch-and-Bound. Jouglet *et al*. [7] propose a similar approach but they use the genetic algorithm to provide an initial upper bound to a constraint programming method. In Basseur *et al*. [8] a bi-objective unrelated parallel machine problem is tackled with a genetic algorithm that provides an initial pareto front to a 2-phase Branch-and-Bound. Tsai *et al*. [9] in their research work proposed Evolved Bat Algorithm (EvBA) for solving numerical optimization problems based on the framework of BA. The EvBA was established with a new definition of the bats' movements. They introduced the sound speed to calculate the distances and update the bat's movements.

Ahmed [10] in his research work presents a new hybrid algorithm for solving integer-programming problems. Also, the proposed algorithm is called accelerated bat algorithm (ABATA). In ABATA, acceleration of the search process is invoked by Nelder-Mead method and Bat Algorithm used a local search technique in order to refine the best obtained solution at each iteration. Osama *et al*. [11] introduced a better version of a Bat Meta-Heuristic Algorithm, (IBACH), for solving integer programming problems in their research work. The proposed algorithm used chaotic behavior to produce a candidate solution in behaviors similar to acoustic monophony. Numerical results showed that the IBACH is able to acquire the optimal results in comparison to traditional methods (branch and bound), standard Bat algorithm, particle swarm optimization algorithm (PSO) and other harmony search algorithms.

Also, Morita and Shio [12] proposed a hybridized branch and bond technique with Genetic Algorithm (GA) for flexible flow-shop scheduling problem, which provided the precise optimal solution. The GA was used to derived an effective upper bound in the bounding scheme to realize the frequent update of current solution, which may cause the reduction of the number of nodes to be searched. Furthermore, the computational efficiency was improved compared with the conventional branch and bound method because of the reduction of the number of searched nodes.

Borchers and Mitchell [13] show some preliminary results on various combination of Branch-and-Bound and genetic algorithms: they tried using Branch-and-Bound-like methods as local search operator of a genetic algorithm leading to a heuristic hybrid method. Bacanin and Tuba [14] presented a hybrid algorithm; they used the Branch-and-Bound to find promising nodes and thus generate the initial genetic algorithm population and then used the genetic algorithm results to give hints to the Branch-and-Bound on where there can potentially be interesting solutions. Manquinho *et al.* (2007) in its survey on BA hybrid methods distinguishes between integrative combinations that tend to use one method as an operator of the other and collaborative methods. This second category was also categorized between sequential and parallel execution.

PROBLEM DEFINITION

Solving complex discrete optimization problems to optimality is often an immense job requiring very efficient algorithms, and BnB is one of the main tools used. Branch and Bound (BnB) is one of the famous exact optimization algorithms for solving complex optimization problems. However, it suffers from high complexity, since it explores a hundred of nodes in a big tree structure when solving a large-scale problem [15]. A BnB algorithm searches the complete space of solutions of a given problem for the best-fit solution. However, explicit enumeration is normally impossible due to the exponentially increasing number of potential solutions. Bat algorithm (BA) is a promising nature inspired algorithm inspired from the echolocation behavior of the micro-bats but BA converge very quickly at the early stage and then convergence rate slow down, thus getting trapped in local minima [10].

Complex optimization problems can be tackled by means of exact algorithms like BnB as well as by means of meta-heuristic methods like BA. Nevertheless, the exact techniques give us a guarantee of optimality while meta-heuristic methods do not. On the other hand, the metaheuristic methods can handle large and complex optimization problems while exact methods tend to fail as the size of the optimization problem increases. Thus, it is good to combine these two strategies to obtain better solutions to the problem that is being addressed.

Optimization problems are ubiquitous in science and engineering, as well as daily life routine. For instance, timetabling problems in various educational or organizational sectors, the choice of line we stand in at the banks or supermarket or deciding how patients get to see the doctor on arrival. In general, an optimization problem at the mathematical level is given by an objective function $f\colon\ K \to R$, and one searches for $x^* \in K$, such that $f(x^*) \leq f(x)$ for all $x \in K$. However, K is a subset of R^n defined by constraints [16]. Hence, the optimization problem is formulated as;

$$\begin{aligned} f(x) &\to \min & x &\in R^n, \\ c_j(x) &\geq 0 & j &\in I, \\ c_j(x) &= 0 & j &\in E, \end{aligned} \tag{1}$$

where I is an index set denoting the inequality constraints and E is an index set denoting the equality constraints of the problem, whose sizes is denoted by $m_I = |I|$ and $m_E = |E|$ and $c_j : R^n \to R$ are the constraints.

LITERATURE REVIEW

Branch-and-Bound

Consider a general combinatorial optimization problem like

z = max _$f(x)$: $x \in S$_ (2)

Branch-and-bound is a divide and conquer strategy, which decomposes the problem to sub-problems over a *tree* structure, which is referred to as *branch-and-bound tree*. The decomposition works based on a simple idea: If S is decomposed into $S1$, $S2$, $S3$ and $S4$ such that $S = S1 \cup S2 \cup S3 \cup S4$, and we define sub-problems zk to be

$zk = \max \{f(x): x \in Sk\}$ for $k = 1, 2,$
then $z = \max_k zk$ (3)

Each sub-problem denotes a *node* on the tree. Figure 1 shows a schematic of a branch-and-bound tree. The main problem with problem S is at the *root node*, and is then divided into four sub-problems $S1$, $S2$, $S3$ and $S4$, where we have $S1 \cup S2 \cup S3 \cup S4 = S$. The process of dividing a node sub-problem into smaller sub-problems is called *branching* and sub-problems $S1$, $S2$, $S3$ and $S4$, are called *branches* created at node S. In Figure 1 sub-problem $S2$ and $S3$ is further branched into smaller sub-problems. The branching does not necessarily have to be two-way, and multiway branching is also possible. Note that branching indefinitely will only result in explicit enumeration of the feasible region of S.

Therefore, to avoid this, the branch-and bound algorithm handles this problem by *bounding* and *pruning*. Bounding involves ignoring partial solutions that cannot be better than the current best solution by setting a bound on the solution quality. To this end, lower and upper bounds are maintained. Pruning can be defined as trimming off the branches in the solution tree whose solution attribute is estimated to be poor. Bounding and pruning are the key concepts of the branch-and bound procedure, because they are used to effectively reduce the algorithm search space. In an optimization problem, a lower bound is a solution which is less than or equal to every feasible solution in the search space.

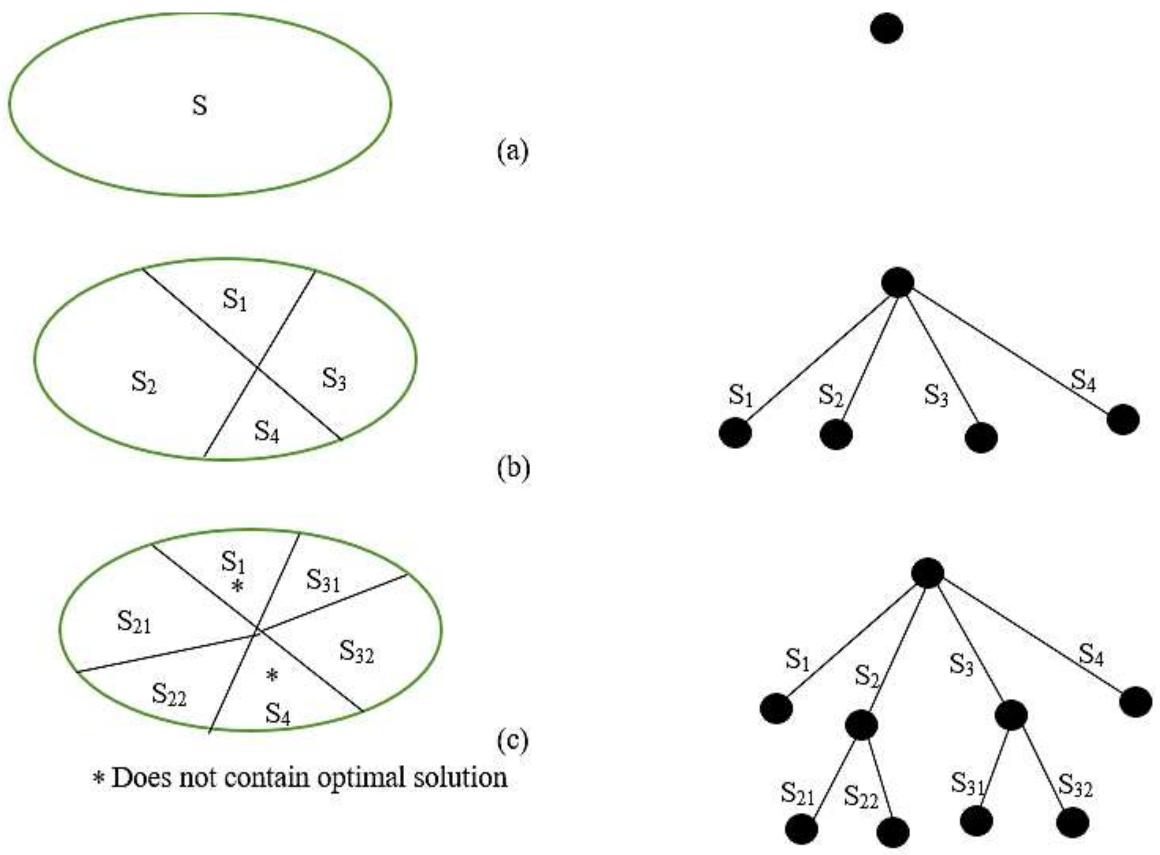

***Fig.1:** Illustration of the Search Space of BnB. [17]*

Bat Algorithm

In the Bat Algorithm, at the iteration t, each of the bat in the population changes position randomly with a velocity vit and a position xit. The location of each bat (solution) is assessed by measuring its fitness function value (xi) and the overall best solution $x*$ is assigned according to this value. The location xi and the velocity vi for each solution in the population are modified as shown in Equations 4, 5 and 6. The Bat Algorithm is regarded as a frequency-tuning algorithm, since each bat is randomly assigned a frequency$f$, $f\in$ [, $fmax$]. The frequency parameter is very important to balance between the exploration and the exploitation procedures in Bat Algorithm. Also, in the initial population, each bat is randomly assigned a frequency then these values are adjusted as shown in Equation 4 [18].

$$f_i = f_{min} + (f_{max} - f_{\min})\, \beta \qquad (4)$$

Where $\beta \in [0,1]$ is a random vector drawn from a uniform distribution. The initial velocity and position of each bat in the population are assigned randomly then they are updated as shown in Equations 5 and 6 respectively.

$$v_i^t = {}^{(t-1)} + ({}^{(t-1)} - x^*)\, f_i \qquad (5)$$

where x^* is said to be the best solution in the population.

$$x_i^t = {}^{(t-1)} + v_i^t \qquad (6)$$

In addition, the loudness parameter A_i and the pulse emission rate parameter r_i are very important parameters in a bat algorithm, since they control and switch between inspection and utilization procedure during the search. The loudness decreases once a bat has establish its prey, while the pulse emission rate increases. The values of loudness have to vary between A_{max} and A_{min}, when A_{min}=0 means that a bat has establish the prey. The value of loudness can be updated during the search as follow.

$^{(t+1)} = \propto A_i{}^t$ (7)

Where $\propto$ is a constant and has the same effect as the cooling factor in simulated annealing algorithm. In addition, the pulse emission rate parameter can be updated as follow.

$r_i{}^t = r_i{}^0[1 - e^{(-\gamma t)}]$ (8)

Where γ is a constant and $\gamma > 0$.

Each bat (solution) in the population is evaluated by calculating its fitness function value and the overall best solution is selected as a current best solution $x*$. Once the best solution is selected, each bat in the population generates a new solution by using a random walk method as follow:

$x_{\text{best}} = x_{old} + \epsilon\, A^t$

Where ϵ is a random number and $\epsilon \in [-1,1]$, $At = <Ait>$ is the average loudness of all the bats at the current iteration.

Algorithm 1: Pseudocode of a Basic Bat Algorithm (BA) [19]

```
Begin
Initialize Bat population: $Xi$ ($i$ = 1, 2, ..., $n$), minimum frequency $f_{min}$, maximum frequency $f_{max}$, loudness constant $\alpha$, pulse rate constant $\gamma$, initial loudness $A_0$, minimum loudness $Amin$, initial pulse rate $r_0$ and maximum iterations $Max_{itr}$.
Set initial values of pulse rates $ri$ and loudness $Ai$.
While iteration $t < Max_{itr}$ do
Generate the initial bat population $x_{it}$ and velocities $v_{it}$ randomly.
Assign the initial frequency $f_i$ to each $x_{it}$.
Evaluate the initial population by calculating the objective function ($xit$) for each solution in the population.
End While
Generate new bat solutions $x_{it}$ by adjusting the frequency $f_i$
Update the bat velocities $v_{it}$
Evaluate the new population by calculating the objective function $f(x_{it})$ for each solution in the population
Select the best solution $x^*$ from the population.
 IF $rand > ri$ Then
Select a solution among the best solutions
Generate a local search solution around the selected best solution
End IF
Generate a random new solution
IF $rand < Ai$ & $f((xit) < f(x*))$ Then
Accept the new solutions.
Increase pulse emission $ri$ and reduce loudness $A_0$
End IF
Rank the current best solution $x_{\text{best}}$ and rank the bat
Post-processing the results and visualization
End
```

PROPOSED METHODOLOGY

In this proposed algorithm, we merge the Bat Algorithm, which has a good capability of exploring the search space and the branch-and-bound algorithm, which is one of the most important exact algorithms and also has a powerful performance as a local search technique. This proposed algorithm starts with initial population that is generated randomly. Then bat solutions are updated by moving randomly with a velocity vit and a position xit. Afterwards, each solution is measured by calculating its fitness function and the best solution is selected from the bat population according to its fitness function value. BnB is incorporate to select a local solution among the best solutions generated. It starts from a list of all best solution generated with respect to their weights.

Let s_{os} be the set of best solutions which will be evaluated, b_{sol} be the best evaluated solution for the objective function, x_{best} be the best solution founded and l_b be a lower bound for the solution. The workflow and pseudo code of the proposed algorithm is presented in Figure 2 and Algorithm 2 respectively.

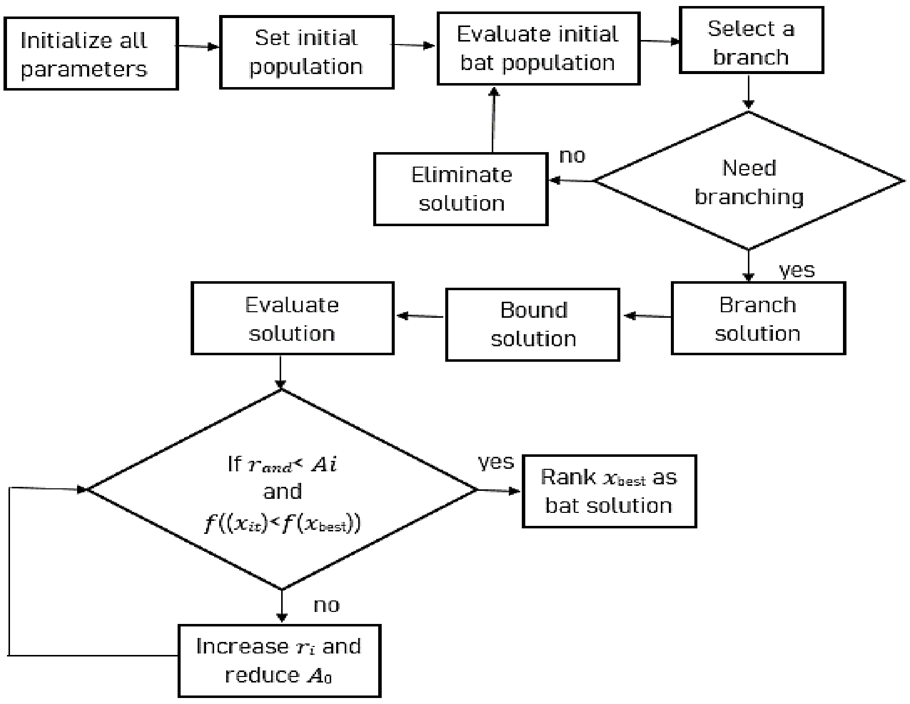
Initialize all parameters
Set initial population
Evaluate initial bat population
Select a branch
Need branching
no
Eliminate solution
yes
Branch solution
Bound solution
Evaluate solution
If r_{and}< Ai and $f((x_{it})$<$f(x_{best}))$
yes
Rank x_{best} as bat solution
no
Increase r_i and reduce A_0

Fig. 2: *Flowchart of the Proposed Algorithm.*

Algorithm 2: Pseudocode for the Proposed Algorithm

```
Begin
Initialize Bat population: $Xi$ $(i = 1, 2, ..., n)$, minimum frequency $f_{min}$, maximum frequency $f_{max}$, loudness constant $\alpha$, pulse rate constant $\gamma$, initial loudness $A_0$, set of best solutions $s_{os}$, best evaluated solution $b_{sol}$, minimum loudness $Amin$, initial pulse rate $r_0$ and maximum iterations $Max_{itr}$, $x_{\text{best}}$ be the best solution founded and $l_b$ be the lower bound for $x^*$.
Set initial values of pulse rates $ri$ and loudness $Ai$.
While iteration $t< Max_{itr}$ do
Generate the initial bat population $x_{it}$ and velocities $v_{it}$ randomly.
Assign the initial frequency $f_i$ to each $x_{it}$.
Evaluate the initial population by calculating the objective function $(xit)$ for each solution in the population.
End While
Generate new bat solutions $x_{it}$ by adjusting the frequency $f_i$
Update the bat velocities $v_{it}$
Evaluate the new population by calculating the objective function $f(x_{it})$
Select the best solution $x^*$ from the population.
 If $r_{and}> r_i$ then
Choose a branching node $k$ from $s_{os}$
Remove node $k$ from new population
Generate a solution from $k$
Generate $l_b$ from the generated solution
For i=1 to n do
If $l_{bi}$ is worse than $x*$ then
Eliminate solution;
Else
$b_{sol} \leftarrow l_{bi}$
$x_{\mathrm{best}} \leftarrow$solution i
Else
Add solution i to $s_{os}$
End For
End If
If $r_{and} < A_i$ & $f((x_{it}) < f(x_{\mathrm{best}}))$ then
Accept the new solutions.
Else
Increase pulse emission $ri$ and reduce loudness $A_0$
End If
Rank $x_{\mathrm{best}}$ as the bat solution
Post-processing result
End
```

CONCLUSION

In conclusion, the problems in engineering design, economics, management, data analysis and statistics can often be defined as mathematical optimization problems. The different techniques exist to take into account multiple objectives or uncertainty in the data. Having presented Branch and Bound Bat Algorithm (BBBA) for optimization problem in this paper, it is concluded that the proposed algorithms can be apply to solve complex optimization problems like job machine scheduling (job shop, open shop and more than three machine flow shop problems), supply chain network problem, electricity distribution problem and other control of non-linear dynamical systems.

REFERENCES

1. Attia S., Hamdy M., O'Brien W., Carlucci S. *Assessing gaps and needs for integrating building performance optimization tools in net zero energy buildings design*. Energy and Buildings. 60, 110-124p, 2013.
2. Ranut P. *Optimization and inverse problems in heat transfer* (Doctoral dissertation, Ph. D. thesis, University of Udine). 2012.
3. Sinha N., Chakrabarti R., Chattopadhyay P.K. *Evolutionary programming techniques for economic load dispatch.* IEEE Transactions on evolutionary computation. 7(1), 83-94p, 2003.
4. Roeva O.N., Fidanova S.S. *Hybrid bat algorithm for parameter identification of an e. coli cultivation process model.* Biotechnology & Biotechnological Equipment. 27(6), 4323-4326p, 2013.
5. Puchinger J., Raidl G.R. *Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification*. In International Work-Conference on the Interplay Between Natural and Artificial Computation. Springer, Berlin, Heidelberg, 41-53p, 2005.
6. Portmann M.C., Vignier A., Dardilhac D., Dezalay D. *Branch and bound crossed with GA to solve hybrid flowshops.* European Journal of Operational Research. 107(2), 389-400p, 1998.
7. Jourdan L., Basseur M., Talbi E.G. *Hybridizing exact methods and metaheuristics: taxonomy*. European

Journal of Operational Research. 199(3), 620-629p, 2009.

8. Basseur M., Seynhaeve F., Talbi E.G. Path relinking in pareto multi-objective genetic algorithms. In International Conference on Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg. 120-134p, 2005, March
9. Tsai P.W., Pan J.S., Liao B.Y., *et al. Bat Algorithm Inspired Algorithm for Solving Numerical Optimization Problems.* Applied Mechanics and Materials. 37(1), 148-149p, 2011. doi: 10.4028/www.scientific.net/AMM.148-149.134, 2011.
10. Fouad A. *Bat Algorithm for Solving Integer Programming Problems.* Egyptian Computer Science Journal. 39(1), 25-40p, 2015.
11. Osama A., Mohamed A., Ibrahim E. *An Improved Chaotic Bat Algorithm for Solving Integer Programming Problems.* International Journal of Modern Education and Computer Science. 8(2), 18-24p, 2014. DOI: 10.5815/ijmecs.2014.08.03
12. Morita, Shio. *Hybrid branch and bond method with genetic algorithm for flexible flowshop scheduling problem*. Journal of Advanced Mechanical Design, System and Manufacturing. International Journal series C. 18(1), 46-52p, 2005.
13. Borchers B., Mitchell J. *Using an Interior Point Method in a Branch and Bound. Algorithm for Integer Programming*. Technical Report, Rensselaer Polytechnic Institute, Troy, New-York, USA. 2002.
14. Bacanin N., Tuba M. *Artificial Bee Colony (ABC) Algorithm for Constrained Optimization Improved with Genetic Operators.* Studies in Informatics and Control. 21(2), 137-146p, 2012.
15. Jovanovic R., Tuba M. *Ant Colony Optimization Algorithm with Pheromone Correction Strategy for Minimum Connected Dominating Set Problem.* Computer Science and Information Systems (ComSIS). 9(4), 276-285p, 2012.
16. Parkinson A.R., Balling R.J., Hedengren J.D. *Optimization Methods for Engineering Design: Applications and Theory.* Doctoral dissertation, Ph. D. thesis submitted to Brigham Young University. 2013
17. Manquinho V., Marques J., Oliveira S., Sakallah K. *Branch and Bound Algorithms for Highly Constrained Integer Programs.* Technical Report, Cadence European Laboratories, Portugal. 2007
18. Yang X.S., He X. *Bat Algorithm: Literature review and applications.* International journal of biological inspired Computation. 5(3), 141-149p, 2013.
19. Induja S., Eswaramurthy V. *Bat Algorithm: An Overview and its Applications.* International Journal of Advanced Research in Computer and Communication Engineering. 5(1), 76-90p, 2016.