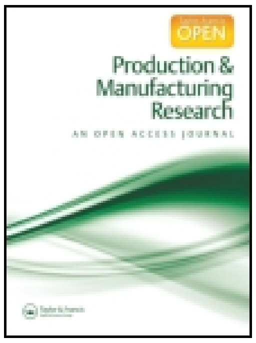
An enhancement to the Bees Algorithm with slope angle computation and Hill Climbing Algorithm and its applications on scheduling and continuous-type optimisation problem

Yuce, B. , Pham, D. T. , Packianather, M. S. and Mastrocinque, E.



Production & Manufacturing Research
An Open Access Journal

ISSN: (Print) 2169-3277 (Online) Journal homepage: http://www.tandfonline.com/loi/tpmr20

An enhancement to the Bees Algorithm with slope angle computation and Hill Climbing Algorithm and its applications on scheduling and continuous-type optimisation problem

B. Yuce, D.T. Pham, M.S. Packianather & E. Mastrocinque

Production & Manufacturing Research: An Open Access Journal, 2015
Vol. 3, No. 1, 3–19, http://dx.doi.org/10.1080/21693277.2014.976321

An enhancement to the Bees Algorithm with slope angle computation and Hill Climbing Algorithm and its applications on scheduling and continuous-type optimisation problem

B. Yuce[a]*, D.T. Pham[b], M.S. Packianather[c] and E. Mastrocinque[d]

[a]*School of Engineering, BRE Institute of Sustainable Construction, Cardiff University, Newport Road, The Parade, Queen's Building, Cardiff CF24 3AA, UK;* [b]*School of Mechanical Engineering, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK;* [c]*Institute of Mechanical and Manufacturing Engineering, Cardiff University, Queen's Buildings, The Parade, Cardiff CF24 3AA, UK;* [d]*School of Management, Royal Holloway, University of London, Egham Hill, Egham, Surrey TW20 0EX, UK*

(*Received 3 November 2013; accepted 9 October 2014*)

This paper focuses on improvements to the Bees Algorithm (BA) with slope angle computation and Hill Climbing Algorithm (SACHCA) during the local search process. First, the SAC was employed to determine the inclination of the current sites. Second, according to the slope angle, the HCA was utilised to guide the algorithm to converge to the local optima. This enabled the global optimum of the given problem to be found faster and more precisely by focusing on finding the available local optima first before turning the attention on the global optimum. The proposed enhancements to the BA have been tested on continuous-type benchmark functions and compared with other optimisation techniques. The results show that the proposed algorithm performed better than other algorithms on most of the benchmark functions. The enhanced BA performs better than the basic BA, in particular on higher dimensional and complex optimisation problems. Finally, the proposed algorithm has been used to solve the single machine scheduling problem and the results show that the proposed SAC and HCA-BA outperformed the basic BA in almost all the considered instances, in particular when the complexity of the problem increases.

Keywords: the Bees Algorithm; slope angle computation; Hill Climbing Algorithm; benchmark functions; single machine scheduling

1. Introduction

Optimisation algorithms play a big role in the area of operational research, artificial intelligence, mathematics, computer science, robotics and many other areas, and are utilised to find the optimum solution for highly complicated problems. Several optimisation algorithms have been proposed to solve a variety of problems. Many real-life problems are not easy and are hard to solve in polynomial time which are called NP-Hard (Woeginger, 2003). NP-Hard-type problems are generally solved with stochastic-based optimisation algorithms such as genetic algorithm, ant colony optimisation, tabu search, simulated annealing and the Bees Algorithm (BA). Every algorithm has advantages and disadvantages depending on the type of problem which it is solving. Although there is no one best algorithm to solve all types of optimisation problems (Wolpert & Macready, 1997), the

*Corresponding author. Email: yuceb@cardiff.ac.uk

BA has been used successfully to solve many problems including mechanical design, job shop scheduling, supply chain optimisation, robot path planning, chemical engineering problems, assembly problems and several other applications (Ang, Ng, & Pham, 2013; Ang, Ng, Pham, & Soroka, 2013; Castellani, Pham, & Pham, 2012; Lien & Cheng, 2014; Mastrocinque, Yuce, Lambiase, & Packianather, 2013; Pham, Castellani, & Fahmy, 2008; Pham, Afify, & Koç, 2007; Pham, Castellani, & Ghanbarzadeh, 2007; Pham, Darwish, Eldukhri, & Otri, 2007; Pham, Koc, Lee, & Phrueksanant, 2007; Pham, Otri, Afify, Mahmuddin, & Al-Jabbouli, 2007; Pham, Soroka, Koç, Ghanbarzadeh, & Otri, 2007; Yuce, Mastrocinque, Lambiase, Packianather, & Pham, 2014). Despite these several successful applications of the BA, a careful examination shows that the algorithm needs to be more robust and sensitive to a particular problem. Several attempts have been made in the past to improve the BA using methods such as early neighbourhood search and efficiency-based recruitment (Pham, Packianther, Imaguliyev, & Yuce, 2012), adaptive neighbourhood search and site abandonment strategy (Yuce, Packianather, Mastrocinque, Pham, & Lambiase, 2013), combination of adaptive enlargement and reduction of the search neighbourhood (Azfanizam, Pham, & Faieza, 2014) developing new local search strategies (Ahmad, 2012), tuning the algorithms parameters (Otri, 2011), introducing new parameters (Pham & Ghanbarzadeh, 2007), combining with other optimisation algorithms (Sholedolu, 2009) novel initialisation based on the patch concept and Levy flight distribution (Hussein, Sahran, & Abdullah, 2014). In this paper, another such improvement is proposed to address the insensitive movement of the BA based on a determination of promising sites and convergence to the related sites with deterministic approach using a hybrid BA combining with hill climbing. The first step is to define a measurement technique to determine the direction along which promising solutions can be found. This is based on the steepness angle mimicking the direction along which a scout bee performs its figure-of-eight waggle dance during the recruitment of forager bees. The second step is to develop a hybrid algorithm combining BA and a Hill Climbing Algorithm (HCA) based on the threshold value of the steepness angle. The proposed enhanced BA has been tested on continuous-type benchmark functions and compared with other optimisation techniques, and finally has been applied to solve a well-known manufacturing problem such as the single machine scheduling problem, in order to show its benefit in solving a real-world problem.

This paper is organised as follows: basic version of the BA is presented in Section 2. The improved version of the BA follows in Section 3. The algorithm tests conducted on benchmark functions and results are given in Section 4. The single machine scheduling problem, its solution by the proposed algorithm and results are discussed in Section 5. Finally, the conclusions are given in Section 6.

2. The basic BA

The BA is a type of swarm-based algorithm, which was proposed by Pham, Ghanbarzadeh, Koc, Otri, Rahim, and Zaidi (2005, 2006). Swarm-based algorithms mimic the swarm behaviour of animals in nature such as ant colonies, bee colonies, bird flocking, animal herding, bacterial growth, fish schooling and many other (Beni & Wang, 1989). These behaviours have been implemented in artificial intelligence and swarm intelligence.

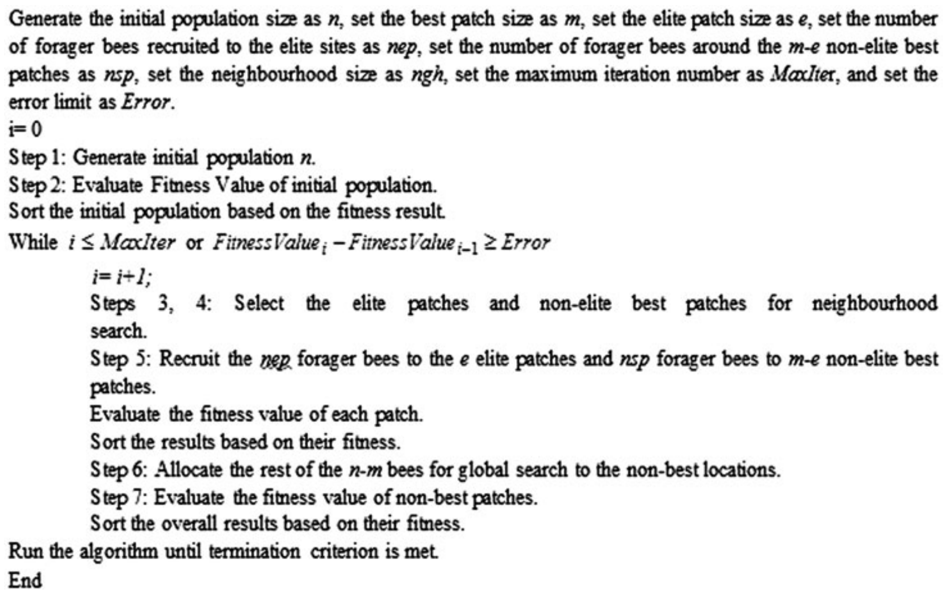
BA is also a type of swarm intelligence which has been inspired from foraging behaviour of the honey bees. A swarm of honey bees consists of a queen bee and thousands of worker bees which are allocated to do different works in a colony such as cleaning their nest, foraging and constructing comb (Seeley, 1995). In the nature when a scout bee (forager) finds a food source (nectar), it returns to the hive and performs special ritual which is a communicational movement, known as the waggle dance; this informs the colony about the nectar (Gould & Gould, 1988; Seeley, 1995). The waggle dance contains information about direction, distance and quality of the flower patch found by the bee (Talbi, 2009). After the waggle dance, the colony decides the number of bees needed to be assigned for that particular food source as promising patches should have more bees than less promising ones. The recruited bees evaluate the related patch and look around of the patch as well as share this information with their peers in the hive. The same process happens for all patches emulating local search. After the recruitment stage, the scout bees will continue searching for other promising patches in a random way emulating the global search and whenever they find another patch, this information will be given to the colony randomly (Von Frisch, 1955). This behaviour of honey bees was computationally modelled as an optimisation algorithm. The parameters and the pseudo-code of the algorithm are given below in Table 1 and Figure 1, respectively.

According to Figure 1, the BA has following steps: the first step is placing the ‘*n*’ scout bees on the search space, and then in the next step, fitness values of the visited patches are evaluated. Step 3 is related to the selection of the best patches with respect to their fitness value where these selected best patches will be split into two groups containing more scout bees to the elite patches ‘*e*’, and less scout bees to the non-elite best patches ‘*m–e*’ in step 4. The next step 5 covers the neighbourhood search in the patches given in steps 3 and 4, and so according to neighbourhood search, the patches’ fitness values will be evaluated. In step 6, the remainder bees, which are created in initial population, will be recruited for the random search to find better random solutions. In step 7, the random patches’ fitness values will be evaluated and this process will continue until one of the stopping criteria is met.

The procedure described above equips the algorithm to combine exploitative neighbourhood search with explorative global search enabling effective location of the globally optimal solution to a problem (Yuce, 2012). Due to the powerful search capability of the algorithm, the basic BA was successfully utilised to solve different types of optimisation problems such as job scheduling (Pham, Koc et al., 2007), forming manufacturing cells (Pham, Afify et al., 2007), data clustering (Pham, Otri et al., 2007), tuning a fuzzy logic controller (Pham, Darwish et al., 2007), training neural networks (Pham, Koc, Ghanbarzadeh, & Otri, 2006), finding the multiple feasible solutions to preliminary design problems (Pham, Castellani et al., 2007) and design of mechanical

Table 1. The parameter for the basic BA.

Parameter	Symbols
Number of scout bees in the selected patches	*n*
Number of best patches out of the selected patches	*m*
Number of elite patches out of the selected best patches	*e*
Number of recruited bees in the elite patches	*nep*
Number of recruited bees in the non-elite best patches	*nsp*
The size of neighbourhood for each patch	*ngh*
Number of iterations	*MaxIter*
Difference between fitness values of consecutive iterations	*Error*



```
Generate the initial population size as $n$, set the best patch size as $m$, set the elite patch size as $e$, set the number
of forager bees recruited to the elite sites as $nep$, set the number of forager bees around the $m$-$e$ non-elite best
patches as $nsp$, set the neighbourhood size as $ngh$, set the maximum iteration number as $MaxIter$, and set the
error limit as $Error$.
i= 0
Step 1: Generate initial population $n$.
Step 2: Evaluate Fitness Value of initial population.
Sort the initial population based on the fitness result.
While $i \le MaxIter$ or $FitnessValue_i - FitnessValue_{i-1} \ge Error$
        $i= i+1;$
        Steps 3, 4: Select the elite patches and non-elite best patches for neighbourhood
        search.
        Step 5: Recruit the $nep$ forager bees to the $e$ elite patches and $nsp$ forager bees to $m$-$e$ non-elite best
        patches.
        Evaluate the fitness value of each patch.
        Sort the results based on their fitness.
        Step 6: Allocate the rest of the $n$-$m$ bees for global search to the non-best locations.
        Step 7: Evaluate the fitness value of non-best patches.
        Sort the overall results based on their fitness.
Run the algorithm until termination criterion is met.
End
```

Figure 1. Pseudo-code for the basic BA.

components (Pham, Soroka et al., 2007). The improvements made to the basic BA with slope angle computation and Hill Climbing Algorithm (SACHCA) is given in the following section.

3. The BA with SACHCA

This study presents a new enhancement to the basic BA based on the SACHCA. Even though the BA has both local and global search capability, it still has some weakness such as a high level of randomness, computational time and blind search in the local search process (Yuce, 2012). The HCA is an iterative single element-based local search algorithm, also known as gradient ascent/descent algorithm. Although the local minimum of an optimisation problem can be found by the HCA, the global optimum is not guaranteed (Grosan & Abraham, 2011).

The SACHCA-based improved BA is concerned with the process of locating the best sites. SAC is employed to determine the inclination of the current sites. In general, the promising location will be far from the current position if the slope angle is close to 90º. The current location will be close to a local optimum point if the slope angle is close to 0º. The direction of the local optimum can be determined according to the slope angle orientation.

The slope angle is computed using the first-order numerical derivation. The numerical derivation of each site is calculated from its neighbourhood. The two end points in the vicinity of the neighbourhood are used to compute the numerical derivation. The central difference method is utilised for numerical derivation (see Equation (1)).

$$\text{Slope-angle} = F'(X) = \frac{F(X + \frac{\Delta X}{2}) - F(X - \frac{\Delta X}{2})}{\Delta X} \tag{1}$$

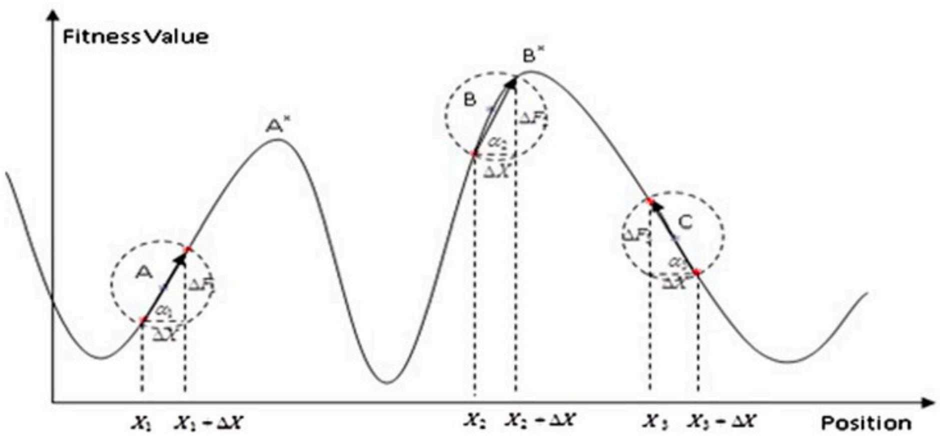


Figure 2. Slope angle for each of the best selected sites.

If the slope angle is very steep, then the promised location is far from the selected site but when the slope angle is close to zero, the promised location is very close to the selected site, as shown in Figure 2.

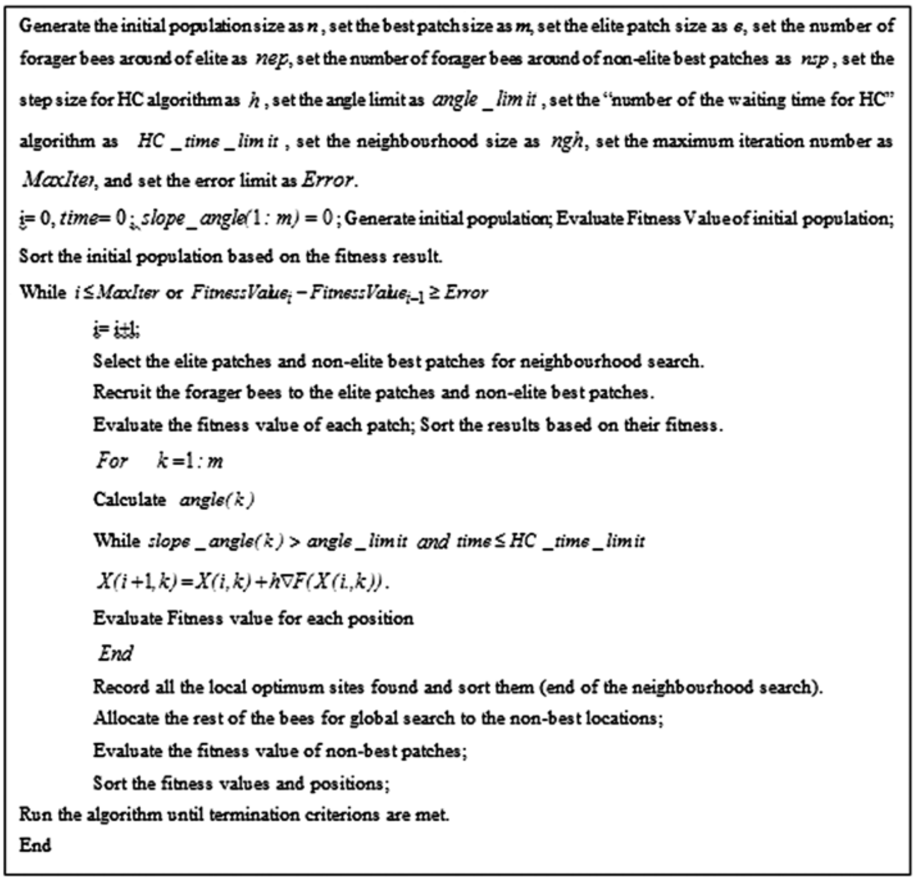
There are three selected (best) sites shown in Figure 2. Around site *A*, the angle direction is towards site A^x, which is a local optimum. The direction of sites *B* and *C* are towards the site B^x, which is another local optimum. At the end of the searching process, all the local optima will be sorted and the biggest will be selected as the global optimum. In the case of Figure 2, the site B^x was selected as global optimum. The local search process was accomplished with the use of the HCA, as shown in Equation (2).

$$X(i+1) = X(i) + h\nabla F(X_i) \tag{2}$$

where i is the iteration number, $X(i)$ is the current position, $X(i+1)$ is the next position, h is the incremental size and $\nabla F(X_i)$ is the gradient of the current position. Further, the pseudocode of the algorithm is given in Figure 3.

4. Benchmark functions tests

To measure the performance of the proposed algorithm, it was tested on 10 continuous-type benchmark functions. These functions are given in Table 2 (Ahmad, 2012; Pham & Castellani, 2009; Pham, Packianther, Imaguliyev, & Yuce, 2012). The BA requires a number of parameters to be set manually as given in Table 1. The other parameters of the proposed algorithm are given in Table 3. Several experiments were conducted in order to tune the algorithm parameters and to find the combination of them giving the best trade-off between the number of iterations and optimum that was found. All the experiments were carried out with the maximum number of iterations fixed to 5000 and $ngh = 1$. Five different combinations of the BA parameters were tested on benchmark functions according to Table 4. The tests were repeated 100 times for each function and the average number of iterations and the average optimum results obtained with the SACHCA-BA are given in Table 5. The combination number 3 gave the best results in terms of number of iterations and optimum found, and for this reason, it has been



```
Generate the initial population size as $n$, set the best patch size as $m$, set the elite patch size as $e$, set the number of forager bees around of elite as $nep$, set the number of forager bees around of non-elite best patches as $nsp$, set the step size for HC algorithm as $h$, set the angle limit as $angle\_limit$, set the "number of the waiting time for HC" algorithm as $HC\_time\_limit$, set the neighbourhood size as $ngh$, set the maximum iteration number as $MaxIter$, and set the error limit as $Error$.
$i = 0$, $time = 0$; $slope\_angle(1:m) = 0$; Generate initial population; Evaluate Fitness Value of initial population;
Sort the initial population based on the fitness result.
While $i \leq MaxIter$ or $FitnessValue_i - FitnessValue_{i-1} \geq Error$
        $i = i+1$;
        Select the elite patches and non-elite best patches for neighbourhood search.
        Recruit the forager bees to the elite patches and non-elite best patches.
        Evaluate the fitness value of each patch; Sort the results based on their fitness.
        For $k = 1:m$
        Calculate $angle(k)$
        While $slope\_angle(k) > angle\_limit$ and $time \leq HC\_time\_limit$
        $X(i+1,k) = X(i,k) + h\nabla F(X(i,k))$.
        Evaluate Fitness value for each position
        End
        Record all the local optimum sites found and sort them (end of the neighbourhood search).
        Allocate the rest of the bees for global search to the non-best locations;
        Evaluate the fitness value of non-best patches;
        Sort the fitness values and positions;
Run the algorithm until termination criterions are met.
End
```

Figure 3. Pseudo-code of improved BA with SACHCA strategy.

selected for the successive experiments. Then, the results were utilised for further analysis and comparison.

The first comparison were carried out between the proposed technique, the basic BA and other well-known optimisation techniques such as particle swarm optimisation (PSO), evolutionary algorithm (EA) and artificial bee colony (ABC) whose parameters values can be found in Table 6 (Ahmad, 2012). The performance of the algorithm was assessed according to the accuracy and the average evaluation numbers (Tables 7 and 8). Experimental results for BA, PSO, EA and ABC were extracted from another source (Ahmad, 2012).

Further, *T*-test was utilised to measure the statistical significance of the proposed algorithm and the basic BA and results are given in Table 9.

4.1. Discussion the results of benchmark functions

In this study, the neighbourhood search in the BA was identified as the focus for improving the BA by utilising SACHCA. The accuracy of the proposed algorithm was computed with average absolute differences of the best results. According to this, the

Table 2. Test functions (Ahmad, 2012; Pham & Castellani, 2009).

No	Function name	Interval	Function	Global optimum
1	Goldstein & Price (2D)	[−2, 2]	$\min F = [1 + (X_1 + X_2 + 1)^2(19 - 14X_1 + 3X_1^2 - 14X_2 + 6X_1X_2 + 3X_2^2)]$ $[30 + (2X_1 - 3X_2)^2(18 - 32X_1 + 12X_1^2 + 48X_2 - 36X_1X_2 + 27X_2^2)]$	$X = [0, -1]$ $F(X) = 3$
2	Schwefel (2D)	[−500, 500]	$\min F = \sum_{i=1}^{2} [-X_1 \sin(\sqrt{\lvert X_i \rvert})]$	$X = [0, 0]$ $F(X) = -837.658$
3	Schaffer (2D)	[−100, 100]	$\min F = 0.5 + \frac{(\sin(\sqrt{X_1^2+X_2^2}))^2 - 0.5}{1+0.001(X_1^2+X_2^2)^2}$	$X = (0, 0)$ $F(X) = 0$
4	Rosenbrock (10D)	[−1.2, 1.2]	$\min F = \sum_{i=1}^{10} [100(X_i^2 - X_{i+1})^2 + (1 - X_i)^2]$	$X = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$ $F(X) = 0$
5	Sphere (10D)	[−5.12, 5.12]	$\min F = \sum_{i=1}^{10} X_i^2$	$X = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$ $F(X) = 0$
6	Ackley (10D)	[−32, 32]	$\min F = -20e^{-0.2\sqrt{\frac{\sum_{i=1}^{10} X_i^2}{10}}} - e^{\frac{\sum_{i=1}^{10} \cos(2\pi X_i)}{10}} + 20 + e$	$X = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$ $F(X) = 0$
7	Rastrigin (10D)	[−5.12, 5.12]	$\min F = 100 + \sum_{i=1}^{10} (X_i^2 - 10\cos(2\pi X_i))$	$X = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$ $F(X) = 0$
8	Martin & Gaddy (2D)	[0, 10]	$\min F = (X_1 - X_2)^2 + (\frac{X_1+X_2-10}{3})^2$	$X = [5, 5]$ $F(X) = 0$
9	Easom (2D)	[−100, 100]	$\min F = -\cos(X_1)\cos(X_2)e^{-((X_1-\pi)^2+(X_2-\pi)^2)}$	$X = [\pi, \pi]$ $F(X) = -1$
10	Griewank (10D)	[−600, 600]	$\min F = \frac{1}{4000}\sum_{i=0}^{i=10} (x_i - 100)^2 - \prod_{i=0}^{i=10} \cos\left(\frac{x_i - 100}{\sqrt{i+1}}\right)$	$X = [100, 100, 100, 100, 100, 100, 100, 100, 100, 100]$ $F(X) = 0$

Table 3. The test parameter for the proposed SACHCA.

Parameters (symbols)	Symbols
Number of scout bees in the selected patches	*n*
Number of best patches in the selected patches	*m*
Number of elite patches in the selected best patches	*e*
Number of recruited bees in the elite patches	*nep*
Number of recruited bees in the non-elite best patches	*nsp*
The size of neighbourhood for each patch	*ngh*
Number of Iterations	*MaxIter*
Difference between fitness values of consecutive Iterations	*Error*
Angle limit	*angle_limit*
Step size for HC algorithm	*h*
Number of waiting time for HC algorithm	*HC_time_limit*

Table 4. Combinations of parameters values tested.

	Parameters									
Combination	*n*	*m*	e	*nep*	*nsp*	*h*	*ngh*	*MaxIter*	*angle_limit*	*HC_time_limit*
1	10	5	1	3	2	.01	1	5000	.5	1000
2	25	10	2	6	4	.05	1	5000	.5	1000
3	50	15	5	12	8	.1	1	5000	.5	1000
4	100	30	10	24	16	.5	1	5000	.5	1000
5	200	60	20	50	30	1	1	5000	.5	1000

more accurate results are closer to zero. The proposed algorithm performed significantly better than the other methods on most of the test functions such as, 2D-Goldstein&Price,2D-Schewel, 2D-Schaffer, 10D-Rosenbrock, 10D-Sphere, 2D-Martin&Gaddy and 2D-Easom where the average absolute difference of these functions was found to be 0 (see Table 6). However, the average absolute difference for 10D-Ackley was found to be .0013 and for 10D-Griwank was .003. Further, all the results for SACHCA were found to be better than the results of the basic BA. This behaviour was verified with the number of evaluations given in Table 8 and with the t-test given in Table 9.

As shown in Table 7, the average numbers of evaluations for 2D-Goldstein & Price 2D-Schewel, 2D-Schaffer, 10D-Rosenbrock, 10D-Sphere and 10-Rastrigin were found with proposed algorithm as 250,049, 1049, 118,049, 60,195.5, 11,292.3 and 49,678, respectively, where the average number of evaluation for the basic BA were found as 554,000, 1140, 121,088, 935,000, 285,039 and 885,000. However, the results found from 2D-Martin Gaddy and 2D-Easom using the proposed algorithm were not better than the basic BA. Further, the results found for the 10D-Ackley and 10D-Griwank were better for the proposed algorithm compared with the basic BA. The proposed algorithm performed better overall, especially for higher dimensional functions. According to *T*-test results (Table 9), the results found with the enhanced BA were significant compared with the basic BA in half of the cases when the confidence level was selected to be 95% ($\alpha < .005$).

Table 5. SACHCA-BA results for the different combinations of parameters values tested on continuous type functions.

		Combinations									
		Combination 1		Combination 2		Combination 3		Combination 4		Combination 5	
No	Function	Avg. Iter	Avg. Opt	Avg. Iter	Avg. Opt	Avg. Iter	Avg. Opt	Avg. Iter	Avg. Opt	Avg. Iter	Avg. Opt
1	Goldstein & Price (2D)	5000	3.610	5000	3.381	1427	3.000	1869	3.000	2756	3.000
2	Schwefel (2D)	5000	−837.163	5000	−837.428	6	−837.658	426	−837.658	1568	−837.658
3	Schaffer (2D)	5000	.632	5000	.258	675	.000	2503	.000	2846	.000
4	Rosenbrock (10D)	5000	.293	5000	.179	344	.000	745	.000	1956	.000
5	Sphere (10D)	5000	.635	5000	.458	65	.000	305	.000	1420	.000
6	Ackley (10D)	5000	.389	5000	.162	5000	.001	5000	.001	5000	.001
7	Rastrigin (10D)	5000	.575	5000	.348	284	.000	532	.000	2763	.000
8	Martin & Gaddy (2D)	5000	.415	5000	.326	8	.000	268	.000	1235	.000
9	Easom (2D)	5000	−.548	5000	−.759	1429	−1.000	2126	−1.000	3154	−1.000
10	Griewank (10D)	5000	.765	5000	.421	5000	.000	5000	.000	5000	.000

Note: Avg_Iter, average iterations; Avg_Opt, average optimum.

Table 6. The test parameters for the EA, PSO and ABC (Ahmad, 2012).

EA				
Parameters			Crossover	No crossover
Population size			100	
Evaluation cycles (max number)			5000	
Children per generation			99	
Crossover rate			1	0
Mutation rate (variables)			.05	.8
Mutation rate (mutation width)			.05	.8
Initial mutation interval width α (variables)			.1	
Initial mutation interval width ρ (mutation width)			.1	
PSO				
Parameters			Value	
Population size			100	
PSO cycles (max number) T			5000	
Connectivity			See below	
Maximum velocity			See below	
C1			2	
C2			2	
$w_{\max}$			.9	
$w_{\min}$			.4	
Velocity of the each connectivity (connectivity, u)			Max particle velocity u	
Connectivity (number of neigbourhood)	(2, .005)	(2, .001)	(2, .05)	(2, .1)
	(10, .005)	(10, .001)	(10, .05)	(10, .1)
	(20, .005)	(20, .001)	(20, .05)	(20, .1)
	(100, .005)	(100, .001)	(100, .05)	(100, .1)
ABC				
Parameters				Value
Population size				100
ABC cycles (max number)				5000
Employed bees n_e				50
Onlooker bees n_e				49
Random scouts				1
Stagnation limit for site abandonment *stlim*				50× dimension

5. Application on single machine scheduling problem

5.1. Single machine scheduling problem

The proposed algorithm has been tested on a well-known manufacturing problem such as the single machine scheduling problem.

A single machine scheduling problem is a well-studied optimisation problem where a set of n-jobs with given deterministic processing times p_i and a common due date d have to be processed on a machine according to some constraints. The goal is to find a schedule for the n-jobs which minimises the sum of all the penalties which can occur due to the constraints. This is a challenging optimisation problem and is, therefore, chosen to test the performance of the proposed BA.

There are many solution techniques which could be applied to solving single machine scheduling problems. If the objective is to minimise the average flow time in Equation (7), then the optimal schedule will have the shortest processing time. If the objective is to minimise the maximum tardiness in Equation (5), then the optimal schedule will have the earliest due date. The Hodgson's algorithm gives an optimal solution if the objective is to minimise the number of jobs with tardiness greater than zero. The lateness in Equation (6) calculates any deviation from the due date, where a positive value for lateness indicates tardiness and a negative value is earliness. In this study, the goal will be to find a schedule for the n-jobs which jointly minimises the sum of earliness and tardiness penalties according with Equation (3).

$$\text{SUM-P} = \sum_{i=1}^{n} \alpha_i E_i + \beta_i T_i \tag{3}$$

where α_i and β_i are the earliness and tardiness penalties for job i per unit time, and E_i and T_i are the jobs' earliness and tardiness for job $\underline{i}$, respectively, derived from

$$E_i = \max(0, d - C_i) \tag{4}$$

$$T_i = \max(0, C_i - d) \tag{5}$$

$$\text{Lateness} = C_i - d \tag{6}$$

$$\text{Flow time} = \text{End date} - \text{Start date} \tag{7}$$

where d is due date and C_i is the completion time of the job i.

5.2. *Experimental procedure*

The data-sets available in OR-Library website (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/schinfo.html) were used to find the common due dates for different jobs assigned to a single machine. A data-set is composed by nj (number of jobs) with given deterministic processing times p_i and a common due date d. Say that the jobs have to be processed on one machine and that for each of the jobs, an individual earliness E_i and tardiness T_i penalty is given, which is incurred, if a job is finished before or after the common due date d, respectively.

The format of the data file is as follows:

For each problem in turn: nj

for each job i ($i = 1, \dots, nj$) in turn: p_i, E_i, T_i are given.

The common due date d is calculated by:

$$d = round[\text{SUM\_}P \times rf] \tag{8}$$

where round[X] gives the biggest integer which is smaller than or equal to X, SUM_P denotes the sum of the processing times of the nj jobs and the parameter rf(restrictive factor) is used to calculate more or less restrictive common due dates.

The proposed SACHCA-BA was implemented in Matlab code on a Pentium i7(2.93 GHz) PC and was used to solve the single machine scheduling problem. The data-sets available in OR-Library (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/schinfo.html) were used to find the common due dates for different jobs assigned to a single machine. There are seven different data-sets with different numbers of jobs (nj = 10, 20, 50, 100,

Table 7. Accuracy of proposed algorithm compared with other well-known optimisation techniques.

	PSO		EA		ABC		BA		SACHCA-BA	
Function	Avg. Abs. Dif.	Std. Dev.	Avg. Abs. Dif.	Std. Dev.	Avg. Abs. Dif.	Std. Dev.	Avg. Abs. Dif.	Std. Dev.	Avg. Abs. Dif.	Std. Dev.
1. Goldstein & Price	.0000	.0000	.0000	.0000	.0000	.0001	.0000	.0003	.0000	.0000
2. Schwefel	4.7376	23.4448	4.7379	23.4448	.0000	.0000	.0000	.0005	.0000	.0000
3. Schaffer	.0000	.0000	.0009	.0025	.0000	.0000	.0000	.0003	.0000	.0002
4. Rosenbrock	.5998	1.0436	61.5213	132.6307	.0965	.0880	44.3210	112.2900	.0000	.0001
5. Sphere	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0003	.0000	.0000
6. Ackley	.0000	.0000	.0000	.0000	.0000	.0000	1.2345	.3135	.0013	.0032
7. Rastrigin	.1990	.4924	2.9616	1.4881	.0000	.0000	24.8499	8.3306	.0000	.0038
8. Martin& Gaddy	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0003	.0000	.0000
9. Easom	.0000	.0000	.0000	.0000	.0000	2.0096	.0000	.0003	.0000	.0002
1. Griewank	.0008	.0026	.0210	.0130	.0052	.0078	.3158	.1786	.0003	.0005

Note: Abs. Dif., absolute difference; Std. Dev., standard deviation.

Table 8. Average evaluation of proposed algorithm compared with other well-known optimization techniques.

	PSO		EA		ABC		BA		SACHCA-BA	
Function	Avg. Eval.	Std. Dev.	Avg. Eval.	Std. Dev.	Avg. Eval.	Std. Dev.	Avg. Eval.	Std. Dev.	Avg. Eval.	Std. Dev.
1.Goldstein & Price	3262	822	2002	390	2082	435	504.0000	211.0000	250,049	0
2. Schwefel	84,572	90,373	298,058	149,638	4750	1197	1140	680	1049	0
3. Schaffer	28,072	21,717	219,376	183,373	21,156	13,714	121,088	174,779	118,049	0
4. Rosenbrock	492,912	29,381	500,000	0	497,728	16,065	935,000	0	60,195.5	16,092.3
5. Sphere	171,754	7732	36,376	2736	13,114	480	285,039	277,778	11,292.3	2343
6. Ackley	236,562	9119	50,344	3949	18,664	627	910,000	0	15,426	5402.6
7. Rastrigin	412,440	67,814	500,000	0	207,486	57,568	885,000	0	49,678.12	17.341.2
8.Martin & Gaddy	1778	612	1512	385	1498	329	600	259	1300.8	672.3
9. Easom	16,124	15,942	36,440	28,121	1542	201	5280	6303	250,049	0
10. Griewank	290,466	74,501	490,792	65,110	357,438	149,129	4300,000	0	87,777.8	38,285.3

Note: Avg. Eval., average evaluation; Std. Dev., standard deviation.

Table 9. The statistical significance between the improved BA and the basic BA.

		Significance between the basic BA and the enhanced BA	
No.	Function	Significant ($\alpha < .05$)	α
1	Goldstein & Price (2D)	Yes	1.58E−08
2	Schwefel (2D)	No	.4553
3	Schaffer (2D)	No	.2599
4	Rosenbrock (10D)	Yes	.027
5	Sphere (10D)	Yes	6.16E−09
6	Ackley (10D)	Yes	6.08E−06
7	Rastrigin (10D)	Yes	.0098
8	Martin & Gaddy (2D)	Yes	.0018
9	Easom (2D)	No	.1040
10	Griewank (10D)	No	.9567

200, 500 and 1000) and four value of restrictive factor (*rf* = .2, .4, .6 and .8). Each data-set contains 10 instances. Therefore, the algorithm was run on 280 instances and the results were compared with those obtained using the basic BA. The parameter values in common of the basic BA and the SACHCA-BA were selected according to Table 3.

The percentage offset between the basic BA and the SACHCA-BA results are calculate for each value of *nj* and *rf* as follows:

$$\%\text{Offset} = \frac{\text{SACHCA BA} - \text{BA}}{\text{BA}} 100 \tag{9}$$

5.3. Discussion the results of single machine scheduling

In order to evaluate the performance of the proposed algorithm, it has also been implemented on the single machine scheduling problem. Moreover, the results of the proposed algorithm were comparing with the results of basic BA using the offset between the respective results given in Equation (10). Further, the average offset of 10 instances for each couple of *nj* and *rf* are given in Table 10. The results show that the average offset is negative in all the cases, which means that, the SACHCA-BA outperformed compare to the basic BA in solving the single machine scheduling problem, shown in Figure 4.

$$\text{Average \% Offset} = \frac{1}{10}\sum_{k=1}^{10} \%\text{Offset}_k \tag{10}$$

Figure 4 shows the trends of the average offset for each couple of *nj* and *rf*. The performance of the SACHCA-BA improves compared with the BA when the *rf* and the *nj* increase, that is, when the problem becomes more complex. Lower is the value of *rf* and more stable is the average offset when *nj* increases. For instance, when *rf* = .2, the average offset is .00% for *nj* = 10, and −6.33% when *nj* = 1000. On the contrary, when

Table 10. The average offset between the SACHCA-BA and the basic BA.

nj	*rf* = .2	*rf* = .4	*rf* = .6	*rf* = .8
10	.00%	−.37%	−2.25%	−3.07%
20	−.97%	1.79%	−1.12%	−2.00%
50	−1.72%	−2.42%	−9.52%	−11.05%
100	−1.32%	−2.77%	−9.26%	−20.80%
200	−1.96%	−4.17%	−9.52%	−21.51%
500	−5.00%	−11.15%	−13.59%	−19.22%
1000	−6.33%	−24.04%	−35.75%	−45.33%

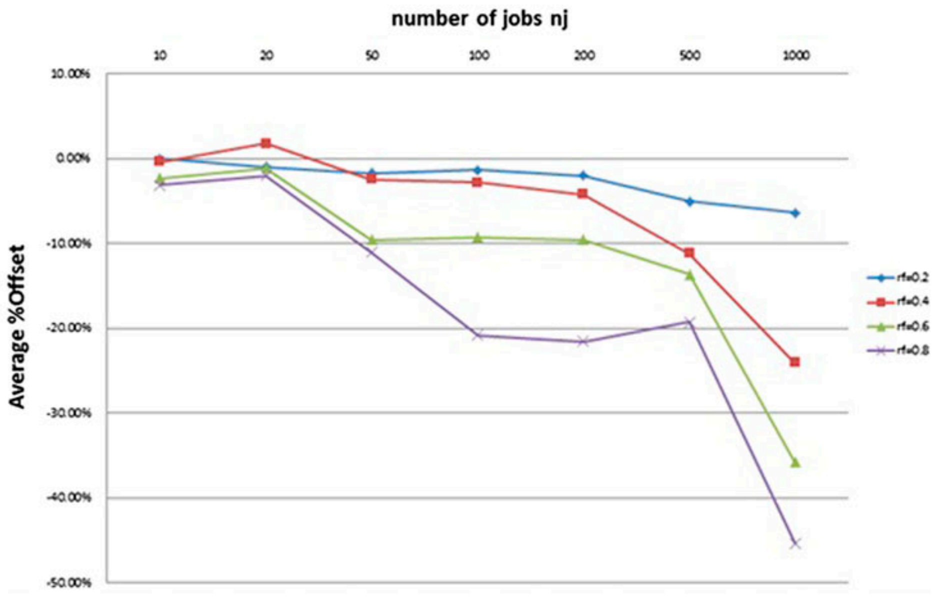


Figure 4. Average % offset values for each value of *nj* and *rf*.

rf value is higher, the average offset varies significantly with the *nj* value. For *rf* = .8, the average offset is −3.07% for *nj* = 10 and −45.33% for *nj* = 1000. Similarly, lower is the number of jobs *n* and closer are the average offsets when *rf* varies. For instance, when *nj* = 10, the average offset is .00% for *rf* = .2 and −3.07% for *rf* = .8. On the contrary, when *nj* value is higher, the average offset varies significantly with the *rf* value. For *nj* = 1000, the average offset is −6.33% for *rf* = .2 and −45.33% for *rf* = .8.

6. Conclusions

In this paper, an enhancement to the BA has been presented. The basic BA was modified to find the most promising patches, especially local optima, using SAC followed by HCA, for convergence, during the local search process. By including the improvement, both the search speed was improved and more accurate results were obtained. The enhancement to the BA was tested on 10 selected benchmark functions first and on a well-known manufacturing problem as the single machines scheduling in order to show its applicability and the performance in solving a real-world problem. Regarding the benchmark functions, according to the results of the accuracy analysis and the average evaluation, the proposed algorithm performed better on most of the benchmark functions. Based on the *T*-test results, it can be concluded that the proposed algorithm is statistically significant than the basic BA on higher dimensional and complex optimisation problems.

Regarding the single machine scheduling problem, the proposed SACHCA-BA outperformed the basic BA in almost all the considered instances, in particular when the complexity of the problem increases.

References

Ahmad, S. A. (2012). *A study of search neighbourhood in the Bees Algorithm* (PhD thesis). Cardiff University, Cardiff.

Ang, M. C., Ng, K. W., & Pham, D. T. (2013). Combining the Bees Algorithm and shape grammar to generate branded product concepts. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 227*, 1860–1873.

Ang, M. C., Ng, K. W., Pham, D. T., & Soroka, A. (2013). Simulations of PCB assembly optimisation based on the Bees Algorithm with TRIZ-inspired operators. In H. B. Zaman, P. Robinson, P. Olivier, T. K. Shih, & S. Velastin (Eds.), *Advances in visual informatics* (pp. 335–346). Selangor: Springer.

Azfanizam, A. S., Pham, D. T., & Faieza, A. A. (2014). Combination of adaptive enlargement and reduction in the search neighbourhood in the Bees Algorithm. *Applied Mechanics and Materials, 564*, 614–618.

Beni, G., & Wang, J. (1989). *Swarm intelligence in cellular robotic systems*. Proceeding of NATO Advanced Workshop on Robots and Biological System, Tuscany.

Castellani, M., Pham, Q. T., & Pham, D. T. (2012). Dynamic optimisation by a modified Bees Algorithm. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 226*, 956–971.

Gould, J. L., & Gould, C. G. (1988). *The honey bee*. New York, NY: Scientific American Library.

Grosan, C., & Abraham, A. (2011). *Intelligent systems: A modern approach*. Berlin: Springer.

Hussein, W. A., Sahran, S., & Abdullah, S. N. H. S. (2014). Patch-Levy-based initialization algorithm for Bees Algorithm. *Applied Soft Computing, 23*, 104–121.

Lien, L. C., & Cheng, M. Y. (2014). Particle bee algorithm for tower crane layout with material quantity supply and demand optimization. *Automation in Construction, 45*, 25–32.

Mastrocinque, E., Yuce, B., Lambiase, A., & Packianather, M. S. (2013). A multi-objective optimisation for supply chain network using the Bees Algorithm. *International Journal of Engineering Business Management, 5*, 1–11.

OR-Library, Common due date scheduling. Retrieved April 2, 2014, from http://people.brunel.ac.uk/~mastjjb/jeb/orlib/schinfo.html

Otri, S. (2011). *Improving the Bees Algorithm for complex optimisation problems* (PhD thesis). Cardiff University, Cardiff.

Pham, D. T., Afify, A. A., & Koç, E. (2007). Manufacturing cell formation using the Bees Algorithm. In *3rd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS 2007)* (pp. 523–528). Dunbeath: Whittles.

Pham, D. T., & Castellani, M. (2009). The Bees Algorithm: Modelling foraging behaviour to solve continuous optimization problems. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 223*, 2919–2938.

Pham, D. T., Castellani, M., & Fahmy, A. A. (2008). Learning the inverse kinematics of a robot manipulator using the Bees Algorithm. In *Proceedings of INDIN* (pp. 493–498).

Pham, D. T., Castellani, M., & Ghanbarzadeh, A. (2007). Preliminary design using the Bees Algorithm. In *Proceedings Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, LAMDAMAP* (pp. 420–429). Cardiff: Euspen.

Pham, D. T., Darwish, A. H., Eldukhri, E. E., & Otri, S. (2007). *Using the Bees Algorithm to tune a fuzzy logic controller for a robot gymnast*. Proceedings of IPROMS 2007 Conference, Cardiff.

Pham, D. T., & Ghanbarzadeh, A. (2007). *Multi-objective optimisation using the Bees Algorithm*. Proceedings of IPROMS 2007 Conference, Cardiff.

Pham, D. T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., & Zaidi, M. (2005). *The Bees Algorithm* (Report No. MEC 0501). Cardiff: Manufacturing Engineering Centre, Cardiff University.

Pham, D. T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., & Zaidi, M. (2006). The Bees Algorithm – A novel tool for complex optimisation problems. In *Proceedings of the 2nd Virtual International Conference on Intelligent Production Machines and Systems (IPROMS 2006)* (pp. 454–459).

Pham, D. T., Koc, E., Ghanbarzadeh, A., & Otri, S. (2006). *Optimisation of the weights of multi-layered perceptrons using the Bees Algorithm*. 5th International Symposium on Intelligent Manufacturing Systems, Sakarya.

Pham, D. T., Koc, E., Lee, J. Y., & Phrueksanant, J. (2007). Using the Bees Algorithm to schedule jobs for a machine. In *LAMDAMAP, 8th International Conference on Laser Metrology, CMM and Machine Tool Performance* (pp. 430–439), Euspen, Cardiff.

Pham, D. T., Otri, S., Afify, A., Mahmuddin, M., & Al-Jabbouli, H. (2007). *Data clustering using the Bees Algorithm*. 40th CIRP International Manufacturing Systems Seminar, Liverpool.

Pham, D. T., Packianther, M. S., Imaguliyev, A., & Yuce, B. (2012, September). Early neighbourhood search and efficiency based improved the Bees Algorithm. In IMS 2012 Conference (pp. 26–29), Turkey: Sakarya University.

Pham, D. T., Soroka, A. J., Koç, E., Ghanbarzadeh, A., & Otri, S. (2007). *Some applications of the Bees Algorithm in engineering design and manufacture*. Proceedings of International Conference on Manufacturing Automation (ICMA 2007), Singapore.

Seeley, T. D. (1995). *The social physiology of honey bee colonies: The wisdom of the hive*. London: Hardvard University Press.

Sholedolu, M. O. (2009). *Nature-inspired optimisation: Improvements to the Particle Swarm Optimisation Algorithm and the Bees Algorithm* (PhD thesis). Cardiff University, Cardiff.

Talbi, E. G. (2009). *Metaheuristics*. New Jersey, NJ: Wiley.

Von Frisch, K. (1955). *The dancing bees: An account of the life and senses of the honey bee*. New York, NY: Harcourt.

Woeginger, G. J. (2003). Exact algorithms for NP-hard problems: A survey. In M. Juenger, G. Reinelt, & G. Rinaldi (Eds.), *Combinatorial Optimization: Eureka! You shrink!* (pp. 185–207). Aussois: LNCS 2570: Springer.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*, 67–82.

Yuce, B. (2012). *Novel computational technique for determining depth using Bees Algorithm and Blind Image Deconvolution* (PhD thesis). Cardiff: Cardiff University.

Yuce, B., Mastrocinque, E., Lambiase, A., Packianather, M. S., & Pham, D. T. (2014). A multi-objective supply chain optimisation using enhanced Bees Algorithm with adaptive neighbourhood search and site abandonment strategy. *Swarm and Evolutionary Computation, 18*, 71–82.

Yuce, B., Packianather, M. S., Mastrocinque, E., Pham, D. T., & Lambiase, A. (2013). Honey bees inspired optimization method: The Bees Algorithm. *Insects, 4*, 646–662.