BACTERIA COLONY APPROACHES WITH VARIABLE VELOCITY APPLIED TO PATH OPTIMIZATION OF MOBILE ROBOTS

Leandro dos Santos Coelho
Pontifical Catholic University of Paraná, LAS/PPGEPS/CCET/PUCPR, Production and Systems Engineering Graduate Program
Imaculada Conceição, 1155, Zip code 80215-901, Curitiba, Paraná, Brazil
leandro.coelho@pucpr.br

Cezar Augusto Sierakowski
Pontifical Catholic University of Paraná, LAS/PPGEPS/CCET/PUCPR, Production and Systems Engineering Graduate Program
Imaculada Conceição, 1155, Zip code 80215-901, Curitiba, Paraná, Brazil
cezar.sierakowski@pucpr.br

Abstract. *During the course of evolution, colonies of ants, bees, wasps, bacteria and termites have developed sophisticated behavior, intricate communication capabilities, decentralized colony control, group foraging strategies and a high degree of worker cooperation when tackling tasks. Utilizing these capabilities, any bio-inspired optimization techniques using analogy of swarming principles and social behavior in nature — swarm intelligence — have been adopted to solve a variety of engineering and mobile robotics problems.In this paper, new approaches of bacteria colony optimization method with variable velocity based on uniform, Gauss and Cauchy distributions were tested. Bacteria colony, a swarm intelligence methodology, is evaluated for a path planning problem in static environment of mobile robotics. The simulation results are compared with classical bacteria colony approach and genetic algorithms.*

Keywords*: swarm intelligence, bacteria colony, mobile robots, optimization, path planning.*

1. Introduction

Recently, a new class of heuristic techniques, the swarm intelligence is proposed (Bonabeau *et al*., 1999; Kennedy *et al*., 2001). In this context, more recently, biologists and computer scientists in the field of "artificial life" have been turning to insects for ideas that can be used for heuristics. Many aspects of the collective activities of social insects, such as foraging of ants, birds flocking and fish schooling are self-organizing, meaning that complex group behavior emerges from the interactions of individuals who exhibit simple behaviors by themselves (Bonabeau *et al*., 1999; Kennedy *et al*., 2001).

Swarm intelligence is an emerging research area with similar population and evolution characteristics to those of genetic algorithms. However, it differentiates in empathizing the cooperative behavior among group members. Swarm intelligence is used to solve optimization and cooperative problems among intelligent agents, mainly in artificial network training (Van den Bergh and Engelbrecht, 2001), multiobjective optimization problems (Hu and Eberhart, 2002), and cooperative and/or decentralized control (Baras *et al*., 2003).

Swarm intelligence is inspired in nature, in the fact that contribution among living animals of a group contribute with their own experiences to the group, making it stronger in face of others. The most familiar representatives of swarm intelligence in optimization problems are: food-searching behavior of ants (Dorigo and Di Caro, 1999), particle swarm optimization (Shi and Eberhart, 2000), and artificial immune system (Castro and Timmis, 2002).

In this context, the development of bio-inspired swarm intelligence methodologies based on bacteria colony behavior is an emergent research area. During the course of evolution, bacteria colonies have developed sophisticated behavior, intricate communication capabilities, decentralized colony control, group foraging strategies and a high degree of worker cooperation when tackling tasks (Shapiro, 1988; Anderson and McShea, 2001). A particularly interesting group behavior has been demonstrated for several motile species of bacteria, including *Escherichia coli* and *Salmonella typhimurium*, where intricate stable spatio-temporal patterns (swarm) based on stimuli of cell-cell signaling and foraging are formed in semi-solid nutrient media. Chemotaxis is a bias of movement according to the gradient of a chemical agent. Chemotactic signaling is a chemotactic response to an agent emitted by the bacteria (Budrene and Berg, 1995). Basically, chemotaxis is a foraging behavior that implements a type of optimization where bacteria try to climb up the nutrient concentration and avoid noxious substances and search for ways of neutral media. Based on these biological concepts, the definition of an optimization model of *E. coli* bacterial foraging is possible. In this work, the bacteria colony optimization method proposed by Passino (2002) is modified for present velocity variable using uniform, Gauss, and Cauchy probability distribution functions for the movement of bacteria.

The contribution of this paper is to present a new approach of bacteria colony optimization method for problem of path planning with static environment and obstacles. Simulation results for a case study of path planning using bacteria colony are compared with genetic algorithms, a classical approach of evolutionary computation area.

The paper is organized as follows. The fundamentals of bacteria colony are detailed in section 2. In section 4, the path planning of mobile robots is described. The simulation results and conclusions are commented in section 5 and 6, respectively.

2. Bacteria colony

Natural selection tends to eliminate animals with poor foraging strategies and to favor gene propagation of those with good foraging strategies, once these have higher chances of succeeding in reproduction. These evolutionary principles have taken scientists to develop the foraging strategies, turning it appropriate to optimization models (Passino, 2002).

The presence of flagellum allows the bacteria to move, the movement is acquired through the flagellum rotation in the same direction, at a rotating velocity of 100 to 200 rotations per second. Bacteria may move in two different forms: they might run (swim for a period of time), movement achieved by the flagellum rotation counter clockwise, or they can tumble, achieved by the flagellum rotation clockwise. Bacteria switch between these two modes of operation during its entire lifetime (rarely the flagellum stops rotating).

After a run period, a bacterium tumbles, the tumble interval is about 0.14 ± 0.19 s, according to Passino (2002). After the tumble, the bacterium is pointed in a random direction. When the flagellum are rotated counter clockwise, the bacterium will move towards the direction it's turned, at an average velocity of 10–20 μm/s, meaning, about 10 times its length by second, for a mean interval of 0.86 ± 1.18 s.

The local environment where bacteria live might change, either gradually or suddenly. So bacteria can suffer a process of elimination, through the appearance of a noxious substance, or to disperse, through the action of another substance, generating the effects of elimination and dispersion.

A bacterium position, in a time instant, can be determined through equation (1), where the position in that instant is calculated in terms of the position in the previous instant and the step size $C(i)$ applied in a random direction $\phi(j)$, generated in the bacterium tumble,

$$\theta'(j+1,k,l) = \theta'(j,k,l) + C(i) * \phi(j,k,l) \tag{1}$$

To adapt such strategy to optimization problems, an equation to determinate the cost of each position is needed, to possibility the comparison between the position and the environment. The cost is determined by the equation,

$$J(i,j,k,l) = J(i,j,k,l) + J_{cc}(\theta'(j,k,l), P(j,k,l)). \tag{2}$$

Through equation (2) is noticed that the cost (fitness) of a determined position $J(i,j,k,l)$ is also affected by the attractive and repulsive forces existing among the bacteria of the population given by $J_{cc}(\theta'(j,k,l), P(j,k,l))$.

After a determined number of chemotactic steps (steps comprehending the movement and the cost determination of each bacterium position), a reproductive step occurs. In this reproductive step the bacterium are sorted decreasingly by their cumulative cost. The lower half of the list die, these are the bacteria that couldn't gather enough nutrients during the chemotactic steps, and the upper half divide themselves into two new bacteria, located in the same position.

Summarizing, the term taxis refers to the locomotor response of a cell to its environment. In a taxis, a cell responds such that it changes both direction and the duration of the next movement step. The tactic response requires some directional information from the environment that bacteria obtain by comparing an environmental property at two different time steps. If the tactic response is related to information about chemical concentrations (that may be either attractants or repellents), it is called chemotaxis (Brandstätter and Baumgartner, 2002; Passino, 2002).

In Fig. 1 (shown at the end of the article) the algorithm is presented in pseudo code. As seen in the pseudo code, the bacteria colony algorithm is basically composed by an elimination and dispersal loop, inside this loop, there is another one, who is responsible for the bacteria reproduction. Inside this one, there is a third loop, responsible for generating the direction in which each bacterium will run, determining the period the bacterium will move and, as a consequence, determining it's position after the loop execution, and calculating the fitness of these positions. The reproductive loop is responsible for determining which of the bacteria must reproduce and which must be exterminated after the movements executed in loop 3, through a cost analysis of their positions along their movement. The first loop is responsible for eliminating some bacteria, it's ruled by an elimination probability, repositioning them into another random position of the search space.

2.1 New Bacteria Colony Approaches for Velocity of the Movement Setup

The parameter $C(i)$, i=1,2,...S regulates the velocity of the movement taken in one step of bacteria colony, where S is population size. In this work, new approaches for the setup of parameter $C(i)$ are proposed.

In this paper, new approaches to bacteria colony (BC) are proposed which are based on the studies of mutation operators in fast evolutionary programming (Yao and Liu, 1996; Chellapilla, 1998) and fast particle swarm optimisation (Coelho and Krohling, 2003) for the setup of velocity parameter $C(i)$ of bacteria. The aim is to modify the $C(i)$ constant in conventional bacteria colony proposed by Passino (2002) to use it with uniform, Gauss or Cauchy distribution. The use of Cauchy distribution in evolutionary algorithms could be useful to escape of local minima, while the Gauss distribution (normal distribution) could provide a faster convergence in local searches.

The new BC approaches tested are based on the following configurations:

- BC(1): $C(i)=3$, $i=1,2,...S$;
- BC(2): $C(i)=4$, $i=1,2,...S$;
- BC(3): $C(i)=5$, $i=1,2,...S$;
- BC(4): $C(i)=6$, $i=1,2,...S$;
- BC(5): $C(i)=2\cdot rand(\cdot)+3$, $i=1,2,...S$, where $rand(\cdot)$ are random numbers in the range [0,1] generated according to a uniform probability distribution, i.e., $C(i)$ is setup in the range [3, 5];
- BC(6): $C(i)=4\cdot rand(\cdot)+2$, $i=1,2,...S$, where $rand(\cdot)$ are random numbers in the range [0,1] generated according to a uniform probability distribution, i.e., $C(i)$ is setup in the range [2, 6];
- BC(7): $C(i)=7\cdot rand(\cdot)+1$, $i=1,2,...S$, where $rand(\cdot)$ are random numbers in the range [0,1] generated according to a uniform probability distribution, i.e., $C(i)$ is setup in the range [1, 8];
- BC(8): $C(i)= 4\cdot|Gauss(\cdot)|$, $i=1,2,...S$, where $Gauss(\cdot)$ are random numbers generated according to a *Gaussian* probability distribution with zero mean;
- BC(9): $C(i)=2\cdot Gauss(\cdot)+3$, $i=1,2,...S$, where $Gauss(\cdot)$ are random numbers generated according to a *Gaussian* probability distribution with zero mean;
- BC(10): $C(i)=4\cdot Gauss(\cdot)+2$, $i=1,2,...S$, where Gauss(·) are random numbers generated according to a Gaussian probability distribution with zero mean;
- BC(11): $C(i)=7\cdot Gauss(\cdot)+1$, $i=1,2,...S$, where Gauss(·) are random numbers generated according to a *Gaussian* probability distribution with zero mean;
- BC(12): $C(i)= 8\cdot|Cauchy(\cdot)|$, $i=1,2,...S$;
- BC(13): $C(i)=4\cdot|Cauchy(\cdot)|$, $i=1,2,...S$, where $Cauchy(\cdot)$ are random numbers generated according to a *Cauchy* probability distribution with zero mean;
- BC(14): $C(i)=0{,}5\cdot|Cauchy(\cdot)|$, $i=1,2,...S$, where $Cauchy(\cdot)$ are random numbers generated according to a *Cauchy* probability distribution with zero mean;
- BC(15): $C(i)=|Cauchy(\cdot)|$, $i=1,2,...S$, where $Cauchy(\cdot)$ are random numbers generated according to a *Cauchy* probability distribution with zero mean;
- BC(16): $C(i)=2\cdot|Cauchy(\cdot)|$, $i=1,2,...S$, where $Cauchy(\cdot)$ are random numbers generated according to a *Cauchy* probability distribution with zero mean;
- BC(17): $C(i)=2\cdot Cauchy(\cdot)+3$, $i=1,2,...S$, where Gauss(·) are random numbers generated according to a *Cauchy* probability distribution with zero mean.

3. Trajectory planning of mobile robots

Several applications of path planning to solve trajectory planning in presence of static and/or dynamic obstacles in robotic systems can be found in the literature (Tu and Yang, 2003; Bennewitz *et al*., 2002; Melchior *et al*., 2003). One of the most popular planning methods is the artificial potential field (Tsuji *et al*., 2002). However, this method gives only one trajectory solution that may not be the smaller trajectory in a static environment. The main difficulties in determining the optimum trajectory are due to the fact that analytical methods are extremely complex to be used in real time, and the searching enumerative methods are excessively affected by the size of the searching space.

Recently, the interest in using evolutionary algorithms has increased, genetic algorithms are used in mobile robots trajectory planning, generally when the search space is large (Fujimori *et al*., 1997; Xiao *et al*., 1997; Liu and Wu, 2001; Gemeinder and Gerke, 2003).

The trajectory planning is the main aspect in the movement of a mobile robot. The problem of a mobile robot trajectory planning is typically formulated as follows: given a robot and the environment description, a trajectory is planned between two specific locations that is free of collisions and is satisfactory in a certain performance criteria (Fujimori *et al*., 1997).

Seeing the trajectory planning as an optimization problem is the boarding adopted in this article. In this case, a sequence of configurations that moves the robot from an initial position (origin) to a final position (target) is designed.

A trajectory optimizer must locate a series of configurations that avoid collisions among the robot(s) and the obstacle(s) existing in the environment. The optimizer must also try to minimize the trajectory length found, in order to be efficient. The search space is the group of all possible configurations.

In the present study, it's considered a 2-dimensional trajectory' planning problem for mobile robot, in which the position of the mobile robot R is represented by Cartesian coordinates (x, y) in the xy plane. The initial and destination points of the robot are (x_0, y_0) and (x_{np}, y_{np}), where n_p is a design parameter. The initial point is always (0,0).

Only the trajectory' planning problem is empathized in this paper, the robot control problem is not the focus of this paper. However, details of the robots movement equations can be found in Fujimori *et al.* (1997). It's assumed that the obstacles are circular in the robot's moving plan. Besides, the hypothesis that the free 2-dimensional space is connected and the obstacles are finite in size and do not overlap the destiny point is true.

The optimization problem formulated consists of a discrete optimization problem, where the objective function $f(x,y)$, which is the connection between the technique used for optimization and the environment, aims to minimize the total trajectory traveled by the mobile robot and is ruled by

$$f(x,y) = \alpha d_{obj} + \lambda n_o \tag{3}$$

$$d_{obj} = \sum_{i=0}^{n_p} \sqrt{\left(x(i+1) - x(i)\right)^2 + \left(y(i+1) - y(i)\right)^2} \tag{4}$$

where α and λ are ponderative factors, d_{obj} representes the euclidian distance between the initial and the destiny point, n_0 denotes the number of obstacles hitten by the robot movement following the planned trajectory, and n_p is the number of points where a trajectory change occurs (project parameter in this work). It's noticed by the equation (3) that an λ term exists, it's a ponderative (penalty) term for unfeasible solutions, meaning, the trajectory that intercepts obstacles. In this case, the fitness function to be evaluated by the bacteria colony aims to maximize

$$fitness = \frac{K_c}{f(x,y) + \varepsilon} \tag{5}$$

where K_c and ε are scale constants.

4. Simulation results

The environment used for the trajectory planning is a 100x100 meters field. The search interval of the parameters is $x_i \in [0,100]$ meters and $y_i \in [0,100]$ m, where i=1,..,np. About the fitness it's adopted α=1, λ=200, K_c =100 and ε=1x10^{-6}. A simulated case and the results achieved by the bacteria colony approaches and genetic algorithms (GAs) are presented.

GAs are composed by a population of individuals and a set of operators, these operators are inspired in biology and are applied in the population. According to the evolutionary theories, the elements that most fit (best fitness) to their environment have a higher chance to survive and to reproduce, transmitting their genetic material towards new generations. The procedure of optimization based on GA is basically composed by following steps (Goldberg, 1989):

(i) generation of a initial population;
(ii) evaluation of each of the elements of the population;
(iii) selection of the best elements (most fit) of the population;
(iv) genetic manipulation, through crossover and mutation operators, creating a new population;
(v) go to step (ii) until that a stop criterion be satisfied.

The GAs setup used to a comparative study with bacteria colony approaches and to simulate the case study have population size 15, size of each chromosome (representation of individual by binary strings) 16 bits, maximum number of generations 200, crossover probability 0.80, and the selection operator is roulette wheel. Four configurations of GA were tested:

- GA(1): mutation probability 0.10 with elitist strategy (hold the best chromosome in the next generation);
- GA(2): mutation probability 0.20 with elitist strategy;
- GA(3): mutation probability 0.10 without elitist strategy; and
- GA(4): crossover probability 0.20 without elitist strategy.

For the bacteria colony algorithm the following parameters needed to be adjusted p (optimization problem's dimension), S (population size), N_c (number of chemotactic steps), N_s (maximum number of steps that a bacterium can swim in a turn), N_{re} (number of reproductions), N_{ed} (number of elimination-dispersals events), p_{ed} (elimination-dispersal probability) and $C(i)$, i=1,2,...S (velocity of the movement taken in one step) it is adopted, S=15, N_c=5, N_s =5, N_{re}=10, N_{ed} =2 and p_{ed} = 0.3 for the tested approaches.

4.1. Case study with 12 obstacles

In Table 1 are presented the center positions (x_c, y_c) of the circular obstacles and their respective radius (in meters) for this case study. The results obtained are restricted to p=4. In Table 2 the results for the case study are summarized.

Table 1. Obstacles for case study with 12 obstacles.

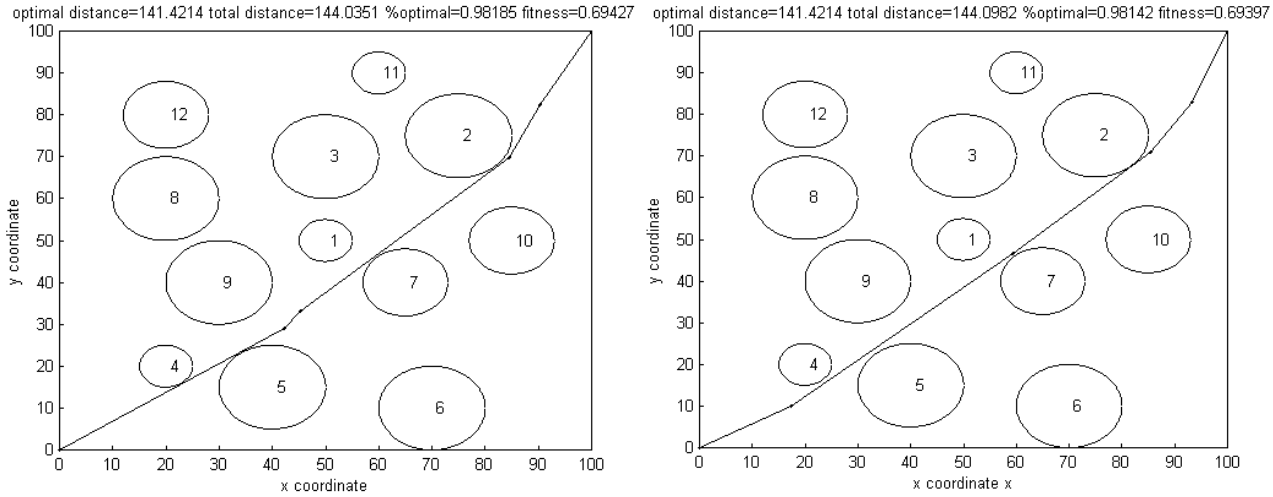
Obstacle number	Radius	Position (x_c, y_c)
1	05	(50, 50)
2	10	(75, 75)
3	10	(50, 70)
4	05	(20, 20)
5	10	(40, 15)
6	10	(70, 10)
7	08	(65, 40)
8	10	(20, 60)
9	10	(30, 40)
10	08	(85, 50)
11	05	(60, 90)
12	08	(20, 80)

Table 2. Results for an environment with 12 obstacles for 10 experiments.

Fitness / Method	$C(i)$ of BC	Maximum	Mean	Minimum	Median	Standard Deviation
BC(1)	constant	0.6723	0.5574	0.2871	0.5609	0.1100
BC(2)	constant	0.6775	0.5892	0.4178	0.6030	0.0852
BC(3)	constant	0.6045	0.5788	0.5449	0.5875	0.0215
BC(4)	constant	0.6513	0.5891	0.5551	0.5821	0.0310
BC(5)	uniform	0.6924	0.5709	0.2803	0.5799	0.1125
BC(6)	uniform	0.6704	0.5795	0.5286	0.5646	0.0398
BC(7)	uniform	0.6783	0.6016	0.5514	0.6000	0.0430
BC(8)	Gauss	0.6903	0.6084	0.5481	0.5831	0.0551
BC(9)	Gauss	0.6894	**0.6270**	0.5572	**0.6304**	0.0523
BC(10)	Gauss	0.6669	0.5852	0.4904	0.5942	0.0442
BC(11)	Gauss	0.6156	0.5656	0.5545	0.5588	**0.0182**
BC(12)	Cauchy	0.6924	0.5861	0.2781	0.5908	0.1235
BC(13)	Cauchy	0.6766	0.5866	0.2915	0.6109	0.1120
BC(14)	Cauchy	0.5529	0.2382	0.1224	0.2158	0.1257
BC(15)	Cauchy	0.6928	0.4927	0.2455	0.5451	0.1588
BC(16)	Cauchy	**0.6943**	0.4713	0.1827	0.5523	0.1902
BC(17)	Cauchy	0.6883	0.6180	0.5646	0.6117	0.0431
GA(1)	-	0.6939	0.6269	**0.5818**	0.6150	0.0509
GA(2)	-	0.6940	0.6214	0.5621	0.6199	0.0518
GA(3)	-	0.6940	0.6240	0.5621	0.6201	0.0517
GA(4)	-	0.6940	0.6245	0.5718	0.6210	0.0566

In comparison with GA, the BC(16) found the best solution (maximum fitness) for the case study presented in table 2. However, it can be seen that the solution found by BC(9) presents the best mean and median fitness (best convergence) than the results obtained by BC(16) and GA(1)-(4).

In terms of computational cost, the GA configurations achieved the best computational time than the BC approaches applied here. Bacteria colony algorithm was implemented using Matlab 5.2, and took, in average, 62.14 seconds to run in a PC-compatible with AMD 1.09 GHz processor and 124 MB RAM. GA was, in average, 34.70 seconds in the same computer for run each experiment. GA(1)-(4) provided good results, with a mean fitness very similar to the solution found with the BC(16). In Figs. 2(a) and 2(b), the best solutions achieved by bacteria colony and GA are presented.



(a) BC(16) (b) GA(1)

Figure 2. Best results achieved by bacteria colony and GA after 10 experiments using BC(16) and GA(1).

The best result achieved for the case study using BC(16) were achieved by the coordinates:
P_1 = (90.3782, 84.6237);
P_2 = (82.3903, 69.8998);
P_3 = (45.1662, 33.0802);
P_4 = (42.2871, 29.1160).

5. Conclusion and future works

A research area with special relevance to mobile robot systems is devising suitable methods to plan optimum moving trajectories. There exist many approaches within the area of evolutionary computation and swarm intelligence to solve the problem of optimization of path planning in mobile robotics. In this paper the application of the genetic algorithms and bacteria colony is explored for this purpose.

In this paper, new approaches of bacteria colony optimization method with variable velocity based on uniform, Gauss and Cauchy distributions were tested. Bacteria colony based on Gauss distribution, BC(9), presents best result of convergence and the BC(16) found the best solution (maximum fitness) for the case study analyzed. The simulation results of bacteria colony were compared with GA approaches. In terms of computational cost, the GA approaches achieved the best computational time that the BC methods applied in path planning optimization.

These heuristic optimization methods were successfully used to obtain path planning of a mobile robot for a case study of static obstacles. The results of these simulations are very encouraging and they indicate important contributions to the areas of swarm intelligence and path planning in robotics. Furthermore, as a continuation of this research, more detailed studies related to the parameters related to the two techniques, specially related to the bacteria colony.

6. Acknowledgements

The authors gratefully acknowledge support from “Fundação Araucária of Paraná” (Araucaria Fundation of Parana state), Brazil, through project with protocol 6234 (005/2005).

7. References

Anderson, C. and McShea, D. W., 2001, “Individual versus Social Complexity, with Particular Reference to Ant Colonies”, Biological Reviews, Vol. 76, pp. 211-237.

Baras, J.S., Tan, X., and Hovareshti, P., 2003, “Decentralized Control of Autonomous Vehicles”, Proceedings of the 42nd IEEE Conference on Decision and Control, Maui, Hawaii, USA, pp. 1532-1537.

Bennewitz, M., Burgard, W. and Thrun, S., 2002, “Finding and Optimizing Solvable Priority Schemes for Decoupled Path Planning Techniques for Teams of Mobile Robots”, Robotics and Autonomous Systems, Vol. 41, No. 2, pp. 89-99.

Bonabeau, E., Dorigo, M. and Theraulaz, G., 1999, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, New York.

Brandstätter, B. and Baumgartner, U., 2002, Particle Swarm Optimization – Mass-Spring Systems Analogon, IEEE Transactions on Magnetics, Vol. 38, No. 2, pp. 997-1000.

Budrene, E.O. and Berg, H.C., 1995, "Dynamics of Formation of Symmetrical Patterns by Chemotactic Bacteria", Nature, Vol. 376, pp. 49-53.
Castro, L. N., and Timmis, J.I., 2002, Artificial Immune Systems: A New Computational Intelligence Approach, Springer-Verlag, London, UK.
Chellapilla, K., 1998, "Combining Mutation Operators in Evolutionary Programming", IEEE Transactions on Evolutionary Computation, Vol. 2, No. 3, pp. 91-96.
Coelho, L. S. and Krohling, R. A., 2003, "Predictive Controller Tuning using Modified Particle Swarm Optimisation based on Cauchy and Gaussian Distributions", Proceedings of the 8th On-line World Conference on Soft Computing in Industrial Applications, WSC8, Dortmund, September.
Dorigo, M., and Di Caro, G., 1999, "The Ant Colony Optimization Meta-Heuristic", New Ideas in Optimization, Edited by D. Corne, M. Dorigo, and F. Glover, McGraw-Hill, pp. 11-32.
Fujimori, A., Nikiforuk, P.N. and Gupta, M.M., 1997, "Adaptive Navigation of Mobile Robots with Obstacle Avoidance", IEEE Transactions on Robotics and Automation, Vol. 13, No. 4, pp. 596-602.
Gemeinder, M. and Gerke, M., 2003, "GA-based Path Planning for Mobile Robot Systems Employing an Active Search Algorithm", Applied Soft Computing, Vol. 3, pp. 149-158.
Goldberg, D.E., 1989, Genetic Algorithms in Search, Optimization and Machine Learning, Reading: MA: Addison-Wesley.
Hu, X. and Eberhart, R.C., 2002, "Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization", Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, pp. 1677-1681.
Kennedy, J.F., Eberhart, R.C., and Shi, R.C., 2001, Swarm Intelligence, San Francisco: Morgan Kaufmann Pub.
Liu, G. -Y. and Wu, 2001, "A Discrete Method for Time-Optimal Motion Planning of a Class of Mobile Robots", Journal of Intelligent and Robotic Systems, Vol. 32, pp. 75-92.
Melchior, P., Orsoni, B., Lavaialle, O., Poty, A., and Oustaloup, A., 2003, "Consideration of Obstacle Danger Level in Path Planning Using A* and Fast-Marching Optimization: Comparative Study", Signal Processing, Vol. 83, pp. 2387-2396, 2003.
Passino, K.M., 2002, "Biomimicry of Bacterial Foraging for Distributed Optimization and Control", IEEE Control Systems, Vol. 22, No. 3, pp. 52-67.
Shapiro, J.A., 1988, "Bacteria as Multicellular Organisms", Scientific American, Vol. 256, No. 6, pp. 62-69.
Shi, Y. and Eberhart, R.C., 2000, "Experimental Study of Particle Swarm Optimization", Proceedings of 4th World Conference on Systems, Cybernetics and Informatics, Orlando, FL, USA.
Tsuji, T., Tanaka, Y., Morasso, P.G., Sanguineti, V., and Kaneko, M., 2002, "Bio-mimetic Trajectory Generation of Robots via Artificial Potential Field with Time Base Generator", IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews, Vol. 32, No. 4, pp. 426- 439.
Tu, J., and Yang, S.X., 2003, "Genetic Algorithm Based Path Planning for a Mobile Robot", Proceedings of the IEEE International Conference on Robotics & Automation, Taipei, Taiwan, pp. 1221-1226.
Van den Bergh, F. and Engelbrecht, A. P., 2001, "Training Product Unit Networks Using Cooperative Particle Swarm Optimisers", Proceedings of the INNS-IEEE International Joint Conference on Neural Networks, Washington, DC, USA.
Xiao, J., Michalewicz, Z., Zhang, L., and Trojanowski, K., 1997, "Adaptive Evolutionary Planner/Navigator for Robots", IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, pp. 18-28.
Yao, X. and Liu, Y., 1996, "Fast Evolutionary Programming", Proceedings of 5th Annual Conference on Evolutionary Programming, San Diego, CA, pp. 451-460.

8. Responsibility notice

The authors are the only responsible for the printed material included in this paper.

DO { $l = l + 1$;
DO { $k = k + 1$;
DO { $j = j + 1$;
FOR EACH bacterium i {
Calculate $J(j,k,l)$;
Assume $J(j,k,l) = J(j,k,l) + J_{cc}(\phi'(j,k,l), P(j,k,l))$;
Save $J_{last} = J(i,j,k,l)$;
Generate a random vector $\Delta(i) \in \Re^{\rho}$, with real numbers, within $[-1,1]$;
Move $\theta'(j+1,k,l) = \theta'(j,k,l) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^{T}(i) * \Delta(i)}}$; % tested new approaches for $C(i)$
Calculate $J(i, j+1, k, l)$;
Assume $J(i, j+1, k, l) = J(i, j+1, k, l) + J_{cc}(\theta'(j+1,k,l), P(j+1,k,l))$;
Assume $m = 0$;
DO {
Assume $m = m + 1$;
IF $J(i, j+1, k, l) > J_{last}$ (optimization problem)
THEN
Assume $J_{last} = J(i, j+1, k, l)$;
Calculate $\theta'(j+1,k,l) = \theta'(j+1,k,l) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^{T}(i) * \Delta(i)}}$;
Calculate $J(i, j+1, k, l)$;
Assume $J(i, j+1, k, l) = J(i, j+1, k, l) + J_{cc}(\theta(j+1,k,l), P(j+1,k,l))$;
ELSE
Assume $m = N_s$;
} WHILE $m < N_s$;
}
} WHILE $j < N_c$;
FOR EACH bacterium i { Calculate $J^{i}_{Health} = \sum_{j=1}^{N_S - 1} J(i,j,k,l)$; }
Sort the bacteria, according to the valor of J_{Health};
Kill the bacteria with the smaller value of J_{Health};
Duplicate the bacteria with the higher values of J_{Health};
} WHILE $k < N_{re}$;
FOR EACH bacterium i {
Eliminate and disperse bacterium with a probability of p_{ed}; }
} WHILE $l < N_{ed}$.

Figure 1. Pseudo code of the foraging theory applied to a bacteria colony.