Marriage in Honey Bees Optimization Algorithm for Flow-shop Problems

Pedro PALOMINOS[1], Francisco TOLEDO[1], Andrés VÉJAR[2], Miguel ALFARO[1]
[1] Industrial Engineering Department, University of Santiago of Chile, Santiago, Chile
[2] INRA, UMR 782 Génie Microbiologique et Alimentaire, AgroParisTech, INRA, 78850 Thiverval-Grignon, France
{pedro.palominos, francisco.toledo,miguel.alfaro}@usach.cl, andres.vejar@grignon.inra.fr

The objective of this work is to make a comparative study of the Marriage in Honeybees Optimization (MBO) metaheuristic for flow-shop scheduling problems. This paper is focused on the design possibilities of the mating flight space shared by queens and drones. The proposed algorithm uses a 2-dimensional torus as an explicit mating space instead of the simulated annealing one in the original MBO. After testing different alternatives with benchmark datasets, the results show that the modeled and implemented metaheuristic is effective to solve flow-shop type problems, providing a new approach to solve other NP-Hard problems.
Keywords: *Metaheuristics, Flow-Shop, Scheduling, Bio Inspired Optimization*

1 Introduction

Evolutionary computation refers to the set of methods that have a biological inspiration and allow solving complex problems. These methods are based on the principle that evolution is an optimization process in which it is attempted to improve the skills of individuals and therefore achieve better adaptation to their environment as a result of greater overall survival. In particular, Swarm Intelligence (SI) is an area of artificial intelligence focused on modeling the behavior of social insects like ants and bees [1]. Ant Colony Optimization (ACO) is one of the better known models of the SI type in which the design of optimization algorithms is inspired by the decentralized and collective behavior of ant colonies [2], [3]. Another metaheuristic of the SI type is Marriage in Honeybees Optimization (MBO), which is inspired by the mating flight of honey-making bees, and was proposed by Abbass [4-6] to apply it to the SAT problem, which is an NP-Complete problem. The purpose of this work is to test the efficiency of the MBO metaheuristic in the permutational flow shop problem and compare the results between the different design alternatives of the mating flight space.

1.1 Marriage in Honey-Bees Optimization

The main stages of the MBO algorithm are:

- The mating flights of the queens with the drones to choose the parents of the future larvae.
- Creation of new offspring by the queens, which correspond to the crossing of genetic material from the queen bee with the drones chosen in the mating flight.
- Improvement of the health quality of the offspring by the workers by mutation.

A schematic diagram is presented on Figure 1.

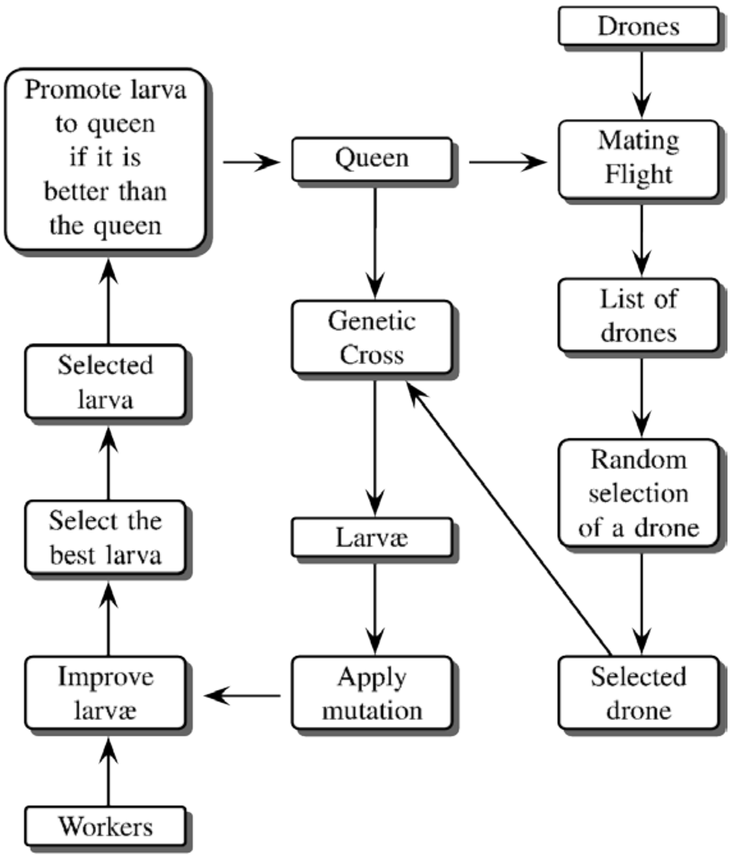

Fig. 1. Graphic representation of the MBO technique

The key stage of the metaheuristic is the mating flight of the queens with the drones, because in this stage the queen can choose to mate with a drone depending on the quality of the drone found. The queen stores the genetic material of the chosen drones in its spermatheca. As the flight takes place, the queen loses energy and speed. The process ends when the queen uses up the available energy or fills its spermatheca to capacity. The loss of speed affects the queen's ability to choose the drones; at first the queen's movements are long and allow it to choose among a large number of drones. As the queen's speed decreases, its movements become shorter, so it can only choose the drones that are closest, within the range defined by its speed.

The energy of a queen at the beginning of a flight is represented by E_0, $0<E_0\leq 1$, and it is chosen uniformly for each queen, $E_0 \sim U(0,1)$. Depending on the selection of parameters proposed in[5, 6], this choice ensures 7 to 17 matings per flight. In this implementation of the MBO algorithm the queen bee moves in the flight space according to a random walk with decreasing steps. The drones, on the other hand, will generate their trajectories according to a scheme inherited from Particle Swarm Optimization (PSO)[7], using individual and social factors at each step. The updating of the position x and speed v of each drone is defined by:

$$x_{k+1} = x_k + v_{k+1} \quad (1)$$

$$v_{k+1} = \phi_0 v_k + \phi_1(p_1 - x_k) + \phi_2(p_2 - x_k) \quad (2)$$

Where $\phi_i>0$, $i\in\{0,1,2\}$, ϕ_0 is the importance factor of speed, ϕ_1 is the importance factor of the individuals drones best position, p_1, and ϕ_2 is the importance factor of the best drone's best position, p_2. If $\phi_1>\phi_2$, more importance is assigned to individual learning, in reference to the best position found by the drone itself. On the other hand, if $\phi_2>\phi_1$, more importance is assigned to the learning of the swarm, in reference to the best position found by the set of drones. Every time a queen chooses a drone for a possible crossing, the probability that the queen will store the drone's genetic material in its spermatheca is given by:

$$P(q,d) = \min\{\exp(-\Delta f/S),1\} \quad (3)$$

Where Δf is the absolute difference between the fitness of the drone *d* and queen *q*, and *S* is the speed of the queen. The probability of mating is high when the distance between theirs fitness is short or when the queen's speed is high. After each transition state, the queen's energy and speed are updated according to the following equations:

$$E_{k+1} = gE_k \quad (4)$$
$$S_{k+1} = (1-a)S_k \quad (5)$$

Where the initial speed S_0 of the queen is chosen uniformly, $S_0 \sim U(0,1)$, and $0<a<1$ is the speed reduction parameter. *E* is the queen's energy, and *g* is the energy reduction factor after each transition, defined by:

$$g = 1-1/(2M) \quad (6)$$

Where *M* is the maximum capacity of the queen bee's spermatheca.
The mating flight ends when the queen uses up all its energy, E=0, or when its spermatheca reaches its maximum available capacity *M*. As seen in (4), a cutoff value $0<E_{min}<0.5$ must be chosen to represent the final value of the queen's energy:

$$E_{k+1} = \begin{cases} gE_k, & gE_k > E_{\min} \\ 0, & gE_k \le E_{\min} \end{cases} \quad (7)$$

When the queen ends its flight, the larva generation process begins. The drones' sperm is chosen randomly from the spermatheca and a larva is created by crossing genetic material, using half of the queen bee's genetic information and completing the rest with the drone's genotype. The following phase is larva mutation and the improvement of the quality of the larvae by the worker bees. The worker bees are represented by local search algorithms or simple heuristics that allow the search for solutions in a reduced domain in a reasonable time. Finally, the lower quality queen bees will be replaced by the larvae that represent better solutions than the present queens. The remaining larvae are eliminated. With the updated set of queens a new flight can be executed. The process ends when a pre-established maximum number of flights are reached or when some other termination condition is reached.

2 Flow-Shop Modeling for MBO

Permutation flow-shop represents a particular case of the flow-shop scheduling problem, having as goal the deployment of an optimal schedule for *n* jobs on *m* machines. Solving the flow-shop problem consists in scheduling *n* jobs (i=1…n) on *m* machines (j=1…m). A job consists of *m* operations, and the j^{th} operation of each job must be processed on machine *j*. So one job can start on machine *j* if it is completed on machine j_1 and if machine *j* is free. Each operation has a known processing time $p_{i,j}$. As a consequence for the permutation flow-shop problem considering the makespan as target function to be minimized, to solve the problem means determining the permutation that gives the smallest makespan value.
Makespan $C_{\max} = C_{(J_n,m)}$ is the termination time for *n* works on *m* machines, and it is calculated with the recursive formula:

$$\begin{aligned} C_{(J_i,k)} &= \max\{C_{(J_{i-1},k)}, C_{(J_i,k-1)}\} + t_{(J_i,k)} \\ & \qquad 2 \le k \le m, \quad 2 \le i \le n \\ C_{(J_1,k)} &= C_{(J_1,k-1)} + t_{(J_1,k)}, \quad 2 \le k \le m \\ C_{(J_i,1)} &= C_{(J_{i-1},1)} + t_{(J_i,1)}, \quad 2 \le i \le n \\ C_{(J_1,1)} &= t_{(J_1,1)} \end{aligned} \quad (8)$$

where:

- $C_{(J_i,k)}$: Time for finishing task J_i on machine *k*.
- $\{J_1,\ldots,J_n\}$: Permutation of *n* tasks.

- $t_{(J_i,k)}$: Time for processing task J_i on machine *k*.
- *n* : Number of tasks.
- *m* : Number of machines.

To be able to represent the problem from the MBO perspective, the following steps were taken:

2.1 Structure

First, the genotypes of the drones, the queens, and the larvae must be established. Each of these genotypes will be represented by a permutational sequence of works of the same length for all the individuals that will depend on the number of tasks existing in each problem. For example, **Fig. 2** depicts representations of a queen, a drone and a larva when eight jobs are available.

Queen Bee	1	2	3	4	5	6	7	8
Drone	4	5	6	3	8	2	1	7
Larva	3	8	6	7	5	2	6	1

Fig. 2. Structure of the individuals' genotype

2.2 Population

A queen bee population is generated, the best of which is the one obtained by means of Palmer's Heuristic. The rest of the queen bees will be obtained by a random procedure, to generate more diversity. The drone population will be obtained by a random procedure in which the tasks that make up their sequence are assigned in such a way that each drone generated is feasible. Also, the drones cannot have the same genetic material, i.e., in different flights different drones will be created, with the purpose of increasing the exploration of permutations.

2.3 Trajectory generation

To be able to generate the trajectories of the queens and the drones they must first be positioned randomly in the n-dimensional space, i.e., initial positions will be created by chance for each individual. This stage will be divided into two: one corresponding to the queen bee transition and the other to the drones. The queen bees will move randomly over a limited neighborhood, and the drones according to (1) and (2).

2.4 Drone selection

As already mentioned, the trajectories of the queen bees will depend on chance. Once that has happened, the probability that the drones have of mating with a queen bee is calculated (3)-(5). Once the probability of mating with a particular drone is obtained, the queen bee chooses randomly if it will open its spermatheca to the chosen drone. For that it will generate a number (W) between 0 and 1 that will represent the queen bee's decision. If that number is smaller than the drone's mating probability, the queen will open its spermatheca to allow the entry of the drone's sperm. This happens because the queen's predisposition to mate is greater than the probability of the drone, so the queen's decision will be to mate with it.

2.5 Genetic crossing

Once the drone selection has been made, the next step is processing that information, or the genetic step, which has two operators, defined as order crossing and mutation. The crossing used is that proposed by Davis [8], called OX. In **Fig. 3**, five jobs are used to describe a crossover between queen Q_1 and drone D_1 to produce two larvae.

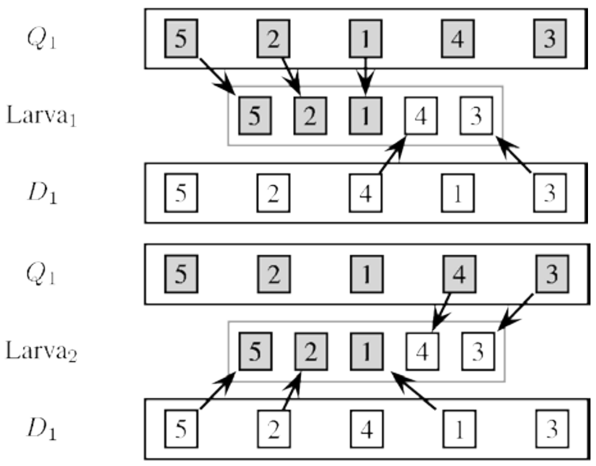

Fig. 3. OX crossing between queen bee and drone

Mutation involves the random alteration of each component in the sequence of jobs. The mutation used is illustrated in **Fig. 4.**

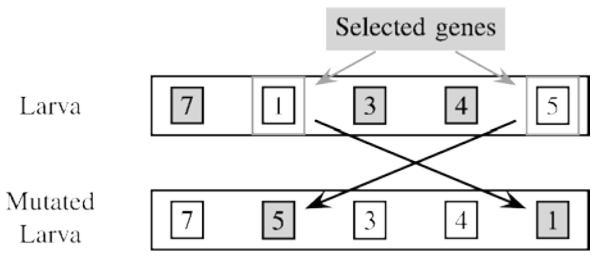

Fig. 4. Mutation for a larva

The mutation is triggered as follows: A random number between 0 and 1 is generated for each larva and it is compared with its mutation probablility; if this number is smaller than the mutation probability, the larva mutates. Otherwise the larva's genotype remains intact.

2.6 Evaluation of the queen and the larva

The evaluation function must determine if an individual is apt or not, in the configuration space, by means of a cost associated with it. This mode of implementation seeks to obtain the best makespan for each individual.

2.7 Updating population

This procedure considers replacing the least adapted queen bees by the best larvae obtained from the crossing of a queen bee and a selected drone.

2.8 Updating parameters

In each of their flights the queen bees loose energy and speed, and they may accept some drone for mating, which implies that they will have less space in their spermatheca to accept another mating. That is why the parameters must be updated in each flight.

2.9 Stopping criteria

When the heuristic is iterated, it will always voraciously reduce the target function. In view of that, the criterion for making a more exhaustive study of the efficiency of the MBO algorithm is the definition of a certain number of flights (iterations) initialized and entered at the start of the program. In this case the stopping criterion was 1000 iterations.

Finally, Table 1 summarizes the values of the metaheuristic's parameters for the flow-shop problem. It should be noted that some were obtained empirically and others were taken from the literature.

Table 1. Parameters for MBO Implementation

Parameters	**Source**	**Value**
Number of queens	Experiment	3
Drone population	Experiment	100
Capacity of spermatheca (*M*)	Experiment	100
Initial speed of queens (S_0)	Experiment	10
Initial energy of queens (E_0)	Literature [6]	*U*(0,1)
Energy reduction factor for queens (*g*)	Literature [6]	0.1

3 Proposed variants for the MBO metaheuristic

The original MBO algorithm is a hybrid metaheuristic of simulated annealing (SA), genetic algorithms (GA), and local search (LS), in view of the use of a mating function similar to SA, and the advantages of GA and LS in the generation and improvement of the larvae by the worker bees. However, the MBO metaheuristic has a large number of particular characteristics that distinguish it from the rest[4-6].

As described previously, the main MBO processes are the following:

- Mating in flight of the queen bee with the drones.
- Creation of new queen bee larva.
- Larvae improvement by the workers.
- Replacement of the less apt queens by the best larva.

The key process of interest for the present work is the flight mating process of the queen bee where the drones are. With the purpose of improving the pure MBO algorithm, we will use the PSO model[7] in a toroidal space to select apt drones in each swarm for mating with the queens.

All the remaining processes will be kept constant and as defined in the MBO developed by Abbas[5, 6]. Given the logic of the PSO algorithm, the creation space of the mating flight is defined as an *n*-dimensional space, and as a way of interfering as little as possible with the drone generation space we will use a toroidal space.

The surface of the 2-dimensional torus used as flight space. In the torus, the coordinates (x_T, y_T) follow:

$$\begin{pmatrix} x_T \\ y_T \end{pmatrix} = \begin{pmatrix} x(\bmod n_x) \\ y(\bmod n_y) \end{pmatrix} \qquad (9)$$

Where n_x and n_y are the length of each coordinate axis and (*x*,*y*) are the coordinates of the Euclidean space.

In PSO the drones fly through the problem's space, where each one keeps a record of its coordinates in the problem's space that are associated with the best solution that it has achieved so far. Each drone will be represented as a vector that is updated at each cycle. Therefore, each position of the drones will be represented by a tuple (x_T, y_T) in the torus. The size of this space will be given arbitrarily by the experiments made, and will depend on the size of the population to be generated. In this case the population was set at 100 drones and the space used was of the order of $(n_x, n_y) = (100,100)$. The distance between two points in the torus was calculated by the following formula:

$$d(u,v) = (\Delta_x + \Delta_y)^{\frac{1}{2}} \qquad (10)$$

Where (Δ_x, Δ_y) represents the shortest path between one point and another for each axis:

$$\Delta_x = \min\{|u_x - v_x|, n_x - |u_x - v_x|\} \quad (11)$$

$$\Delta_y = \min\{|u_y - v_y|, n_y - |u_y - v_y|\} \quad (12)$$

4 Experimental Results

The computational development for testing the proposed heuristic was made on a computer with an AMD Athlon 64 3000+ processor, clock speed of 2.0 GHz, 1 GB RAM, with Linux, programmed in Python. The instances used were extracted from the data base available in the OR-Library[9], where 120 Taillard's instances [10] were considered, grouped as shown in Table 2, where each configuration has 10 test instances.

Table 2. Characteristics of used instances

Configuration	**Tasks** (*N*)	**Machines** (*M*)	**Instances**
1	20	5	ta001-010
2	20	10	ta011-020
3	20	20	ta021-030
4	50	5	ta031-040
5	50	10	ta041-050
6	50	20	ta051-060
7	100	5	ta061-070
8	100	10	ta071-080
9	100	20	ta081-090
10	200	10	ta091-100
11	200	20	ta101-110
12	500	20	ta111-120

The results obtained for each configuration evaluated are presented in Table 3, where the first column gives the number of the configuration; the second gives the number tasks; the third gives the number of machines; and the fourth gives the mean error of the original

MBO metaheuristic measured as percentage of the best solution found and the best solution from the data base, divided by the latter; the fifth gives the mean efficiency of the modified MBO metaheuristic, measured as percentage of the best solution found and the best solution from the data base, divided by the latter; and the sixth gives the difference between the original MBO metaheuristic and the proposed modified MBO metaheuristic. In Table 3 it is seen that the overall average error for the 120 test instances is 4.93% for the original MBO metaheuristic and 3.19% for the modified MBO metaheuristic, compared with the best solution given in the OR-Libray database. These results are considered satisfactory for the proposed metaheuristic, since there are significant differences between the two metaheuristics studied.

Table 3. Results

Configuration	**Tasks (N)**	**Machines (M)**	**Mean error**	**Mean efficiency**	**Difference**
1	20	5	3.6	1.4	-2.2
2	20	10	4.8	2.5	-2.3
3	20	20	4.7	2.3	-2.4
4	50	5	3.3	0.9	-2.4
5	50	10	7.3	4.4	-2.8
6	50	20	9.1	6.2	-2.9
7	100	5	2.3	0.8	-1.4
8	100	10	4.6	2.7	-1.9
9	100	20	9.1	6.6	-2.5
10	200	10	4.1	2.3	-1.8
11	200	20	8.9	6.6	-2.3
12	500	20	6.3	5.0	-1.3
Mean			4.93	3.19	-1.75
Max			9.1	6.6	
Min			3.6	0.8	

5 Conclusion

The main contribution of this work is the proposal of a new metaheuristic based on honeybee mating, proposed by Abbass [4-6] to tackle a scheduling problem of the flow shop type, achieving very good results for the 120 tested instances. Also, a first application of the MBO technique to the scheduling problem of the combinatorial flow-shop type is presented. However, the proposed MBO metaheuristic is a first approximation, so future work may take two routes: (1) Test the efficiency of the metaheuristic in other types of combinatorial problems like that of the traveling salesman, among others, and (2) study other larva improvement techniques and carry out a parameterization study, in view of the large number of parameters used by this technique.

Acknowledgment

This project was supported by The University of Santiago of Chile (060517PB DICYT-USACH).

References

[1]E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. Oxford University Press, USA, 1999.

[2]M. Dorigo and L. Gambardella, “Ant colony system: A cooperative learning approach to the traveling salesman problem,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[3]M. Dorigo and G. Di Caro, “Ant colony optimization: a new meta-heuristic,” in *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 2.1em

plus 0.5em minus 0.4emIEEE, 1999, pp. 1470–1477.

[4]H. Abbass, “A Single Queen Single Worker Honey–Bees Approach to 3-SAT,” in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, San Francisco, USA, 2001, pp. 807–814.

[5]H. Abbass, “MBO: Marriage in Honey Bees Optimization - A Haplometrosis Polygynous Swarming Approach,” in *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 1.1em plus 0.5em minus 0.4emIEEE, 2001, pp. 207–214.

[6]H. Abbass, “A monogenous MBO approach to satisfiability,” in *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA*, Las Vegas, NV, USA., 2001.

[7]J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” in *Proceeedings of the 1995 IEEE International Conference on Neural Networks*, vol. 4, Piscataway, NJ, USA, 1995, pp. 1942–1948.

[8]L. Davis, “Job shop scheduling with genetic algorithms,” in *Proceedings of the 1st International Conference on Genetic Algorithms*.1em plus 0.5em minus 0.4emHillsdale, NJ, USA: L. Erlbaum Associates Inc., 1985, pp. 136–140.

[9]J. E. Beasley, “OR-Library: distributing test problems by electronic mail,” *Journal of the Operational Research Society*, vol. 41, no. 11, pp. 1069–1072, 1990.

[10] E. Taillard, “Benchmarks for basic scheduling problems,” *European Journal of Operational Research*, vol. 64, no. 2, pp. 278–285, January 1993.

Dr. Pedro Iván PALOMINOS is currently Provost of the University of Santiago of Chile. He holds a Bachelor Degree and a Professional Degree in Industrial Engineering at the University of Santiago of Chile. He also holds a M.Sc. Degree in Engineering Production at the Federal University of Rio de Janeiro, Brazil and a Dr. Degree in Industrial Engineering at the University of Catalunya, Spain. Dr. Palominos has done teaching and research in Operations Management and Metaheuristics.

Francisco J. TOLEDO received his BSc. in Industrial Engineering (2007) from the University of Santiago of Chile, and the M.Sc. in sciences of the engineering, mention engineering industrial (2008) from the University of Santiago of Chile, and PhD student in Innovation (2010-2013) from the University of Lorraine, ERPI, Nancy, 5400, France. His research interests include the impacts of innovation and entrepreneurship, sustainable supply chains, sustainable development innovation and models based on systems multiagents.

Miguel D. ALFARO (b. May 4, 1956) received his BSc. in Industrial Engineering (1985) from the University of Santiago of Chile, and the M.Sc. in Engineering Economy (1990) from The Catholic University of Chile, and the Doctor degree in Automatic Production (1998) from the University Henri Poincaré, Nancy I, France. He is full-time professor at the Industrial Engineering Department, University of Santiago of Chile since 1998. His current research interests include Non linear Systems Dynamics, Artificial Intelligence in Manufacturing Systems and Agent-based Modeling Systems. He has authored over 30 articles in journals and proceedings, and participated in more than 50 conferences and workshops. He has acted as chairperson at several conferences in Chile, member of scientific committees, and referee of various journals.