3

Comparison of Swarm Optimization and Genetic Algorithm for Mobile Robot Navigation

Petar Ćurković, Bojan Jerbić and Tomislav Stipančić
University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture
Croatia

1. Introduction

Swarm optimization, swarm intelligence and swarm robotics are the fields considering a group of relatively simple individuals able cooperate to perform complex tasks, in decentralized manner. The inspiration is found in the first line within animal societies, such as birds, ants and bees. Social insects exhibit successful behavior in performing complex tasks on the level of the group, and are able to eliminate noise, errors, failure of swarm members. These swarms are robust, able to adapt to constant environmental changes in conditions of limited communications among members and lack of global data. In the context of swarm optimization, the example of Dorigo's "Ant Colony Optimization " (ACO) and Kennedy ad Eberhart "Particle swarm Optimization" (PSO) are most known examples of applying swarm-based concepts to development of optimization algorithms able to cope with hard optimization problems. These algorithms are justifiably called swarm algorithms, because they are run asynchronously and in decentralized manner (Benni, 2004). They also mimic the stigmergic (communication by dynamically changing environment) behavior of swarm of insects.

PSO is inspired by flocking behavior of the birds searching for food. Although PSO shares many common attributes with the field of Genetic Algorithms (GA), such as stochastic nature, population of solution candidates, PSO methods, unlike GA use a kind of cooperation between the particles to drive the search process. PSO methods have no evolutionary operators like crossover and mutation. Each particle keeps track of its own best solution, and the best solution found so far by the swarm. It means that the particles posses own and collective memory, and are able to communicate. The difference between the global best and personal best is used to direct particles in the search space.

ACO employs the search process that is inspired by the collective behavior of trail deposit and follow-up, which is observed within real ant colonies. A colony of simple agents, the ants, communicates indirectly via dynamic modifications of their environment (trails of pheromones) and thus proposes solution to a problem, based their collective experience.

Honey Bees Mating Algorithm (HBMA) can also be observed as a typical swarm based approach to optimization. The algorithm is inspired by behavior of eusocial insects, which are characterized by three main features: cooperation among adults in brood care and nest construction, overlapping of at least two generations, and reproductive division of labour,

respectively. In a recent work, Abbas proposed an optimization algorithm based on honey-bees mating process (Abbas, 2001; Abbas, 2002).
The path planning problem of a mobile robot is to find a safe and efficient path for the robot, given a start location, a goal location and a set of obstacles distributed in a workspace (Latombe, 1991.). The robot can go from the start location to the goal location without colliding with any obstacle along the path. In addition to the fundamental problem, we also try to find a way to optimize the plan, i.e. to minimize the time required or distance travelled (Du et al., 2005; Sadati and Taheri,2002; Ramakrishnan and Zein-Sabatto, 2002).
The popular methods are the visibility graph algorithm and the artificial potential field algorithm. However, the former lacks flexibility and the latter is prone to suffer from difficulties with local minima (Alexopoulos and Griffin, 1992; Chen and Liu, 1997). Neural network and genetic algorithm have been shown to be very efficient in robot navigation (Zarate et al., 2002). General path planning methods based on neural network always establish the neural network model for a robot from the start position to the goal position and entail much computational time. The input data of the model are the previous distance values and position or direction from the sensors. The output data are the next position or direction by self-learning process.
Genetic algorithm is multisearch algorithm based on the principles of natural genetics and natural selection (Goldberg, 1989). Genetic algorithm provides a robust search in complex spaces and is usually computationally less expensive than other search algorithms. Genetic algorithm searches the solution from a population of points and is less likely to be trapped in a local optimum. Many results in the literature show the good application of genetic algorithm in robot path planning (Khoogar and Parker, 1991; Ram et al., 1994).
In this chapter, concept of swarm intelligence, as an optimization technique is proposed for finding collision free paths in work space containing differently shaped and distributed obstacles. Thus, the problem of path planning is considered as an optimization problem, whereat collision free paths receive higher fitness values relative to those resulting in collision with an obstacle. Performance of HBMA algorithm is compared to the performance of a GA developed for the same purpose on two examples, Diophantine equation problem and path planning problem.
Organization of the chapter is as follows: in section 2 we briefly describe colony of Honey Bees, as they are in nature. Section 3 describes proposed abstraction and simplification and describes core elements of the algorithm. In section 4 and 5 HBMA is compared with GA for the first test case, Diophantine equation, and the performances of both algorithms in terms of completeness of the solution and speed of the convergence are discussed. In sections 5 and 6 both algorithms are applied to the second test case, path planning. We conclude with section 7 by finally comparing both algorithms and proposing further possibilities of improving and testing of the described algorithms.

2. Structure of a Honey-Bee Colony

A honey-bee colony typically consists of a single egg laying queen, usually from zero to several thousands drones and 10000 to 60000 workers. Drones are the fathers of the colony. They are haploid and act to amplify their mother's genome without alteration of their genetic composition except through mutation. Workers specialize in brood care and sometimes lay eggs. Broods arise from either fertilized or unfertilized eggs, whereby the

former represent potential queens or workers, and the latter represent prospective drones. The mating process occurs during mating-flights far from the nest. A mating flight starts with the dance where the drones follow the queen and mate with her in the air. In a typical mating-flight, each queen mates with seven to twenty drones. In each mating, sperm reaches the sprematheca and accumulates there to form the genetic pool of the colony. Each time a queen lays fertilized eggs, she retrieves at random a mixture of the sperms accumulated in the spermatheca to fertilize the egg.

3. Artificial Model

The main processes of the algorithm are: mating flight of the queen with the drones, creation of new broods by the queen, improvement of the broods by workers, adaptation of workers fitness, replacement of the queen with the fitter brood. The mating flight may be considered as a set of transitions in a state-space (the environment) where the queen moves between the different states in some speed and mates with the drone encountered at each state probabilistically, according to (1).

At the start of the flight, the queen is initialized with some energy content, typically this is a random value from range (0,1] and returns to her nest when energy content equals to zero or when her spermatheca is full. In developing the algorithm, the functionality of workers is restricted to brood care, and therefore, each worker may be represented as a different heuristic which acts to improve a set of broods.

A drone mates with a queen probabilistically according to annealing function:

$$prob(Q,D) = e^{-\frac{\Delta(f)}{S(t)}} \tag{1}$$

Where *prob*(Q,D) represents the probability of successful mating, i.e. the probability of adding drone's D sperm to queen's Q spermatheca. Δ(f) is the absolute difference between the fitness of the drone and the queen, and S(t) is the speed of the queen at time t.

According to defined annealing function, the probability of mating is high when either the queen is the start of her flight, and therefore, her speed is high, or when the fitness of the new potential drone is similar to the queen's fitness. The main steps of the algorithm are presented in Fig. 1.

After each transition in space, the queen's speed S(t) and energy E(t) decay using the following equations:

$$S(t+1) = \alpha \cdot S(t) \tag{2}$$

$$E(t+1) = E(t) - \gamma \tag{3}$$

Where α is a factor in range [0.5, 1] and γ is calculated according to expression:

$$\gamma(t) = \frac{0.5 \cdot E(t)}{M} \tag{4}$$

And M is the size of sphermatheca.

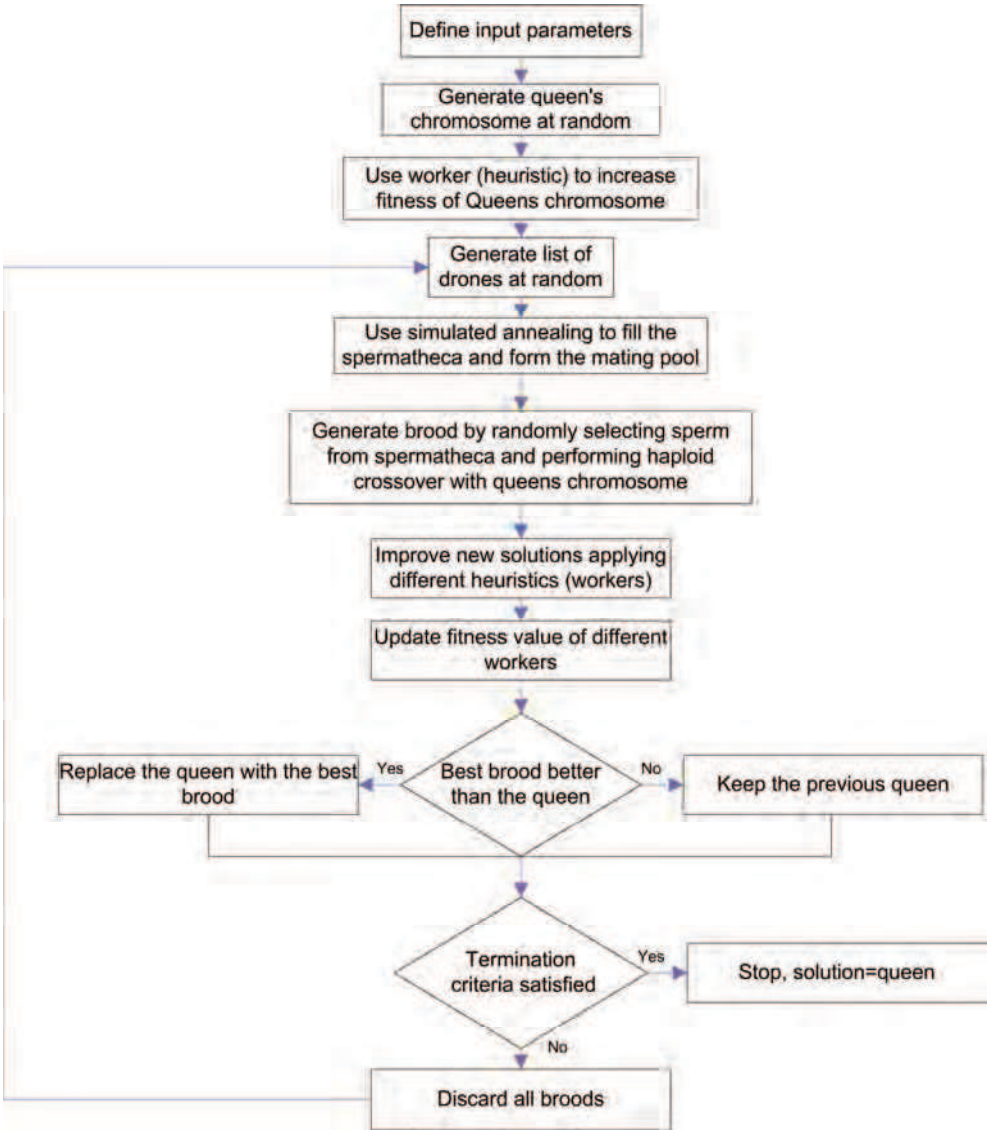


Fig. 1. Flowchart of HBMA algorithm

4. Algorithm Application to Diophantine Equation

In order to perform initial test of the algorithm, we apply the HBMA to a benchmark Diophantine problem. Diophantine equation is an algebraic function (Bull et al., 2006) which must be solved over the integers $x_i \in \mathbb{Z}$. Diophantine problems have a long pedigree in number theory. They also constitute some of the hardest problems in modern mathematics.

Behavior and results of HBMA and GA applied to the Diophantine nonlinear equation, i.e. Markoff equation:

$$x^2 + y^2 + z^2 = 3xyz \tag{5}$$

which has important applications in number theory and known solutions. This example is chosen because it is known how to generate all the solutions in a cube of given size. In the first test case, the problem is reduced to a 2*D* space by fixing *z*=433, to have a unique solution, and finding integers that satisfy:

$$x^2 + y^2 + 433^2 - 1299xy = 0 \tag{6}$$

with the search space highly complex in size, as presented with Fig.2.

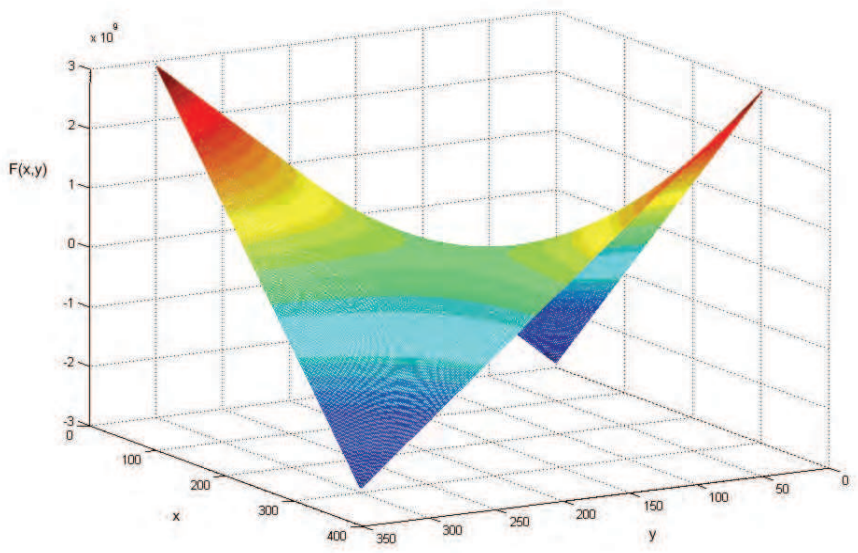


Fig. 2. Search space for the reduced Diophantine problem

5. Results for Diophantine Equation

HBMA and GA were applied to find solutions of the described problem by searching for values in the range $[0, 400]$. Both algorithms were successful finding solutions for the problem, resulting with monotonous shape of the fitness functions, as presented with Fig. 3. and Fig. 4.

The fitness function equals the value defined with eq. (6) and is normalized to the range $[0,1]$. In other words, pairs of numbers which yield lower values of fitness function have higher chances of survival, ideally approaching zero value for solution and termination criteria satisfaction.

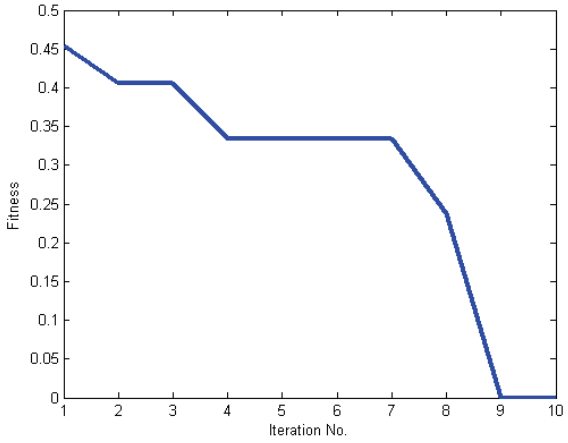


Fig. 3. Fitness value for the HBMA for Diophantine equation

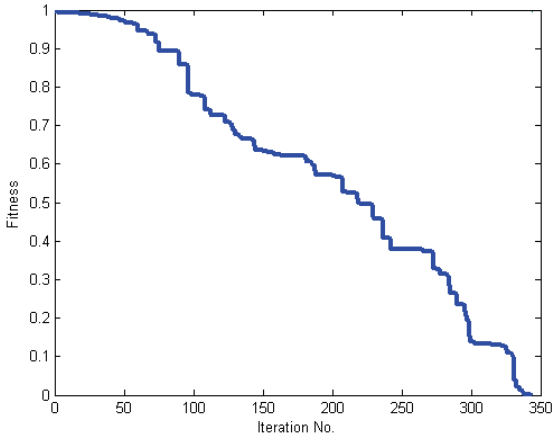


Fig. 4. Fitness value for the GA for Diophantine equation

It is important to notice that performance of the HBMA depend on the *depth of stochastic search.* In this example, only two workers i.e. different heuristics were included; namely, random walk (RW) and two point crossover (2PCO). That means that in each generation of the main loop, a number of local iterations (heuristics) take place for improvement of the brood. In our examples, depth of the local searches is set to 100 iterations.

HBMA has implicitly included elitist function, because the queen is always represented by the best chromosome found so far over all previous generations.
For the GA, 10% elitism is included, meaning that 10% of best chromosomes are directly copied to the new generation, resulting with keeping of best genetic material through the whole evolutionary search. Results and behavior of HBMA and GA are presented with Fig.3. and Fig.4. Fitness values are normalized, and it is possible to directly compare fitness values. In the case of HBMA, the search starts from initial value 0.45, what is likely to be the consequence of the first worker applied to initial queen's chromosome.

	No. of runs	No. of solutions found	Average No of generations	σ_G
GA	30	28	340	44.6
HBMA	30	30	22.9	4.6

Table 1. GA vs. HBMA performance comparison

Both algorithms were tested for 30 runs, with results presented in the Table 1. It could be summarized that HBMA outperforms GA in terms of the completeness of the solution. In terms of speed of the convergence, one should bear in mind that HBMA has inner loop with different heuristics. In each generation, there are number of iterations, defined by the depth of stochastic search, taking place.
Parameters of the GA are: crossover probability: 0.7, mutation probability: 0.01, population size: 30, survival selection: generational, initialization: random, termination condition: solution found or no fitness improvement over the last 50 generations.
Parameters of the HBMA are: sphermatheca size: M=12, stochastic search depth: 100, number of broods: 30, queens energy E and speed S randomly initialized on range $[0.5,1]$, energy reduction step $\gamma = \frac{0.5 \cdot E(t)}{M}$, heuristics included: random walk and two point crossover. Termination conditions are: solution found or no queens fitness improvement over last 50 generations.

6. Path Planning Results

HBMA algorithm is implemented to solve the problem of navigation of the mobile robot through the space containing arbitrarily distributed obstacles. The environment presentation is based on occupancy grid representation. Occupancy grids represent the world as a two-dimensional array, with each cell having particular value of 1 (if occupied) or 0 (free cell). In our study, obstacles are presented with pairs of nodes connected by mathematically defined lines. This is a more compact way of presenting obstacles which will be shown as very useful for determining collisions with the KBA. It is possible to create different obstacles as lines, or polygons, both convex or concave easily using this compact representation. To be able to treat the mobile robot a point in the environment, a minimum safety distance is added on the nodes producing a safety shadow around the actual obstacles.

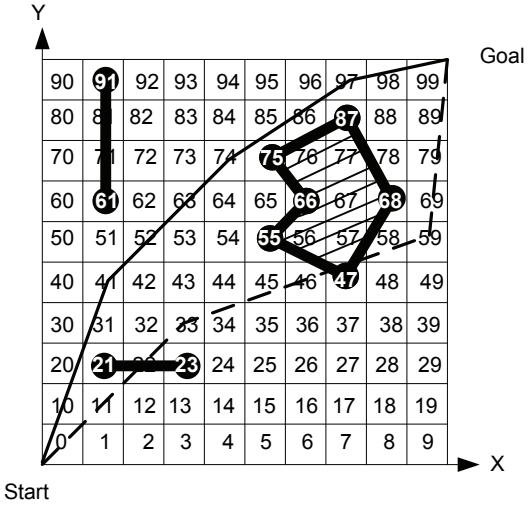


Fig. 5. Environment presented in form of occupancy grid. Numbers denote different nodes. Bold lines: obstacles; solid lines: feasible path; dashed line: unfeasible path.

One possible mobile robot environment is presented in Fig. 2. Obstacles are defined as lines connecting corresponding nodes e.g. nodes 21 and 23 are occupied and connected with the first line, making the intermediate node 22 also occupied. Nodes 55 and 47 are connected with the second line etc. Lines can create different shapes, making nodes falling into the polygons, "unavailable" for the robot. In case of vertical lines, which cannot be defined as mathematical functions since mapping x→y is not uniform, a threshold value is defined such that threshold → 0 and added on the x value of second boundary node of the line. In such manner, line is slightly rotated around the first node, without real impact on the obstacle position and mathematical consistence is preserved.

6.1 Objective Function

Impact of the objective or fitness function has a crucial role on the overall performance of the evolutionary-based algorithms. The main concept in evolutionary robotics has so far been the definition of an effective fitness function (Mermigikis & Petrou, 2006). The authors propose some kind of methodology and state that in order to achieve evolution of useful behaviours, the corresponding fitness function must have the simplest possible form (implicit), it must be possible to be calculated by means of the robot itself (intrinsic) and includes elements of the behaviour itself rather than functional details of how this can be achieved. Proper form and tuning of the parameters can significantly increase speed of the convergence and reduce the possibility of trapping in local optima. In evolutionary-based algorithms, objective function has the role of selection of individuals competing to be selected for the breeding pool and to transfer their genetic material to the new population through the offspring. In the problem being in focus here, the objective function has to reward those individuals (paths) that result in minimal number of collisions with obstacles and travel minimal distance from the start to goal position at the same time. Fitness function is presented by eq. 7:

$$Fitness\_value = \frac{w_1}{A + \sum_{i=1}^{k} a_i} + \frac{w_2}{\sum_{j=1}^{n} d_j} \tag{7}$$

Where w1 and w2 are weight constants, $w_1 \wedge w_2 > 0$, $\sum a_i$ is number of collisions of current trajectory with obstacles; $\sum d_j$ is total Euclidean distance travelled from origin to destination point for current trajectory composed of n components; A is a constant and $A > 0$.

Fitness function penalizes trajectories resulting with more collisions and larger total distance travelled. To check collisions of the trajectory i and obstacle k, two cases can occur. Case 1: a going-through node falls onto the obstacle. This situation is easy to detect and to handle. Case 2: a part of the trajectory between two consecutive going-through nodes intersects obstacle. This case is handled by solving linear systems of equations for each line segment of the trajectory and for each obstacle as a result of following system of presented with Eq. 8.

$$\mathbf{X} = \mathbf{A}^{-1} \cdot \mathbf{B} \tag{8}$$

Matrices $\mathbf{A}$ and $\mathbf{B}$ contain coefficients derived from lines that describe obstacles and line segments of current path. Matrix $\mathbf{X}$ contains solution of linear system and contains point of intersection of obstacle and linear segment of the trajectory. If intersection point of any line segment S and any obstacle O lies on that particular line segment, then trajectory τ intersects obstacle O. Otherwise obstacle O doesn't intersect trajectory τ.

Formally:

$$\begin{aligned}
&\Theta = \bigcup_{i=1}^{n} O_i \\
&p_i := \left\{ (x,y) | x \in \mathbb{R};\ y - y_< = \frac{y_> - y_<}{x_> - x_<} (x - x_<) \right\} \\
&\lambda_i(y) = \frac{y - y_{<i}}{y_{>i} - y_{<i}} \\
&O_i := \left\{ (x,y) | x \in \mathbb{R}; 0 \le \lambda_i(y) \le 1 \right\} \\
&\tau := \bigcup_{j=1}^{k} S_j \\
&\tau \cap \Theta = \bigcup_{i=1}^{n} \tau \cap O_i \\
&\tau \cap O_i = \bigcup_{j=1}^{k} S_j \cap O_i \\
&S_j \cap p_i = \{ x_{ji}, y_{ji} \} \\
&\textit{Intersection of segment } S_i \textit{ and obstacle } O_i\textit{:} \\
&S_j \cap O_i = \begin{cases} \{ x_{ji}, y_{ji} \} \textit{ for } 0 < \lambda_i(y_{ji}) \le 1 \\ \varnothing \textit{ otherwise} \end{cases}
\end{aligned} \tag{9}$$

Values of weight factors are environment dependent and determined experimentally in this study, although parameterization of environment with regards on number and distribution of the obstacles is considered for future work. This parameterization will include number of obstacles, distribution (spread or clustered) and position of obstacles in environment (along

the path connecting initial and goal position, or in corner away of main pathways). Through parameterization, correlation of form of objective function, neural architecture and presented environment could be revealed and thus efficiency of the algorithm further increased.

6.1 Simulation Results

Different environmental setups were used for the experiments. Performance of both algorithms significantly depends on the distribution of the obstacles, namely, whether obstacles are cluttered, concentrated, in the vicinity of the goal position etc. The most difficult environmental setup is when obstacles are cluttered around the proximity of the goal position.

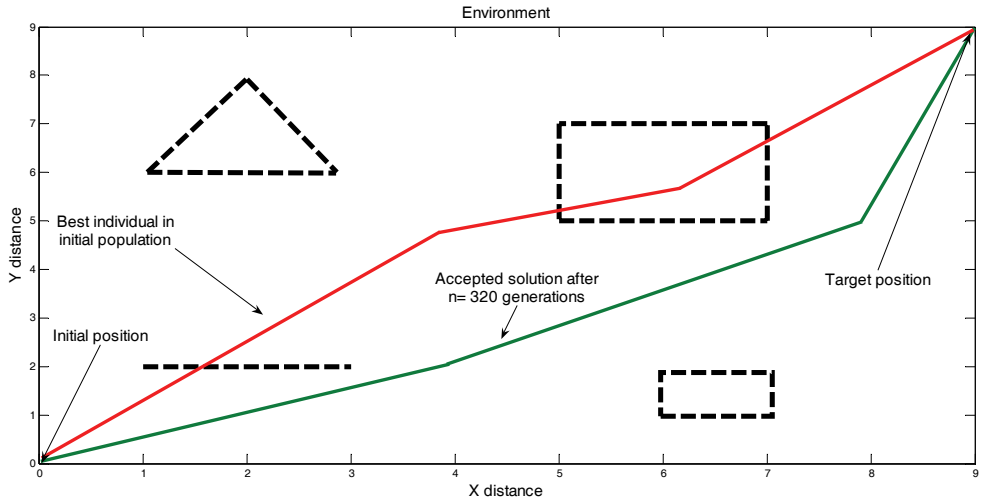


Fig. 6. Environment with obstacles, dashed, unfeasible path, red and feasible path in green colour

One possible environment setup is presented with Fig. 6. Four obstacles are present in the environment with given initial and destination position. For the environment presented with Fig.6., comparison of HBMA and GA is conducted. Results are presented in the Table 2.

	No. of runs	No. of solutions found	Average No of generations	σ_G
GA	500	478	3120	430
HBMA	500	493	430	58

Table 2. GA vs. HBMA performance comparison for path planning problem.

For simplicity, 10 x 10 grid is applied to the environments. Parameters of the GA are: Population size = 50, crossover probability: 0.8, adaptive mutation rate: start with 0.1,

increment 0.1 if no fitness improvement over 50 consecutive steps. Selection is roulette-wheel generational, with the best member of previous generation replacing the worst member of current population. Maximum length of chromosomes (degrees of freedom of trajectory) =15.
Both algorithms are able to find solutions for the presented environment with relatively high confidence. Again is the completeness (total number of the solutions found by the algorithm) slightly on the side of the HBMA. At the same time, number of iterations required is lesser for the HBMA, but CPU time is larger, because of the presence of the internal loop for the brood improvement.
Parameters of the HBMA were the same as in the Diophantine equation example. Regarding the problem of appropriate parameter selection, it is known to be difficult to tune parameters for optimal algorithm behavior, for both algorithms. Parameters were experimentally chosen..

7. Conclusions

HBMA algorithm was developed and compared with performance of the GA algorithm for two test cases. The firs test case was a benchmark Diophantine equation problem. It is shown that HBMA is comparable to the performance of well known GA in terms of CPU time, with the time slightly on the side of the GA. In terms of completeness of the solution, HBMA was able to find all solutions for the given problem, whereas GA twice did not find the solution for given termination criteria.
Similar behavior was observed for the second test case, namely collision free path planning for the mobile robot. However, it is not easy to conclude that HBMA outperforms GA in any way, since both algorithms are stochastic and dependant on the proper selection of parameters. Although both algorithms and objective were designed to be as simple as possible, to enable fair comparison, additional experiments should be performed to achieve more reliable behavior and merits for the algorithms.
HBMA could be further improved by adding additional workers (heuristics) and by monitoring success of different heuristics on different problems. GA could be improved by tailoring specific evolutionary operators for given problems.

8. References

Abass, H.A. (2001). A single queen single worker honey bees approach to 3-SAT, *Proceedings of the Genetic and Evolutionary Computation Conference,* San Francisco, 2001.

Abass, H.A. (2002). Marriage in honey bees' optimization: A haplometrics polygonus swarming approach, *Proceedings of the Congress on Evolutionary Computation,* Seoul, 2001.

Alexopulous, C.; Griffin, P.M. (1992). Path planning for a mobile robot. *IEEE Transactions on Systems, Man and Cybernetics,* Vol.22, No.2, page numbers 318-322.

Beni, G. (2004). From Swarm Intelligence to Swarm Robotics, In: *Swarm Robotics,* Erol Sahin (Ed.), 1-10, Springer LNCS, 3-540-24296-1, Berlin

Bull, P.; Knowles, A.; Tedesco, G. (2006). Diophantine benchmarks for the b-cell algorithm. *In proceedings of International Conference on Artificial Immune Systems* Canterbury, Great Britain

Chen, L.; Liu, D.Y. (1997). An efficient algorithm for finding a collision-free path among poly obstacles. *Journal of Robotics Systems, Vol.7, No.1, page numbers 129-137.*

Du, X.; Chen, H.H.; Gu, W. (2005). Neural network and genetic algorithm based global path planning in a static environment. *Journal of Zheijang University SCIENCE,* Vol.6, No.6, 2005, page numbers 549-554, ISSN 1009-3095

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison-Weseley, ISBN 02011575675, USA

Khoogar, A.R.; Parker, J.K. (1991). Obstacle Avoidance of Redundant Manipulators Using Genetic Algorithms. *Proceedings of IEEE International Conference on Robotics and Automation,* pp.317-320, Sacramento 1991

Latombe, J.C. (1991). *Robot Motion Planning,* Kluwer Academic Publishers, ISBN 0-7923-9129-2, Boston

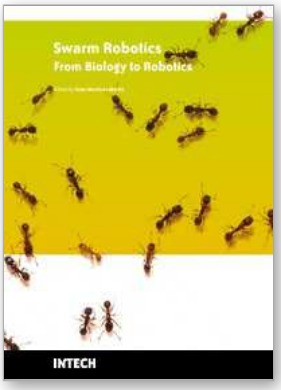
Mermigkis, I.; Petrou, L. (2006). Exploring coevolutionary relations by alterations in fitness function: Experiments with simulated robots (2006) *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol.3 No.47 , pp. 257-284.

Ram, A.; Arkin, R.; Boone, G., (1994). Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation. *Adaptive Behavior,* Vol.2., No.2, 1994, page numbers 100-107.

Ramakrishnan, R.; Zein-Sabatto, S. (2002). Multiple Path planning for a Group of Mobile Robots in a 3D Environment Using Genetic Algorithms. *Proceedings of IEEE Southeast Con,*pp.359-363, South Carolina 2002

Sadati, N., Taheri, J. (2002). Genetic Algorithm in Robot Path Planning Problem in Crisp and Fuzzyfied Environments. *Proceedings of IEEE International Conference on Industrial Technology*, pp.175-180, Bangkok 2002

Zarate, L.E.; Becker, M.; Garrido, B.D.M.; Rocha, H.S.C. (2002). An Artificial Neural Network Structure Able toObstacle Avoidance Behavior Used in Mobile Robots. Proceedings of *IEEE 28th Annual Conference of the Industrial Electronics Society*, pp.2457-2461. Spain



Swarm Robotics from Biology to Robotics

Edited by Ester Martinez Martin

ISBN 978-953-307-075-9
Hard cover, 102 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

In nature, it is possible to observe a cooperative behaviour in all animals, since, according to Charles Darwin's theory, every being, from ants to human beings, form groups in which most individuals work for the common good. However, although study of dozens of social species has been done for a century, details of how and why cooperation evolved remain to be worked out. Actually, cooperative behaviour has been studied from different points of view. Swarm robotics is a new approach that emerged on the field of artificial swarm intelligence, as well as the biological studies of insects (i.e. ants and other fields in nature) which coordinate their actions to accomplish tasks that are beyond the capabilities of a single individual. In particular, swarm robotics is focused on the coordination of decentralised, self-organised multi-robot systems in order to describe such a collective behaviour as a consequence of local interactions with one another and with their environment. This book has only provided a partial picture of the field of swarm robotics by focusing on practical applications. The global assessment of the contributions contained in this book is reasonably positive since they highlighted that it is necessary to adapt and remodel biological strategies to cope with the added complexity and problems that arise when robot individuals are considered.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821