Available online at www.sciencedirect.com
ScienceDirect
Computers and Mathematics with Applications 56 (2008) 777–784
An International Journal
computers & mathematics
with applications
www.elsevier.com/locate/camwa

Design optimization of wastewater collection networks by PSO

Joaquín Izquierdo*, Idel Montalvo, Rafael Pérez, Vicente S. Fuertes

Centro Multidisciplinar de Modelación de Fluidos, Universidad Politécnica de Valencia, Camino de Vera s/n, 46022, Valencia, Spain

Abstract

Optimal design of wastewater collection networks is addressed in this paper by making use of the so-called PSO (Particle Swarm Optimization) technique. This already popular evolutionary technique is adapted for dealing both with continuous and discrete variables as required by this problem. An example of a wastewater collection network is used to show the algorithm performance and the obtained results are compared with those given by using dynamic programming to solve the same problem under the same conditions. PSO is shown to be a promising method to solve optimal design problems regarding, in particular, wastewater collection networks, according to the results herein obtained.

Keywords: Particle Swarm Optimization; Wastewater collection networks; Optimal design; Dynamic programming; Evolutionary method

1. Introduction

High costs associated with wastewater collection systems have shown to be one of the main limiting factors for the construction of these kinds of systems, especially in third world countries. The search for more economic solutions have led several researchers to try different optimization methods: linear programming [1], dynamic programming [2, 3], heuristic programming [4], classical optimization with the use of Lagrange multipliers [5], genetic algorithms [6], ant colony optimization [7], amongst others.

Wastewater system design is required to define the performance and impacts of the combined and separate sewer systems not only in terms of physical capacity and overflow performance, but also in terms of loadings generated into the receiving environment. Effective control of combined and separate sewer overflows depends on understanding the process behaviour and the ability to design in order to get suitable process dynamics.

To fully address the design of this kind of networks, transient analysis must be taken into account. That is to say, flowrates through the pipes must be considered as series of values depending on time, rather than peak values. Obviously, this makes it more difficult the design and virtually prohibits the use of techniques such as dynamic programming. In spite of the fact that dynamic programming has been used in different occasions and by different authors to address the design of wastewater collection systems, its use is rather limited since the need for discretization

* Corresponding author.
E-mail addresses: jizquier@gmmf.upv.es (J. Izquierdo), imontalvo@gmmf.upv.es (I. Montalvo), rperez@gmmf.upv.es (R. Pérez), vfuertes@gmmf.upv.es (V.S. Fuertes).

0898-1221/$ - see front matter
doi:10.1016/j.camwa.2008.02.007

and the division into states and stages, which is inherent to this technique, makes it nonviable to perform massive network analyses, even for networks of moderate size.

Addressing the optimal design of wastewater collection networks through evolutionary methods can take advantage of the use of any of the currently available mathematical models used for modelling flows within wastewater pipe networks, such as EPA-SWMM [8], which allows transient analysis. The search process would then be endowed with the capacity of performing the required network analyses as the solution is continually improved in search of a lower value for the fitness function used for the optimization problem.

In this paper we use an evolutionary algorithm, which has been relatively recently developed, known as Particle Swarm Optimization (PSO) [9,10]. This algorithm, initially designed to work with continuous variables, has been adapted in this paper to work also and simultaneously with discrete variables, as required by the problem under consideration.

Even though PSO has already become popular in different areas, such as travelling salesman problem, job scheduling, design of water supply systems, etc. (see, amongst others, [11–14]), in the knowledge of the authors there are no applications to the design of wastewater collection networks, being a mixed discrete–continuous nonlinear problem.

PSO has very deep intelligent background and it is suitable for science computation and general engineering applications. One of the important factors that make particle swarm optimization really attractive is that it is simple and there are very few parameters to adjust. It can achieve optimal or near-optimal solutions in a rather short time without enormous iterative computations in digital implementation. Even though a thorough mathematical foundation has not been developed yet, PSO has proven to be very effective for application.

The rest of this paper is as follows. First, PSO is concisely presented. Then, the proposed adaptations are described and applied to a specific wastewater collection network. The obtained results are compared with those given by dynamic programming. Finally, conclusions are presented and a number of future work actions suggested.

2. PSO description

Particle Swarm Optimization is an evolutionary computation technique that was first developed by Kennedy and Eberhart [9]. The particle swarm idea originated as a simulation of a simplified social system, the graceful but unpredictable choreography of a flock of birds. The word ‘swarm’ is used after a paper by Millonas [15], who developed several models in artificial life and considered certain principles in swarm intelligence. The selection of the term ‘particle’ comes from mechanics and is justified by the fact that positions and velocities are applied to the population elements, despite they being considered to have zero mass and volume. Kennedy and Eberhart’s first original idea was to simulate the social behaviour of a flock of birds in their endeavour to reach, when flying through the field (search space), their unknown destination (fitness function), e.g. the location of food resources. In PSO, each problem solution is a bird of the flock and is referred to as a particle. In this algorithm, birds evolve in terms of their individual and social behaviour and mutually coordinate their movement towards their destination [16].

To this end, each bird keeps track of its coordinates in the problem space and aims at a specific direction: the best solution (best local position) it has achieved so far. Birds also communicate among them and are able to identify the bird in the best position. In a coordinated way each bird evolves by changing its velocity so that it accelerates towards both its best position and the best position obtained so far by any bird in the flock (best global position). This enables each bird to explore in the search space from its new location. The process is repeated until the best bird reaches certain desired location. It is worth noting here that, according to the description, the process involves not only intelligent behaviour but also social interaction. This way, birds learn both from their own experience (local search) and from the group experience (global search).

PSO exhibits common evolutionary computation features including (i) initialization with a population of random solutions, (ii) search for optima by updating generations, and (iii) particles evolution through the problem space by following some specific strategies.

Thus, the process initially starts with a group of particles, which have been randomly generated, representing different solutions of the (hydraulic) problem. The ith particle is represented by its location in an s-dimensional space, where s corresponds to the number of variables of the problem. Any set of values of the s variables, determining the particle location, represents a candidate solution for the optimization problem.

During the process, as already explained, each particle i is associated with three vectors:

- its current location

$$X_i = (x_{i1}, x_{i2}, \ldots, x_{is})^t \tag{1}$$

- the better location it has reached so far,

$$P_i = (p_{i1}, p_{i2}, \ldots, p_{is})^t \tag{2}$$

- and its velocity, which enables it to evolve to a new location,

$$V_i = (v_{i1}, v_{i2}, \ldots, v_{is})^t . \tag{3}$$

In each cycle (iteration) the particle that best fits the objective function is obtained; its location will play an important role in the calculation of the movement evolution of every other bird. The updates of the particles are achieved as described by the following equations.

Individual particle's position in the solution space is updated by

$$newX_i = currentX_i + newV_i, \tag{4}$$

where the new velocity, $newV_i$, is given by

$$newV_i = \omega \cdot currentV_i + c_1 \cdot rand() \cdot (P_i - currentX_i) + c_2 \cdot rand() \cdot (P_g - currentX_i). \tag{5}$$

The new elements in Eq. (5) are as follows. P_g is the best position ever attained by any particle during the flight. c_1 and c_2 are the acceleration constants and represent the weighting of the stochastic acceleration terms that pull each particle simultaneously towards its best position and the best global position. These constants are also sometimes referred to as learning rates or factors. *rand*() is a function generating uniform pseudo-random numbers between 0 and 1. Note that both *rand*() numbers in (5) are independently generated. ω is an inertia term, proposed by Shi and Eberhart [16], that controls the impact of the velocities history into the new velocity, provides improved performance in a number of applications and can be suitably adapted during the calculation process. This operator allows a balance between local and global search and typically decreases with time, so that initially global search is favoured, but this trend is shifted towards local search as the solution process evolves, and results in less iteration on average to find an optimal solution.

The use of PSO is thus supported by two sociometric principles. Particles flow through the search space and are influenced by both the best particle within the population and the best individual solution ever found [17]. The second term in the RHS of Eq. (5) represents the cognition of individual thinking of a particle that is realised by comparing its current position, X_i, with the best position it has ever had, P_i. The last term of (5), in its turn, accounts for the social cooperation among the particles that is embodied by the comparison between X_i and the best position ever attained by any particle during the flight, P_g.

Particles' velocities on each dimension are confined to minimum and maximum velocities, which are user defined parameters.

$$V_{\min} \leq V_i \leq V_{\max}. \tag{6}$$

If the sum of accelerations causes the velocity on a specific dimension to fall out of the accepted range, then this velocity is sensibly limited to either $V_{\min}$ or $V_{\max}$. These are very important parameters. They determine the resolution with which regions between the present position and the target (best so far) positions are searched. If $V_{\max}$ is too high, particles might fly through good solutions. If $V_{\max}$ is too small, on the other hand, particles may not explore sufficiently beyond locally good regions. In fact, in this case, they could easily be trapped in local optima and are unable to move far enough to reach a better position in the problem space.

From the procedure described it can be inferred that the algorithm starts with a randomly generated population, evaluates its particles according to the fitness values, updates the population and searches for the optimum with random techniques. A simplified representation of the algorithm is now given by the next pseudo-code.

- Generate a random population of N particles (hydraulic solutions)
- Select particle 1 as the best particle
- Repeat the next block until fulfilment of termination condition
  - Determine the value of the inertia parameter ω

- Begin cycle from 1 to number of particles
  - Start
    - Calculate fitness function for particle i
    - If particle i has better fitness value than the fitness value of the best particle in history then set particle i as the new best particle in history
    - Calculate new velocity for particle i according to (5)
    - Update position of particle i according to (4)
  - End
- Show the solution given by the best particle.

The termination condition may be stated either in terms of a maximum number of iterations or in the event that certain value of the fitness function has been achieved [14]. In this work the algorithm will stop if after a number of a priori defined iterations the best solution has not changed.

PSO shares with other evolutionary techniques that it does not guarantee the global optimum. But, on the other hand, PSO does not need specific operators (such as crossover and mutation in the case of Genetic Algorithms, or pheromone updating in Ant Colony Optimization, amongst others), since particles update themselves with internal velocity. They also have memory and receive information only from the best particle in history, which is a simpler mechanism of information transmission than those used in Genetic Algorithms, for example. Particles try to converge to the best solution quickly, but PSO's main drawback is that it is difficult to keep good levels of population diversity and to balance local and global search.

On the other hand, PSO can only deal with continuous variables and cannot be directly applied to discrete problems, like the one under consideration. A number of proposals have been developed to adapt PSO to deal with discrete variables [18,12,6]. A modified method is described in this paper to deal with both continuous and discrete variables, as required by the design of wastewater collection networks.

3. Application of PSO to the design of wastewater systems

The design of wastewater collection networks involves the simultaneous use of continuous and discrete variables. For the sake of simplicity and comparison purposes, let us consider a network with no special elements, such as pumps, drops, tanks and other sewer system elements. The decision variables will then be pipes' diameters and slopes. While slopes are clearly continuous variables, diameters must be treated as discrete, since they have to adjust to the range of commercial pipes that are available for the design. Another continuous variable is the depth of excavation.

Next, we present a variant of PSO that is able to deal simultaneously with both continuous and discrete variables. Some considerations must be made in order to balance the treatment given to both groups of variables. This can be achieved by handling continuous variables as proposed in standard PSO and treating discrete variables in a way similar to the algorithm described in [11], which is an adaptation of PSO to handle only discrete variables. The approach we propose here for discrete variables involves plainly the use of the integer part of the different components of velocities. This way, the new velocity vector will be an integer vector and, as a consequence, the new updated positions will share this characteristic since the initial population, in its turn, must also have been generated by using only integer numbers. According to this simple idea, Eq. (5) will be replaced by

$$newV_i = fix(\omega \cdot actualV_i + c_1 \cdot rand() \cdot (P_i - currentX_i) + c_2 \cdot rand() \cdot (P_g - currentX_i)), \tag{7}$$

where $fix(\cdot)$ is a function that takes the integer part of its argument. On the other hand, it must be taken into account that the new velocity values should not exceed the bounds given by (6). In that event, the new velocities will be set to the maximum or minimum velocities, as appropriate.

The update of the position of a particle will be made according to (4), as stated by the standard PSO. In this particular case, if any of the components of the position vector takes a negative value, it will be set to zero. On the other hand, if it is greater than the maximum value for that component it will be saturated to this maximum. All the other aspects related to the PSO standard algorithm are dealt with in the standard manner.

There is, however, a singular aspect regarding velocity bounds in (6) that must be taken into consideration in order for the algorithm to treat both continuous and discrete variables in a balanced way. In this particular case, it has been found that using different velocity limits for discrete and continuous variables has produced better results.

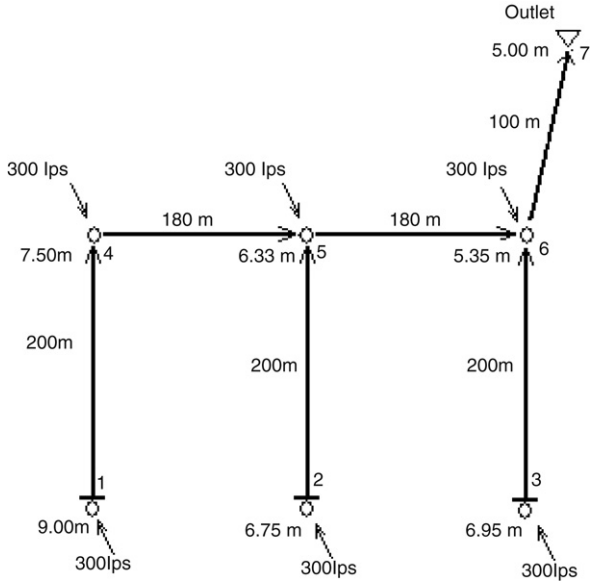

Fig. 1. Network under study.

In order to evaluate the algorithm, as proposed, working both with continuous and discrete variables, the design of a small water collection system has been performed. The network is shown in Fig. 1.

Peak flows, in litres per second, at the inlets of the network are associated to the different nodes, numbered 1 to 7. A more general treatment of this problem should consider sets of hydrographs at system inlet points rather than fixed pipe design flows. As we intend to use dynamic programming, for comparison purposes, we consider here only steady state behaviour. Nevertheless, different distributions of input flows, allowing more realistic designs compatible with different loads, may be considered by the PSO-based technique used here in a straightforward manner. In Fig. 1, also the length of the lines and the ground elevation of the nodes are given.

The design constraints consider the ratio of flow depth to the diameter in any pipe to be lower than 80% and the velocity not to exceed 4 m/s. Also, all the pipes must be buried at least one meter deep. Neither minimum velocities nor minimum ratio of water depth were considered in this simplified case. The fitness function accounts for the costs of the pipes and the excavation works. Finally, as in any hydraulic problem continuity and energy equations complete the set of constraints. Nonlinearity comes from the energy equations, all the other constraints and fitness function being linear.

Formally, the considered problem is formulated as the mixed continuous–discrete nonlinear problem

$$\text{Min } C = \sum_{i=1}^{\#pipes} diam.\_cost \cdot length\_pipe_i + \sum_{i=1}^{\#pipes} excav.\&related\_costs \tag{8}$$

$$\text{subject to } \begin{cases} wetted\_depth_i/diam_i < 0.8, \quad i = 1, \ldots, \#pipes \\ velocity_i < 4 \text{ m/s}, \quad i = 1, \ldots, \#pipes \\ pipe\_cover_i > 1 \text{ m}, \quad i = 1, \ldots, \#pipes \\ 0\% < slope_i < 5\%, \quad i = 1, \ldots, \#pipes \\ continuity\text{-}equations \\ energy\text{-}equations. \end{cases}$$

According to the pseudo-code given in the previous paragraph for the implementation process, particles randomly generated initially and/or generated during the evolutionary process can violate the system constraints resulting in infeasible particles. The evaluation of illegal particles in PSO and, in general, in every optimization problem with constraints, is crucial, especially for nonlinear programming problems, as the one herein considered. Therefore, the handling of system constraints, particularly the energy equations, which are nonlinear constraints, and the assessment of infeasible particles are of research interest. Currently, several methods have been developed to deal with system

Table 1
Commercially available pipe diameters

Diameter (mm)	Cost (euros/m)
150	5.78
200	8.48
300	18.38
400	34.28
500	56.18
600	84.08
700	117.98
800	157.88
1000	255.68
1200	377.48
1400	523.28

Table 2
Costs incurred by civil works

Weeding cost (€/m^2)	20.3
Transport (€/m^3)	3.19
Granular materials cost (€/m^3)	13.97
Selected filling cost (€/m^3)	6.47
Other filling cost (€/m^3)	4.55
Excavation cost (€/m^3)	14.21
Sponginess factor	1.3

constraints. These methods mainly consider preserving feasibility of solutions, penalty strategies and searching for feasible solutions, and they have several drawbacks. Amongst them, the penalty function methods are particularly promising, as evidenced by recent developments ([19], amongst others), and this is the approach we have used. Even though there are more sophisticated methods for constraint handling [7], we have decided to use a simple approach that works in the same way for all heuristics under investigation.

As a consequence, the total cost of the network is considered as the sum of the network cost (8) and a penalty cost, defined as

$$C = \sum_{i=1}^{\#pipes} diam.\_cost \cdot length\_pipe_i + \sum_{i=1}^{\#pipes} excav.\&related\_costs + \sum_{j=1}^{\#constr} penal_j \sum_{i=1}^{\#pipes} viol_{ij}^2,$$

where $viol_{ij}$ is the jth constraint violation at pipe i and $penal_j$ represents the penalty parameter corresponding to constraint j with a large value to ensure that infeasible solutions will have a cost greater than any feasible solution.

Commercially available pipe diameters for the problem under study are shown in Table 1. Table 2 shows civil work factors together with their incurred costs.

For the sake of simplicity, manhole costs have not been considered.

Flow behaviour through the pipes can be modelled by any of the codes to analyse sewer systems available in the market. In this particular case, use is made of the package EPA-SWMM [8]. Even though any of the analysis provided by this package may be used within the evolutionary algorithm, in the case of this paper only steady state formulation (for peak flows) will be performed for comparison purposes. In effect, given the low dimensionality of the network under study good results can be obtained by using dynamic programming. We use these results as a reference to compare with that given by PSO, since dynamic programming is theoretically capable of finding the global optimum solution with the only limitation of the discretization used for continuous variables, pipe slopes in our case.

The parameters used by this algorithm have been selected after preliminary tuning experiments following a number of suggestions [11,12,14,20]:

$$c1 = 3 \qquad c2 = 2$$
$$\omega = 0.5 + \frac{1}{2\,(\ln(k) + 1)},$$

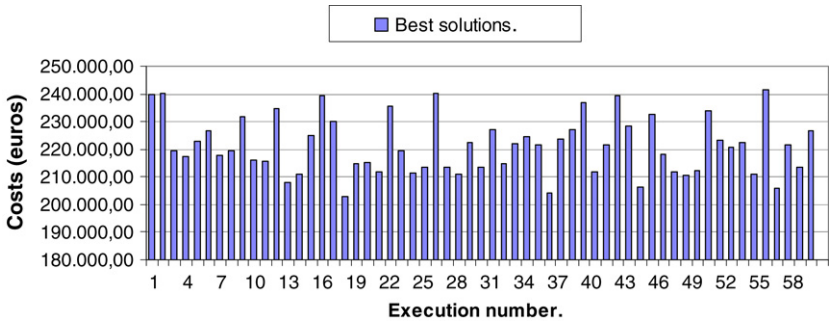

Fig. 2. Best solutions in any of the 59 runs.

Table 3
Diameters and slopes according to the best solution found with PSO and with dynamic programming

Pipe	Upstream node	Downstream node	PSO		Dynamic programming	
			Diameter (mm)	Slope (‰)	Diameter (mm)	Slope (‰)
1	1	4	500	6.8	500	7.5
2	2	5	800	10.87	600	3.1
3	3	6	400	15.2	400	16
4	4	5	600	7.7	600	7.6
5	5	6	600	3.0	700	12.1
6	6	7	1000	10.3	1000	5.49
Best cost using dynamic programming: 206.7×10^3 €						
Best cost using PSO: 203.055×10^3 €						

where

k is the iteration number

Maximum velocity for continuous variables = 20% of variable range

Maximum velocity for discrete variables = 50% of variable range

Minimum velocity = −Maximum velocity (in all the cases)

Number of particles (population size) = 100.

Sixty iterations were performed for the problem at hand. The termination condition stopped the process if after 800 iterations no improvement in the solution had been obtained. In only one out of the sixty cases the algorithm was not able to find a feasible solution. In all the other 98.3% of cases different feasible solutions were found. The costs of the best solutions for the different runs are shown in Fig. 2. The cost average for these best solutions is 221.29×10^3 € euros.

Table 3 shows the obtained solutions both with PSO and dynamic programming. Best costs using both methods are also shown. It is seen that the best solution obtained by PSO is better than the solution obtained by using dynamic programming.

4. Conclusions

The use of PSO in the studied example has shown very satisfactory results. Also, PSO has found a better solution than the one provided by dynamic programming, since a discretization of 0.2 m for the excavation depth has been used with this last technique. However, an additional experiment by using dynamic programming with a finer discretization of 0.1 m was also performed, the obtained result being 204.0×10^3 €. Of course, this value is lower than the one obtained with the coarser discretization but still it does not improve upon the best solution obtained with PSO. Thus, this algorithm shows itself able to go further beyond the limits of very fine dynamic programming discretizations, but hopefully without being trapped into the dimensionality curse.

Some of the simplifications and assumptions that have been made in the studied example cannot be considered, in general, in real life engineering problems. Comparison purposes with solutions found by using well developed and classical techniques, like dynamic programming, are the explanation for that oversimplification. Nevertheless, it

is worth to note here that considering more general problems (fitness functions, different loads, etc.) would be also possible in a straightforward manner, since they would not represent additional difficulties for an algorithm like PSO. In these more general cases, comparisons could be performed by using other evolutionary algorithms.

On the other hand, parametric studies should be developed in order to fine-tune the behaviour of PSO. These studies would provide answers about the influence of the population size, or the velocity limits, or the inertia factor, amongst other aspects of the algorithm. However, the results we have shown here are very promising and show that PSO is a reliable algorithm that must be considered as an excellent alternative to face the problem of optimum design of water collection networks, in particular, and other optimization problems, in general.

Acknowledgements

This work has been performed under the support of the projects Investigación Interdisciplinar no. 5706 (UPV) and DPI2004-04430 of the Dirección General de Investigación del Ministerio de Educación y Ciencia (Spain), including the support for I+D+i projects from the Consellería de Empresa, Universidad y Ciencia of the Generalitat Valenciana, and FEDER funds.

References

[1] A.A. Elimam, C. Charalambous, et al., Optimum design of large sewer networks, Journal of Environmental Engineering 115 (6) (1989) 1171–1190.
[2] J.B.M. Rodríguez, Optimización del diseño de redes y sistemas de alcantarillado, Revista Ingeniería Hidráulica 4 (1983) 119–129.
[3] A. Botrous, I. El-Hattab, M. Dahab, Design of wastewater collection networks using dynamic programming optimization technique, in: Procs. Of the ASCE Nat. Conf. on Environmental and pipeline Engineering, Kansas City, MO, United States, American Society of Civil Engineers, 2000, pp. 503–512.
[4] D.P. Desher, P.K. Davis, Designing sanitary sewers with microcomputers, Journal of Environmental Engineering 112 (6) (1986) 993–1007.
[5] P.K. Swamee, Design of Sewer Line, Journal of Environmental Engineering 127 (9) (2001) 776–781.
[6] L.Y. Liang, R.G. Thompson, et al., Optimising the design of sewer networks using genetic algorithms and tabu search, Engineering, Construction and Architectural Management 11 (2) (2004) 101–112.
[7] M.H. Afshar, Partially constrained ant colony optimization algorithm for the solution of constrained optimization problems: Application to storm water network design, Advances in Water Resources 30 (4) (2007) 954–965.
[8] L. Rossman, Storm Water Management Model user's manual (version 5.0), U. S. E. P. Agency, Cincinati (United States), 2005.
[9] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, Perth, Australia, IEEE Service Center, Piscataway, NJ, 1995, pp. 1942–1948.
[10] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proc. 6th Int Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39–43.
[11] Y.X. Jin, H.Z. Cheng, et al., New discrete method for particle swarm optimization and its application in transmission network expansion planning, Electric Power Systems Research 77 (3-4) (2007) 227–233.
[12] C.J. Liao, T. Chao-Tang, et al., A discrete version of particle swarm optimization for flowshop scheduling problems, Computers and Operations Research 34 (10) (2007) 3099–3111.
[13] I. Montalvo, J. Izquierdo, R. Pérez, M. Tung, Particle Swarm Optimization applied to the design of water supply systems, Computers and Mathematics with Applications 56 (3) (2008) 769–776.
[14] X.H. Shi, Y.C. Liang, et al., Particle swarm optimization-based algorithms for TSP and generalized TSP, Information Processing Letters, (2007) (in press). Corrected Proof.
[15] M.M. Millonas, Swarms, phase transition, and collective intelligence, in: C.G. Langton (Ed.), Artificial Life III, Addison Wesley, Massachusetts, 1994, pp. 417–445.
[16] Y.F. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: IEEE International Conference on Evolutionary Computation, 1998, pp. 69–73.
[17] M.S. Voss, Social programming using functional swarm optimization, in: IEEE Swarm Intelligence Symposium, Indiana, USA, 2003, pp. 103–109.
[18] B. Al-kazemi, C.K. Mohan, Multi-phase discrete particle swarm optimization, in: Fourth International Workshop on Frontiers in Evolutionary Algorithms, 2002.
[19] R.Y.K. Fung, J.F. Tang, D. Wang, Extension of a hybrid genetic algorithm for nonlinear programming problems with equality and inequality constraints, Computers and Operations Research 29 (2002) 261–274.
[20] Y.F. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization, in: Proc. of the 7th Annual Conf. on Evolutionary Programming, 1998, pp. 591–600.