Jumping Frogs Optimization:
a new swarm method for discrete optimization

F. Javier Martínez García, José A. Moreno Pérez

Grupo de Computación Inteligente.
Departamento de Estadística, I.O. y C.
Instituto Universitario de Desarrollo Regional.
Universidad de La Laguna.
{jamoreno,fmartinz}@ull.es

DOCUMENTO DE TRABAJO – DEIOC 3/2008

Abstract

We propose a new Swarm Intelligence optimization method for discrete problems. The procedure is similar to the PSO, however we do not use the velocity concept but we keep the notion of attraction of the leaders. The effect of the attractions on the particles consists in a series of improving moves approaching the attractor. The method also includes a local search to improve the evolving solutions. The described method is applied for the discrete p-median problem. It is an NP-Hard problem for which most of the metaheuristics and nature-inspired procedures have been applied. The procedure is compared with the known discrete versions of the PSO for this problem.

Keywords. Swarm Intelligence. Particle Swarm Optimization. *p*-Median. Metaheuristics. Logistics.

1. Introduction

Swarm intelligence [12], [13] is a new computational intelligence paradigm combining ideas derived the social behaviour in nature. Swarm optimization comprises several tools derived from the collective intelligence of a decentralized group of individuals. The main swarm optimization methods are Ant Colony Optimization (ACO) [9] and Particle Swarm Optimization (PSO) [6].

Particle Swarm Optimization, PSO is a promising relatively recent metaheuristic introduced by James Kennedy and Russel Eberhat [10], [18]. It is a evolutionary method inspired by the social behaviour of individuals within swarms in nature, like flocks of birds or banks of fish. In this method a swarm of particles fly in the virtual space of the possible solutions that interact. The movement of the particles is conducted by the inertia, its memory and the attraction of the positions with the best performance. Each particle has associated a position in the solution space and a velocity or rate of change. The particles remember at which position they achieved their highest performance. Each particle communicates with a subset of the swarm that constitutes its neighbourhood that can change dynamically. Every particle can also get which particle achieved the best overall position in its neighbourhood. The dynamic of the swarm is influenced by the attractors that, in some models include in addition of the best position of the own particles and the best position of their neighbourhood, the global best position.

Since there is not possible to thrown to fly particles in a discrete space, several adaptations to discrete problems have proposed, known as *Discrete Particle Swarm Optimization,* DPSO. In this work we propose a swarm inspired optimization method for discrete problems. Since inertia and velocity lose sense in a discrete space we do not consider such components but include a local search. We consider an implementation of this method that takes account the real separation between two solutions considers.

The following sections introduce the standard version of the PSO and review the discrete PSO (DPSO) appeared in literature. We describe the new method that avoid the notion of inertia and velocity and keep the use of best position as attractors. Finally we describe the computational experience made. The paper ends with some brief conclusions.

2. Particle Swarm Optimization

The Particle Swarm Optimization (PSO) is inspired by the continuous movement of the particles that form swarm in nature subject to the effect of the inertia and of the attraction of the best members that lead the swarm. Two of the most important characteristics in the development of this metaheuristic are the facility of implementation and the use of the evolution of the social relations as a computational model. In a PSO heuristic, particles of the swarm are interpreted like search agents who cross the space of solutions. The PSO [6] was initially proposed as a procedure to solve optimization problems with continuous variables.

For an optimization problem with d continuous variable, each particle i of the swarm $S = \{1, 2, \ldots, s\}$ has associated its position vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{id})$ and its velocity (or rate of change) vector $\boldsymbol{v}_i = (v_{i1}, \ldots, v_{ij}, \ldots, v_{id})$. Each particle i of the swarm communicates with a subset of S or environment or neighbourhood $N(i) \subseteq S$ that can vary dynamically. Each particle keeps and uses information of its better position during the process search. Also it can obtain the best position reached among the particles of his social neighbourhood, which can be the whole swarm or a part of it. The information of the best positions influences in the behaviour of particles. In all these cases the value of the objective function like the adaptation function or fitness is also stored.

The initial positions and velocities of particles are usually obtained randomly within some limits. At every iteration, the particles update their position and velocity by means of recurrent equations. The position is modified using exclusively its velocity, but in the updating of the velocity they take into account, in addition to the value of the own velocity, the best position of the own particle and the best position of the group of particle of the swarm to which it is related; its social neighbourhood or neighbourhood. These best positions, individual and joint, act with different weights, like centres of attraction for particles. In the standard PSO, procedure according to the proposal of Kennedy and Eberhart [20], the vector equations to update the position $\boldsymbol{x}_i$ and velocity $\boldsymbol{v}_i$ of each particle of the swarm, are the following:

$$\boldsymbol{v}_i = c_1\, \boldsymbol{v}_i + c_2\, \xi\, (\boldsymbol{b}_i - \boldsymbol{x}_i) + c_3\, \xi\, (\boldsymbol{g}_i - \boldsymbol{x}_i)$$

$$\boldsymbol{x}_i = \boldsymbol{x}_i + \boldsymbol{v}_i$$

where vectors $\boldsymbol{b}_i$ and $\boldsymbol{g}_i$ are the best position than has had this particle since the procedure started and the best position between those than has had all particles of the group or social neighbourhood of this particle. The parameter c_1 represents the effect of the inertia whose mission is to control the magnitude of the velocity and to avoid that it grows indefinitely. The values c_2 and c_3 are the weights that represent the degree of confidence of the particle in, itself and its social group, that in many versions are equals. The symbol ξ refers to a random number with uniform distribution in [0,1] that is independently generated each time. These values are usually positive and inferior to one; in most versions $c_1 = 1$, or $c_2 = c_3$ or even $c_1 + c_2 + c_3 = 1$. In the standard version PSO-2007[1] proposed by Maurice Clerc, the values for these parameters are set to $c_1 = 1/(2+ln2) = 0.721$ and $c_2 = c_3 = 0.5 + ln2 = 1.193$. In this version the size of the neighbourhoods is $K = 3$ and the size of the swarm is fixed to $10 + 2\sqrt{d}$ where d is the dimension of the space of solutions. In the former standard version PSO-2006[2], the structure of neighbourhoods is obtained as follows. Since, the number K of particles that informs to each other is fixed, the particles that constitute the neighbourhood of each particle are chosen at random. The neighbourhoods are newly generated at the iterations when the best global position g^* does not improve. In the standard version PSO-2007 structure of neighbourhoods is obtained at random from the fixed number K of informant as follows. Each possible link is activated with probability $p = 1 - (1 - 1/s)^K$.

In addition to the random selection of neighbourhoods, the two most often topologies for the structure of neighbourhood are the ring and the star topologies. With the ring structure, each particle interacts with the previous one and following one (in a cyclic arrangement); i.e., $N(1) = \{ s, 1, 2 \}$, $N(s) = \{ s-1, s, 1 \}$ and $N(i) = \{ i-1, i, i+1 \}$, for $1 < i < s$. In the star structure each particle interacts with all particles of the swarm; $N(i) = S$, for $1 \le i \le s$.

Parameter selection for PSO has been considered in [11], [14], [34] and [37]. The effect of the neighbourhood structure has been analysed in [17]

3. Discrete PSO

The PSO was originally proposed for optimization problems with continuous variables, but since, in words of its creators, it is not possible "to throw to fly" particles in a discrete space, several ways have been proposed to adapt the methodology to discrete problems. A group of individuals (agents search) that cross a discrete space simultaneously jumping from solutions to solution without sinking in intermediate positions naturally recalls the behaviour of a group frogs jumping from stone to stone in a pool. The corresponding bio-inspired method is named *Jumping Frog Optimization* (JFO) and the main difference with PSO is that the inertia and velocity components are replaced by sporadically random jumps.

The DPSO proposed by Kennedy and Eberhat [19] for optimization problems with binary variables was the first method named DPSO (Discrete Particle Swarm Optimization). In this method, the position of each particle is a binary vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{id})$ in the d-dimensional binary space, $\boldsymbol{x}_{ij} \in \{0,1\}^d$, but the velocity is still a d-dimensional vector $\boldsymbol{v}_i = (v_{i1}, v_{i2}, \ldots, v_{ij}, \ldots, v_{id})$ of the continuous d-dimensional space, $\boldsymbol{x}_{ij} \in \Re^d$. The velocity is also interpreted as the rate of change of each component of the position vector and is updated by same the formula:

$$\boldsymbol{v}_i = c_1\, \boldsymbol{v}_i + c_2\, \xi\, (\boldsymbol{b}_i - \boldsymbol{x}_i) + c_3\, \xi\, (\boldsymbol{g} - \boldsymbol{x}_i).$$

[1] http://www.particleswarm.info/Standard_PSO_2007.c
[2] http://www.particleswarm.info/Standard_PSO_2006.c

Nevertheless the position is obtained exclusively from the velocity by a new procedure. The original proposal of the authors of the PSO [19] uses the sigmoid function. Then every variable x_{ij} of the particles takes value 1 with probability $(1 + \exp(-v_{ij}))^{-1}$, and 0 otherwise. In order to avoid the explosion in the velocity, besides to use a small coefficient of inertia c_1, it is used a bound for its components (the typical values are around 6,0).

Other proposals to deal problems with discrete variables are the following. In [35] considers a version of the DPSO with a different form to update the velocity. In [1] is used a method based on the DPSO whose particles are influenced alternatively by the best position of their particular route and of its neighbourhood instead of simultaneously. In [27] is used the angular modulation just by four parameters in the continuous PSO. Proposed PSO versions for problems where the solutions are permutations, like in the well known Travelling Salesman Problem TSP, has been applied in [26], [28] [32] and [29]. In [29] a PSO has been applied to several integer programming problems. A discrete PSO with multiple values (MVPSO); i.e. a version for discrete variables with several values; not only binary, is proposed in [30]. The positions of the particles that are one-dimensional in the PSO and are bi-dimensional in the DPSO, are three-dimensional in the MVPSO. The positions of the particles are given by the values x_{ijk} that represent the probabilities that, in the position of the i-th particle, the j-th variable takes its k-th possible value. Therefore the evaluation of particles is at any moment stochastic and the velocity is also three-dimensional.

Correa and Freitas proposed in [8] a discrete PSO for the attribute selection in Data Mining that can be applied to the p-median problem, and to any other problem whose solution space is constituted by a selection of items of a finite universe. In this version, characteristic vectors are used to represent the set of variables. If d is the number of variables between which to make the selection, the position and the velocity of each particle are vectors with dimension d. The position vectors are binary vectors but the velocity vectors are constituted by positive real numbers. Each component of the velocity vector is interpreted as the relative probability of the corresponding binary component of the vector of position of the particle. The equation to update the velocity is:

$$\boldsymbol{v}_i = \boldsymbol{v}_i + c_1\,\boldsymbol{x}_i + c_2\,\boldsymbol{b}_i + c_3\,\boldsymbol{g}_i.$$

Note that, for each particle, the vectors $\boldsymbol{x}_i$, $\boldsymbol{b}_i$ and $\boldsymbol{g}_i$ correspond to particle positions and are binary but the velocity $\boldsymbol{v}_i$ is not binary.

In order to initialize the swarm, the number of items of each particle is selected at random. This number, possibly different for the particles, is fixed throughout the process. Then so many variables are selected at random as it corresponds for each particle. The initial velocity vector of all particles is constituted by ones.

In order to obtain the position of the particle from the corresponding velocity, one of its components is multiplied by a random number ξ uniformly distributed in [0,1]. Then, value 1 is assigned to the variables that reach the greatest products, leaving the rest of the variables with 0. The number of variables that reach value 1 is the corresponding one for each particle.

4. The Jumping Frogs Optimization.

In this work we propose a new discrete method that derives from the PSO but that it works without some important elements of the PSO and adds other new ones reason why it is considered a new metaheuristic of collective natural intelligence; the Jumping Frogs Optimization (JFO). In a discrete space, when lacking continuity, the movement, the velocity and inertia ideas loses sense, reason why the JFO works without these components, it keeps the attraction by the best positions. Instead of velocity and inertia we considered a random component in the movement of particles; that now it has the form of jumps. We use for the update of the position of the particle an expression similar to the one of Correa and Freitas for the update of the velocity. We interpret the weights of the equation as the probabilities that of a behaviour at random or guided by attraction of the best position of the own particle, or by attraction of the best position of its social neighbourhood or by attraction of the best global position. The attraction of these positions causes that the position of the particle evolves approaching some of these attractors whereas improvement; in a similar way a path re-linking improving.

We formally expressed the update equation by:

$$\boldsymbol{x}_i = c_1\,\boldsymbol{x}_i \oplus c_2\,\boldsymbol{b}_i \oplus c_3\,\boldsymbol{g}_i \oplus c_4\,\boldsymbol{g}^*.$$

The result of this operation consists of making random moves with probability c_1, approaching moves towards the best position of the own particle $\boldsymbol{b}_i$ with probability c_2, towards the best position of its social neighbourhood $\boldsymbol{g}_i$ with probability c_3, or towards the best global position $\boldsymbol{g}^*$ with probability c_4. The approach movements can be those of any local procedure search or path re-linking procedure. If the solutions are sets of points, the moves can be the interchange, elimination or insertion of elements, taking into account that the eliminated elements cannot be in the attractor and the inserted ones must came from the attractor. The attraction moves that do not produce improvement are rejected. This result is obtained in the following form.

In order to update the position with this operation, the unit interval [0,1] is divided into four segments with lengths c_1, c_2, c_3 and $c_4 = 1 - (c_1 + c_2 + c_3)$. Then a random number is generated with uniform distribution in [0.1] and based on the segment to which the resulting random number belongs, random improvement movements are applied to the position of the particle towards the corresponding attractor. The number of movements applied in each generation to the position of a particle is chosen at random according to a geometric probability distribution with mean equal to the product of p by the corresponding coefficient c_i. Therefore, after each movement a number generated at random with uniform distribution in [0.1] is multiplied by p and by corresponding coefficient c_i. If the result is greater than 1 a new move is applied, otherwise the movement stops until the next generation. The new move is generated at random and rejected if it does not improve the position.

In addition the JFO to incorporate a local search similar to the memetic algorithms and cultural algorithms. Namely, after each random or approaching to attractor movement, a local search is applied to every particle in the swarm. The local search consists in applying an improvement to the position while it exists. In order to do it, the improving move is chosen following an anxious strategy instead a greedy one. The set of possible moves is explored staring from a random one, and the first move found that improve the position is applied. However, the improving moves stop only when this exploration ends without improvement in the whole set of possible moves for the current position.

5. The *p*-Median Problem

The p-median problem one is one of the most important problems in location of services and constitutes one of the centres of interest in the logistic planning [5]. One of the questions that they confer to him great importance to the p-median problem one is to serve as model and prototype for other excellent problems on which the alternative solutions are based on choosing a fixed number of elements. In order to formalize the p-median problem one as a problem of location of services considers a set of n demand points where are the users of the service denoted by $Z = \{ z_1, z_2, \ldots, z_n \}$, and a set $L = \{ l_1, l_2, \ldots, l_m \}$ of possible locations of the points on watch. Each point of demand z_i usually has associate a weight w_i, that represents the amount that it demands or users in this point. The p-median problem consists of determining p simultaneously points of L (the medians) of which to establish the service, so that the total cost is diminished of transports necessary to satisfy all the demands of the users, supposing that this cost is proportional to the amount of demand and the whole range. Therefore, each user will be taken care of by the point on watch (median) closest than has a limitless capacity. In the classic discrete p-median problem **Z** and L are finite and formed by vertices or nodes of a network. In standard case $Z = L = V$. It is an NP-hard optimization problem (see [23] and [15]). If the p-median problem considers on a graph $G = (V,A)$, where the demand points are vertices V and set L is formed by all the points on the graph, including the vertices in V and all the intermediate points of the edges of A, is possible to restrict the search for p-median only to the vertices (see [16]). Therefore, the set L of possible locations can also be considered restricted to own set V of the vertices of the graph, becoming a discrete p-median problem one. In the exposition commonest all the vertices are simultaneously the points of demand and those of possible location; $Z = L = V$ (in addition all have equal weight). The discrete p-median problem is approached using the matrix of distances D between all the pairs formed by a possible location of L and a point of demand of Z. If the problem is created in a graph or network, these are distances between vertices that can be obtained by well-known algorithms of minimum ways, like those of Dijkstra or Floyd. If the problem is created in the plane, set L of possible locations and the set of points of demand Z are finite set of points and it works with the matrix of Euclidean distances. In the case commonest sets Z and L are equal and the matrix of distances D is a square matrix. Therefore the discrete p-median problem usually considered in strictly mathematical terms from the elements of the matrix of distances $D = [d_{ij} = d(z_i,z_j), i,j = 1, 2,\ldots, n]$, and the vector of weights $w = [w_i, i = 1, 2,\ldots, n]$. The objective is to find p rows of D that the average, weighed by the weights of w, of the minimums of each column in those rows is minimum. That is to say, is to find the set of indices $J \subseteq \{1, 2,\ldots, n\}$ with $|J| = p$, that minimizes:

$$F(X) = \sum_{i=1}^{n} w_i \min_{j \in J} d_{ij}$$

For the discrete p-median problem very different procedures from resolution have proposed; from the initial work of Cooper [7] has been applied most of heuristic the exact methods and. One recent revision of the applied heuristic procedures is in [24] and one annotated bibliography in [31]. Between the applied procedures are evolutionary procedures like the genetic algorithms [2], [25] and the Ant Colony Systems [21]. Method PSO has been applied recently to a problem of location similar to the p-median one but with costs of location in [33].

6. Computational Experiments

The DPSO proposed by its creators [19] for problems of optimization with binary variables, that was the first denominated DPSO (Discrete Particle Swarm Optimization). allows to deal with sets points by means of the characteristic vector (see [20]). Among discrete versions of the PSO proposed in Literature, the most appropriate for the discrete p-median problem is naturally the recent proposal of Correa and Freitas [8]. The only aspect that has to be modified is the number of elements of each particle that comes fixed to p, instead of being generated at random for each particle. The local search for the p-median included in the implementation is the proposal by Maranzana [22] that consists of the alternative application of allocation movements and location, until improvement is not obtained. The allocation consists of assigning each demand point to the nearest median of the solution represented by the position of the particle. The location consists of determining by inspection the median one of each one of these sets. Formally, the Z_j sets are defined by $Z_j = \{z \in Z: d(z,x_j) = \min_k d(z,x_k)\}$ and each point x_j of the position of the particle is replaced by the 1-median of Z_j; the point that minimizes max $\{d(x,z): z \in Z_j\}$.

For the computational experience, we selected problems of the OR-Library [4] that are used assiduously to prove heuristic for the p-median problem. The objective of the experience is to show the behaviour of the new proposals in comparison with the versions of the PSO of Literature. In order to compare the versions considered for p-median the discrete one 10 problems of the 40 available in the OR-Library [4] were selected and that are used frequently to prove heuristic procedures for this problem since they appeared in [3]. In table II are the results obtained with the adaptation of the proposal of Correa and Freitas (DPSO) and our proposal, without (JFO-LS) and with (JFO+LS) local search (LS). In JFO we used $c_1 = 1.5$ and $c_2 = c_3 = 0.33$.

TABLE I. RESULTS FOR THE *P*-MEDIAN

Ints	*n*	*p*	DPSO	JFO-LS	JFO+LS	Optima
1	100	5	5821	5819	5819	5819
2	100	10	4113	4093	4093	4093
3	100	10	4250	4250	4250	4250
4	100	20	3122	3034	3034	3034
5	100	33	1427	1357	1355	1355
6	200	5	7858	7824	7824	7824
7	200	10	5656	5639	5631	5631
8	200	20	4590	4445	4445	4445
9	200	40	2931	2773	2738	2734
10	200	67	1466	1276	1269	1255

The values of the parameters chosen for the tests with p-median were similar for the three procedures. The size of the swarm was 50 and the size of the random neighbourhood was 15. The coefficients of the equation of update of the velocity in DPSO were fixed to $c_1 = 0.1$, $c_2 = 0.2$ and $c_3 = 0.5$. The attraction coefficients in our JFO were fixed to $c_1 = 0.1$, $c_2 = 0.2$, $c_3 = 0.5$ y $c_4 = 0.1$.

7. Conclusions.

In this work the application of the PSO to the p-median problem has been analyzed. The most standard versions of the PSO for the p-median problem have been implemented making a slight adjustment of parameters. New version for the problems that consider the attraction between the solutions represented by particles following a faithful interpretation the problem. In addition the incorporation of a local search has improved the positions of particles after each update. The obtained experimental results show that an adjustment of the parameters of the versions is necessary standard to obtain of them an appropriate yield. Our proposal considerably improves the quality of the contributed solutions. Also another is that the introduction of the local search is able to improve still more the quality of the contributed solutions being reached the optimal solution in enough cases.

As future investigations are deepened in the experimental study of the aspects considered here. In addition the possibility will study of using other structures of neighbourhood and of paralleling the procedure. Also the application to other standard problems of the logistic planning and to mixed and more realistic problems will be considered.

Acknowledge

This work has been partially supported by projects TIN2005-08404-C04-03 (70% FEDER) and PI042005/044.

References

[1] B. Al-kazemi, C.K. Mohan (2000). “Multi-phase Discrete Particle Swarm Optimization”. Fourth International Workshop on Frontiers in Evolutionary Algorithms.

[2] O. Alp, E. Erkut, D. Drezner (2003). “An efficient genetic algorithm for the *p*-median problem”. Annals of Operations Research, 122:21-42.

[3] J.E. Beasley (1985). “A note on solving large *p*-median problems”. European Journal of Operational Research, 21:270-273.

[4] J.E. Beasley (1990). “OR-Library: Distributing test problems by electronic mail”. Journal of the Operational Research Society, 41:1069-1072.

[5] J. Bramel, D. Simchi-Levi (2005) "The Logic of Logistics: Theory, Algorithms, and Applications for Logistics Management" Springer.

[6] M. Clerc (2006). “Particle Swarm Optimization”. ISTE, International Scientific and Technical Encyclopaedia, 2006.

[7] L. Cooper (1964). “Heuristic methods for location-allocation problems”. SIAM Review. 6, 37–53.

[8] E.S. Correa, S.A. Freitas, C.G. Johnson (2006) “A new Discrete Particle Swarm Algorithm Applied to Attribute Selection in a Bioinformatic Data Set”. Proceedings of GECCO 2006, 35–42

[9] M. Dorigo, T. Stützle (2004). “Ant Colony Optimization”. MIT Press, 2004.

[10] R.C. Eberhard, J. Kennedy, (1995) “A new optimizer using particle swarm theory”, Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 39–43.

[11] R.C. Eberhart, Y. Shi, (2000). “Comparing inertia weights and constriction factors in particle swarm optimization”, Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, 16–19.

[12] R.C. Eberhart; Y. Shi, J. Kennedy (2006). "Computational Intelligence". Morgan Kaufmann, 2007.

[13] A.P. Engelbrecht (2006). "Fundamentals of Computational Swarm Intelligence". Springer 2006.

[14] A.I. El-Gallad, M.E. El-Hawary, A.A. Sallam, A. Kalas (2002). "Enhancing the Particle Swarm Optimizer via Proper Parameters Selection". Canadian Conference on Electrical and Computer Engineering, 792–797.

[15] M.R. Garey, D.S. Johnson (1979). "Computers and Intractability: A Guide to the Theory of NP-Completeness". Freeman, 1979.

[16] O. Kariv, S.L. Hakimi (1979). "An algorithmic approach to network location problem. Part 2: The *p*-median". SIAM Journal of Applied Mathematics, 37:539–560.

[17] J. Kennedy, "Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance", Proceedings of the 1999 Conference on Evolutionary Computation, 1931–1938.

[18] J. Kennedy, R. Eberhart (1995) "Particle swarm optimization". Proceedings of the 1995 IEEE International Conference on Neural Networks, IV, 1942–1948.

[19] J. Kennedy, R.C. Eberhart (1997) "A discrete binary version of the particle swarm algorithm". IEEE Conference on Systems, Man, and Cybernetics, Vol. 5, 4104–4108

[20] J. Kennedy, R. Eberhart (2001) "Swarm Intelligence", Morgan Kaufmann Publishers, San Francisco, 2001.

[21] T. Levanova, M.A. Loresh (2004). "Algorithms of Ant System and Simulated Annealing for the *p*-median Problem" Automation and Remote Control 65:431-438.

[22] F.E. Maranzana (1964) "On the Location of Supply Points to Minimize Transport Costs" Operational Research Quarterly 15:261-270.

[23] N. Megiddo, K.J. Supowit (1984). "On the Complexity of Some Common Geometric Location Problems". SIAM Journal of Computing 13(1):182-196.

[24] N. Mladenović, J. Brimberg, P. Hansen, J.A. Moreno-Pérez (2007) "The p-median problem: A survey of metaheuristic approaches". European Journal of Operational Research, en prensa.

[25] J.A. Moreno-Pérez, J.L. García-Roda, J.M. Moreno-Vega (1994) "A parallel genetic algorithm for the discrete *p*–median problem". Studies in Locational Analysis 7:131-141.

[26] G.C. Onwubolu, M. Clerc (2004). "Optimal operating path for automated drilling operations by a new heuristic approach using particle swarm optimisation", International Journal of Production Research, 42(3):473-491.

[27] G. Pampara, N. Franken, A.P. Engelbrecht (2005). "Combining Particle Swarm Optimisation with angle modulation to solve binary problems", IEEE Congress on Evolutionary Computing, Vol. 1, 89-96.

[28] W. Pang, K. Wang, C. Zhou, L. Dong (2004). "Fuzzy Discrete Particle Swarm Optimization for Solving Traveling Salesman Problem", Proceedings of the 4th

International Conference on Computer and Information Technology (CIT04), IEEE Computer Society.

[29] K.E. Parsopoulos, M.N. Vrahatis (2002). "Recent approaches to global optimization problems through particle swarm optimization", Natural Computing 1, No. 2-3, 235–306.

[30] J. Pugh, A. Martinoli (2006). "Discrete Multi-Valued Particle Swarm Optimization". IEEE Swarm Intelligence Symposium, 103–110.

[31] J. Reese (2006). "Solution methods for the *p*-median problem: An annotated bibliography". Networks 48(4):125-142.

[32] B.R. Secrest (2001). "Traveling Salesman Problem for Surveillance Mission using Particle Swarm Optimization", Thesis, School of Engineering and Management of the Air Force Institute of Technology, Air University , 2001.

[33] M. Sevkli, A.R. Guner (2006) "A New Approach to Solve Uncapacitated Facility Location Problems by Particle Swarm Optimization" Proceedings of 5th International Symposium on Intelligent Manufacturing Systems, 237-246.

[34] Y. Shi, R.C. Eberhart (1998) "Parameter selection in particle swarm optimization", Lecture Notes In Computer Science, 1447 (Proceedings of the 7th International Conference on Evolutionary Programming VII): 591–600.

[35] S. Yang, M. Wang, L. Jiao (2004). "A Quantum Particle Swarm Optimization", Proceedings of the Congress on Evolutionary Computing, 2004, Vol. 1, 320–324.

[36] H. Yapicioglu, G. Dozier, A.E. Smith, (2004) "Bi-criteria model for locating a semi-desirable facility on a plane using particle swarm optimization", Proceedings of the 2004 Congress on Evolutionary Computation, 2328–2334.

[37] L. Zhang, H. Yu, S. Hu (2005) "Optimal choice of parameters for particle swarm optimization" Journal of Zhejiang University Science, 6A(6):528-534