Australian Journal of Basic and Applied Sciences, 4(8): 3366-3382, 2010
ISSN 1991-8178

Comparison of Some Evolutionary Algorithms for Approximate Solutions of Optimal Control Problems

Akbar H. Borzabadi, Mohammad Heidari

School of Mathematics and Computer Science, Damghan University, Damghan, Iran.

Abstract: This paper presents the investigations on the application of evolutionary algorithms particle swarm optimization (PSO) and invasive weed optimization (IWO) to find approximate solutions of optimal control problems. For this purpose, discrete form of the problem is converted to a quasi assignment problem, considering a partition of the time-control space. The approximate optimal control function is obtained as a piecewise constant function. Finally the numerical examples are given and by defining some tests, the results of applying evolutionary algorithms are compared.

Key words: Optimal control problem, Evolutionary algorithm, Discretization, Approximation.

INTRODUCTION

In recent years, considerable attention has been focused to use of numerical schemes to obtain approximate solutions of optimal control problem's (OCP). The numerical approaches for obtaining approximate solution of OCP's may be divided into different classes with their own advantages and characteristics. An overview of numerical methods for OCP's described by ordinary differential equations and integral equations can be seen in (Schmidt 2006).

Recently, evolutionary and heuristic algorithms as powerful tools have been raised in detecting optimal trajectories (Dsidri *et al.* 1990; Wuerl *et al.* 2003). Some heuristic and evolutionary algorithms and their extensions have been applied to extract approximate solutions in classical OCP's by converting the OCP to a Quasi Assignment Problem (QAP), (Fard *et al.* 2007; Borzabadi, *et al*. 2009). Our aim is modification of particle swarm optimization (PSO) and invasive weed optimization (IWO) for constructing approximate optimal solutions OCP's and comparison the results using these two methods with another evolutionary algorithm, i.e. genetic algorithm (GA) (Fard *et al.* 2007).

The Evolutionary Optimization Algorithms:

As already pointed out, the objective of this paper is to present a comparative evaluation of GA, PSO, and IWO for solving OCP's. In the next subsections, a summary review of the main characteristics and procedures for PSO and IWO algorithms is described.

Particle Swarm Optimization:

PSO is an evolutionary computation technique developed by Kennedy and Eberhart (Kennedy *et al.* 1995). PSO is a population-based optimization tool, which could be implemented and applied easily to solve various optimization problems. As an algorithm, the main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms. In PSO, the population is initialized randomly and the potential solutions, named particles, freely fly across the multidimensional search space (Jones 2006). During flight, each particle updates its own velocity and position by taking benefit from its best experience and the best experience of the entire population. The aim of a PSO algorithm is to minimize an objective function F which depends on a set of unknown variables, $(x^1, x^2, \ldots, x^n)$. In the PSO formalism, the optimization parameters are coded as a position $\vec{X}_i = (x_i^1, x_i^2, \ldots, x_i^n)^t$ of the particle i. The particle i is also characterized by its velocity $\vec{V}_i = (v_i^1, v_i^2, \ldots, v_i^n)^t$, its personal best position discovered so far $\vec{P}_i = (p_i^1, p_i^2, \ldots, p_i^n)^t$

Corresponding Author: Akbar H. Borzabadi, School of Mathematics and Computer Science, Damghan University, Damghan, Iran.

and the global best position of the entire population $\vec{G}_i = (g_i^1, g_i^2, \ldots, g_i^n)^t$. Let K be the iteration index in the optimization context. The new particle velocity and position are updated according to the move equations (Shi 2004),

$$\vec{V}_i^{k+1} = w^k \times \vec{V}_i^k + c_1 \times rand_1 \times (\vec{P}_i - \vec{X}_i^k) + c_2 \times rand_2 \times (\vec{G}_i - \vec{X}_i^k) \quad (1)$$

$$\vec{X}_i^{k+1} = \vec{X}_i^k + \vec{V}_i^{k+1} \quad (2)$$

where i=1,...,s, and s is the swarm size and $rand_1$,$rand_2$ are two random numbers, uniformly distributed in [0,1] and c_1,c_2 are the acceleration factors and w^k is inertia weight where evolves according to the equation.

$$w^k = w_{max} - \frac{w_{max} - w_{min}}{k_{max}} \times k \quad ,$$

Where w_{min} and w_{max} are minimum and maximum inertia weight, respectively, and K_{max} is maximum number of iterations. the swarm has a full connected topology in which all particles of the swarm are considered as neighbors and $\vec{G}_i = \vec{G}$ is the best position of the entire population. At each iteration, the behavior of a given particle is a compromise between three possible choices:

- to follow its own way
- to go toward its best previous position
- to go toward the best neighbor

Algorithm 1: Pseudo-codes for PSO algorithm

Initialize the position and velocity of each particle in the population randomly.
Calculate fitness value of each particle.

Calculate $\vec{P}$ and $\vec{G}$ for each particle.

Do
Update velocity of each particle using (1).
Update position of each particle using (2).
Calculate fitness value of each particle.

Update $\vec{P}$ for each particle if its current fitness value is better than its $\vec{P}$.

Update $\vec{G}$ for each particle, i.e., choose the position of the particle the best fitness value.

While termination criterion is not attained.

Invasive Weed Optimization:

IWO as a population-based algorithm was introduced by Mehrabian and Lucas in 2006 (Mehrabian *et al.* 2006). This algorithm is a bioinspired numerical optimization algorithm that simply simulates natural behavior of weeds in colonizing and finding suitable place for growth and reproduction. Some of the distinctive properties of IWO in comparison with other evolutionary algorithms are the way of reproduction, spatial dispersal, and competitive exclusion (Mehrabian *et al.* 2006). In IWO algorithm, the process begins with initializing a population. It means that a population of initial solutions is randomly generated over the problem space. Then members of the population produce seeds depending on their relative fitness in the population. In other words, the number of seeds for each member is beginning with the value of S_{min} for the worst member and increases linearly to *Smax* for the best member. For the third step, these seeds are randomly scattered over the search space by normally distributed random numbers with mean equal to zero and an adaptive standard deviation. The equation for determining the standard deviation (SD) for each generation is presented as:

$$\sigma_{iter} = \frac{(iter_{max} - iter)^n}{(iter_{max})^n}(\sigma_{initial} - \sigma_{final}) + \sigma_{final},$$

where $iter_{max}$ is the maximum number of iterations, σ_{iter} is the SD at the current iteration and n is the nonlinear modulation index. The produced seeds, accompanied by their parents are considered as the potential solutions for the next generation. Finally, a competitive exclusion is conducted in the algorithm, i.e. after a number of iterations the population reaches its maximum, and an elimination mechanism should be employed. To this end, the seeds and their parents are ranked together and those with better fitness survive and become reproductive.

Algorithm 2: Pseudo-codes for IWO algorithm

1. Generate random population of N_0 solutions.
2. For *iter*=1 to the maximum number of generations.
a. Calculate fitness for each individual.
b. Compute maximum and minimum fitness in the colony.
c. For each individual w in W.
i. Compute number of seeds of w, corresponding to its fitness.
ii. Randomly distribute generated seeds over the search space with normal distribution around the parent plant (w).

iii. Add the generated seeds to the solution set, W.

d. If $(|W| = N) > P_{max}$.

i. Sort the population W in descending order of their fitness.

ii. Truncate population of weeds with smaller fitness until $N=P_{max}$.
3. Next *iter*.

Converting OCP to QAP:

Consider a classical OCP as follows:

$$minimize \quad I(\mathbf{x}(t), u(t)) = \int_0^{t_f} f_0(t, \mathbf{x}(t), u(t)), \tag{3}$$

$$subject\ to \qquad \dot{\mathbf{x}} = \mathbf{g}(t, \mathbf{x}(t), u(t)), \tag{4}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \ , \ \mathbf{x}(t_f) = \mathbf{x}_f \tag{5}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}$, $\mathbf{g} : \mathbb{R}^{n+2} \mapsto \mathbb{R}^n$, t_f is known and $|u| \leq K$. Our main aim is trajectory planning, i.e. to find a bounded control function $u(\cdot)$ such that the corresponding state $\mathbf{x}(\cdot)$ to this control function satisfies (4)-(5) and minimizes (3). In order to this work, first, the interval $[0,t_f]$ is divided by N equidistance subinterval $[t_{i-1}, t_i], i = 1, 2, \ldots, N$. Since our aim is to detect an approximate optimal control for problem (3)-(5), we focus on finding an optimal piecewise control function $u(\cdot)$ for the problem. Corresponding each interval $[t_{i-1}, t_i], i = 1, 2, \ldots, N$, we partition the interval $[-k,k]$ to m equal subinterval $[u_{j-1}, u_j], j = 1, 2, \ldots, m$, where $u_0 = -K$ and $u_m = K$ and so the space of time and control is discretized to piecewise constant segments. Of course we need to find the best m+1 constants $u_0, u_1, \ldots, u_m$. A typical discretization is given in Figure 1 with n=7 and m=6.

One of approaches for finding the best selection among of all constants in each interval of $[t_{i-1},t_i], i=1,2,\ldots,N$, and finally to find the best approximate control is enumerate of all cases. But we will deal with an extreme hard computational complexity. Because we must check N^m piecewise constant functions. Our aim is to use of PSO and IWO algorithms approach for detecting the best approximate piecewise constant control function. If $u(t)=\sum_{k=1}^{N} u_k \chi_{[t_{k-1},t_k)}(t)$, where $\chi_{[t_{k-1},t_k)}(t)$ is charecteristic function on interval $[t_{k-1},t_k)$, be a piecewise constant function, then by a numerical method as Euler method or Runge-Kutta, we can find trajectory corresponding to $u(t)$ from (4) with initial condition $\mathbf{x}(0)=\mathbf{x}_0$ Thus, if (ξ,υ) be a pair of the trajectory and the control which satisfies (4) with initial condition $\mathbf{x}(0)=\mathbf{x}_0$ and for given a small number $\varepsilon>0$, $\|\xi(t_f)-\mathbf{x}_f\|\le\varepsilon$, then we can claim that, a good approximate pair for minimizing functional I in (3) have been found, here $\|.\|$ is Euclidean norm.

To convert the OCP to QAP, we use a similar framework of solving OCP by Ant Colony Optimization (Borzabadi *et al*. 2009). In fact, we decide to assign constant $\upsilon_k \in \{\upsilon_0,\ldots,\upsilon_m\}$ for every interval $[t_{i-1},t_i], i=1,2,\ldots,N$. The process will be continued until the objective function gets its optimal value. In other word, after discretization of the OCP, the problem is converted to a QAP with an extra objective function, i.e. we add the term $\|\xi(t_f)-\mathbf{x}_f\|$ to the original objective function and then, we apply PSO and IWO algorithms for this new criteria function. Therefore, by applying the method above, the problem (3)-(4) with conditions (5) is converted to:

$$minimize \quad I(\xi(t),\upsilon(t))=\sum_{i=1}^{N} f_0(t_i,\xi(t_i),\upsilon(t_i))+M\|\xi(t_f)-\mathbf{x}_f\|, \tag{6}$$

$$subject\ to \qquad \xi(t_{i+1})=\xi(t_i)+hg(t_i,\xi(t_i),\upsilon(t_i)), \quad i=0,1,...,N-1, \tag{7}$$

$$\xi(0)=\mathbf{x}_0, \tag{8}$$

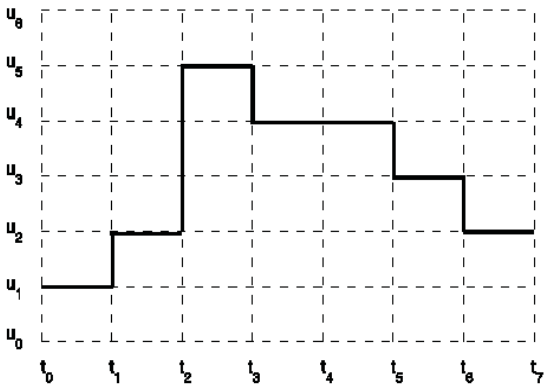

Fig. 1: A typical control function in time-control space discretization.

where $\sum_{i=0}^{N} w_i f_0(t_i,\xi(t_i),\upsilon(t_i))$ is an approximate of integral in (3) which can be obtained from a numerical integration method, as Newton-Cotes methods, w_i, $i=0,1,\cdots,N$, are the weights of numerical

method of integration and is a large positive number (like as big-M method). To obtain a numerical solution for the problem (6)-(7) with boundary conditions (8) via PSO and IWO algorithms, we focus just on the space time and control. Now, we are ready to present the numerical algorithm for achievement to the approximate optimal control.

Modified Algorithms for PSO and IWO:

In this section, modification of PSO and IWO algorithms will be presented, to obtain approximate solution for OCP (6)-(7) with boundary conditions (8).

Algorithm of PSO for OCP:

This algorithm is designed in two stages, initialization step and main steps. In the initialization step the process of construction the fitness function for an n -tuple as $(\upsilon_1,\cdots,\upsilon_n)$ from the control-time space, where $\upsilon_i, i=1,\cdots,n$ are equidistance nodes on the set of control values, is specified. The main steps contain the main structure of algorithm considering initialization step.

Initialization step:

Choose $c_1, c_2, w_{max}, w_{min}$ and k_{max} .

Choose a population of random particles, *i.e.,* random n-tuples as $(\upsilon_1,\cdots,\upsilon_n)$ from the control-time space and initialize the velocity of each particle.

Calculate the trajectory corresponding each particle using (7) and with initial condition (8).

Calculate fitness value of each particle using (6).

Calculate $\vec{P}$ *and* $\vec{G}$ for each particle.

Main steps:

Step 1. Update velocity of each particle using (1).

Step 2. Update position of each particle using (2).

Step 3. Calculate the trajectory corresponding each particle using (7) and with initial condition (8).

Step 4. Calculate fitness value of each particle using (6).

Step 5. Update $\vec{P}$ for each particle if its current fitness value is better than its $\vec{P}$.

Step 6. Update $\vec{G}$ for each particle, *i.e.,* choose the position of the particle the best fitness value.

Step 7. If the termination conditions are satisfied stop, otherwise go to Step 1.

Algorithm of IWO for OCP:

As previous algorithm, this algorithm is designed in two stages, initialization step and main steps. In addition, the process of choose an n-tuple as $(\upsilon_1,\cdots,\upsilon_n)$ from the control-time space, the selection of quantity necessary for applying IWO algorithm method are listed in initialization step. The main steps contain the main structure of algorithm considering initialization step.

Initialization step:

Choose $S_{min}, S_{max}, P_{max}, \sigma_{initial}, \sigma_{final}$ and $iter_{max}$.

Choose a population of random individuals, *i.e.,* random n-tuples as $(\upsilon_1,\cdots,\upsilon_n)$ from the control-time space.

Main steps:

Step 1. Calculate the trajectory corresponding each individual using Eq. (7) and with initial condition (8).

Step 2. Calculate fitness for each individual using Eq. (6).
Step 3. Compute maximum and minimum fitness in the colony.
Step 4. Compute number of seeds of each individual, corresponding to its fitness.
Step 5. Randomly distribute generated seeds over the search space with normal distribution around the parent plant.
Step 6. Add the generated seeds to the solution set, *W*.

Step 7. If $(|W|) > P_{max}$.

(*i*) Sort the population *W* in descending order of their fitness.

(*ii*) Truncate population of weeds with smaller fitness until $|W| = P_{max}$.

Step 8. If the termination conditions are satisfied stop, otherwise go to Step 1.

Some Tests and Numerical Results:

In this section, first we introduce an error estimation as $e(t_f) = \left\| \xi(t_f) - \mathbf{x}_f \right\|$ on $[0, t_f]$, where $\mathbf{x}_f$ and $\xi(t_f)$ are the final states of the exact and approximate solutions, respectively. We compare the efficiency of proposed approaches via three tests as follows:

(*i*) Test #1

In this test, the effect of the number of nodes have been compared. The number of iterations and population size are fixed. The testing conditions are: number of iterations = 100, population size = 20 and the number of nodes are gradually increased from 5 to 50.

(*ii*) Test #2

In this test, the error estimation of different approach's are compared when the number of nodes and iterations are fixed and the population size changes. The testing conditions are: number of nodes = 20, number of iterations = 100, and population size are gradually increased from 5 to 50.

(*iii*) Test #3

In this test, the effect of number of iterations have been compared when the number of nodes and population size are fixed. The testing conditions are: number of nodes = 20, population size = 20 and the number of iterations are gradually increased from 50 to 275.

Meanwhile, Toolbox of Matlab (Ver, 7.8) have been used for extracting the approximate solutions on basis of the given approach in (Fard *et al.* 2007).

Example 1:

In the first example, we consider an OCP of minimizing

$$I(u(.)) = \int_0^1 u^2(t)dt,$$

subject to

$$\dot{x}(t) = \frac{1}{2}x^2(t)\sin(x(t)) + u(t),$$

with initial and final conditions

$$x(0) = 0, x(1) = 0.5,$$

where *u* is bounded control function, $0 \le \mu \le 1$.

The comparison of the error estimations for the given tests which are obtained by applying different algorithms can be seen in Figures 2, 3 and 4. The approximate optimal trajectory and control functions have been shown in Figure 5 and Figure 6, respectively, in case the number of nodes 20, population size 20 and the number of iterations 275.

Table 1: Error estimation for Example 1 by using test # 1.

Number of nodes	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	1.6381e-012	9.5256e-008	1.9587e-007
10	5.4622e-014	6.0200e-009	8.3332e-008
15	3.2902e-012	3.4559e-008	4.0174e-008
20	4.9960e-015	9.9471e-009	3.7466e-008
25	4.3254e-013	8.6163e-009	3.7297e-008
30	5.7109e-013	2.5178e-010	4.8791e-008
35	2.6090e-013	1.3094e-009	2.3968e-007
40	8.8817e-016	3.1297e-009	1.4621e-007
45	2.3314e-015	3.5920e-009	2.4568e-008
50	3.4583e-014	4.1608e-009	1.8205e-009

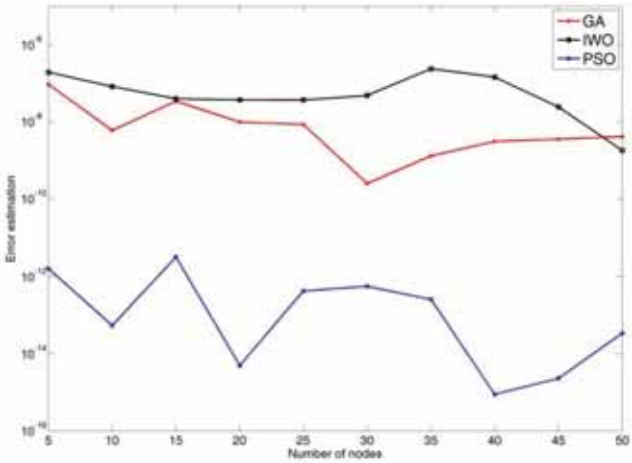

Fig. 2: Error estimation for Example 1 by using test # 1.

Table 2: Error estimation for Example 1 by using test # 2.

Population size	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	1.0877e-012	4.5204e-007	3.9243e-007
10	3.5527e-015	8.3062e-008	7.5282e-008
15	3.3306e-016	3.7564e-008	4.9875e-008
20	9.4591e-014	1.3225e-009	3.7466e-008
25	2.3592e-014	9.5153e-009	8.5742e-010
30	1.1156e-012	5.7734e-009	2.3709e-008
35	3.9968e-015	8.0221e-009	3.8634e-008
40	7.2442e-014	4.4192e-009	2.7428e-008
45	9.6833e-013	9.1611e-010	3.2962e-008
50	8.6597e-015	5.6643e-009	5.1353e-008

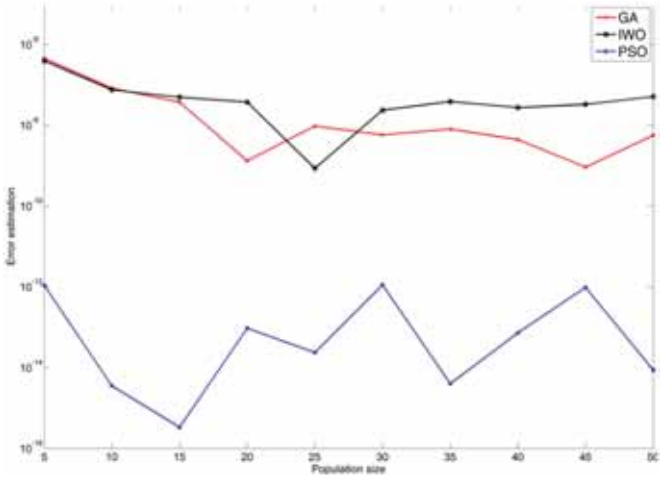

Fig. 3: Error estimation for Example 1 by using test # 2.

Table 3: Error estimation for Example 1 by using test # 3.

Number of iterations	PSO Algorithm	Genetic Algorithm	IWO Algorithm
50	8.3932e-009	6.3092e-008	7.3608e-007
75	1.4216e-011	4.0870e-008	4.0750e-007
100	2.4757e-014	1.3225e-009	3.7466e-008
125	2.2204e-016	8.6241e-009	5.0255e-008
150	2.2204e-016	5.1859e-011	1.2798e-008
175	0	5.2728e-009	3.0745e-008
200	0	1.0137e-009	2.8583e-008
225	0	2.6643e-009	8.8433e-009
250	0	8.0687e-009	8.3972e-009
275	0	2.5036e-010	3.4034e-009

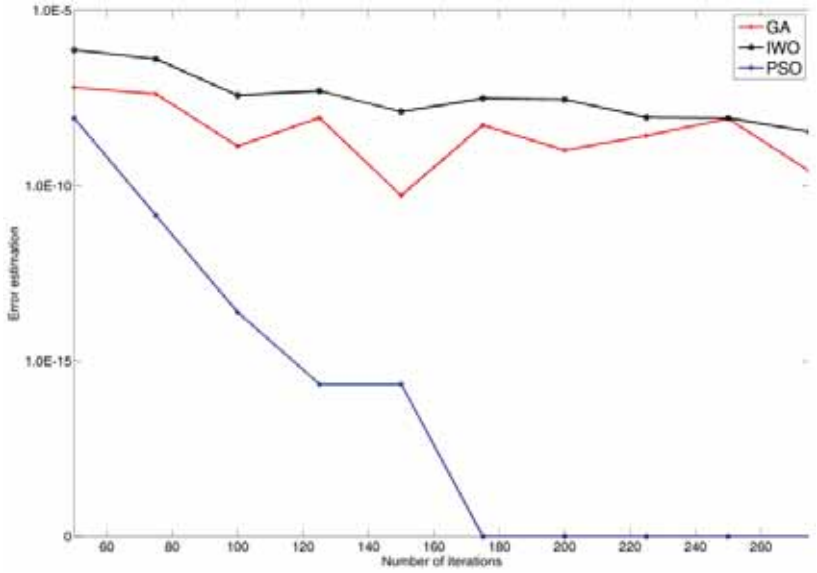

Fig. 4: Error estimation for Example 1 by using test # 3.

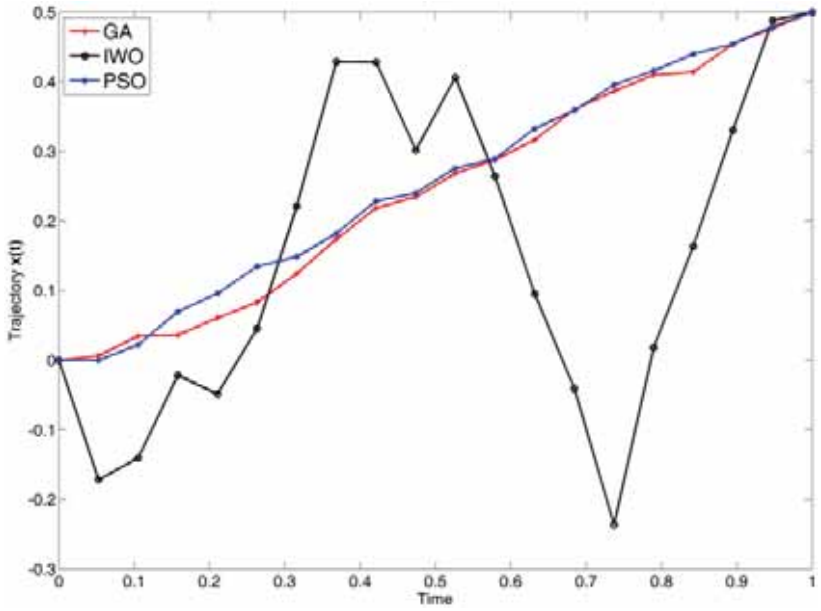

Fig. 5: Trajectory functions for Example 1.

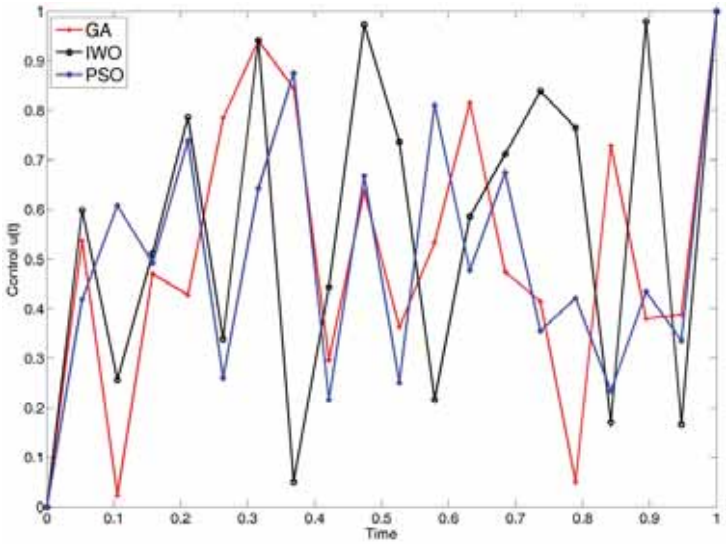

Fig. 6: Control functions for Example 1.

Example 2:

In the second example, we consider an OCP of minimizing

$$I(x(.),u(.)) = \int_0^1 (x(t)-e^t)^2 + (u(t)-t)^2\,dt,$$

subject to

$$\dot{x}(t) = (2(tx(t)-u(t)e^t)+1)e^t,$$

with initial and final conditions

$$x(0)=1,\ x(1)=e,$$

where u is bounded control function and $|u|<1$.

One may observe the comparison of the error estimations for the given tests which are obtained by applying different algorithms in Figures 7,8 and 9. The graph of the optimal approximate trajectories are shown in Figure 10 and also in Figure 11, one may find the graph of approximate optimal controls, in case the number of nodes 20, population size 20 and the number of iterations 275.

Table 4: Error estimation for Example 2 by using test # 1.

Number of nodes	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	2.3620e-011	1.1270e-006	3.7123e-006
10	9.3259e-015	6.1803e-007	4.5446e-006
15	1.1946e-013	5.2553e-007	2.4214e-007
20	2.2204e-014	1.2150e-008	7.3571e-007
25	3.0651e-012	5.7718e-007	5.4230e-007
30	9.8144e-014	2.7795e-007	7.9801e-007
35	3.9568e-013	3.1360e-007	3.4318e-006
40	4.1567e-013	6.2978e-007	2.9440e-006
45	1.8208e-014	3.3611e-007	4.7664e-007
50	2.3652e-012	8.5678e-008	2.6432e-006

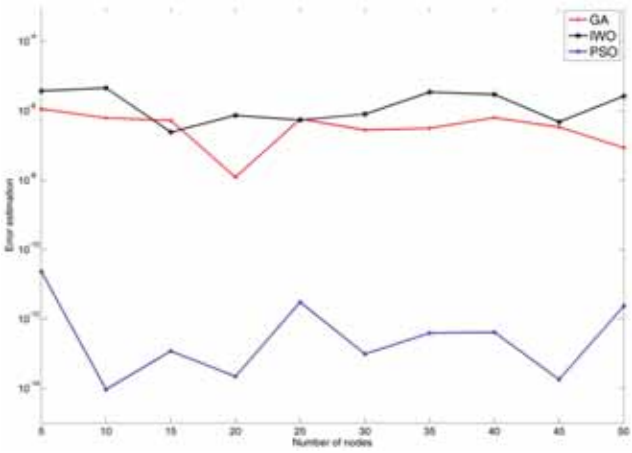

Fig. 7: Error estimation for Example 2 by using test # 1.

Table 5: Error estimation for Example 2 by using test # 2.

Population size	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	2.3452e-012	1.8064e-006	1.5850e-006
10	1.7764e-015	2.7384e-007	1.6574e-006
15	7.7893e-013	1.0238e-006	1.1924e-006
20	2.2204e-014	1.2150e-008	7.3571e-007
25	9.0328e-013	5.7578e-007	3.8036e-007
30	8.7930e-014	1.0076e-007	1.7436e-006
35	2.9843e-013	3.6071e-008	5.2943e-007
40	7.1054e-015	4.2154e-009	4.1747e-006
45	5.7732e-015	2.2104e-009	1.5980e-006
50	1.0214e-013	8.8348e-008	3.7908e-007

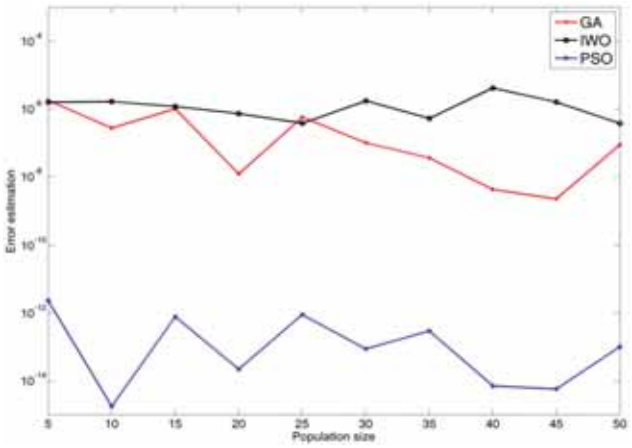

Fig. 8: Error estimation for Example 2 by using test # 2.

Table 6: Error estimation for Example 2 by using test # 3.

Number of iterations	PSO Algorithm	Genetic Algorithm	IWO Algorithm
50	7.0087e-009	2.4022e-006	1.5896e-006
75	6.1765e-011	2.7508e-007	4.2760e-006
100	2.2204e-014	1.2150e-008	7.3571e-007
125	3.0198e-014	2.7668e-007	4.9483e-006
150	1.9096e-014	1.6247e-007	2.8263e-006
175	0	2.2034e-007	5.9669e-008
200	0	4.8382e-009	6.8705e-007
225	0	6.5443e-009	4.0377e-007
250	0	1.6292e-009	4.1014e-007
275	0	1.5095e-009	1.7333e-007

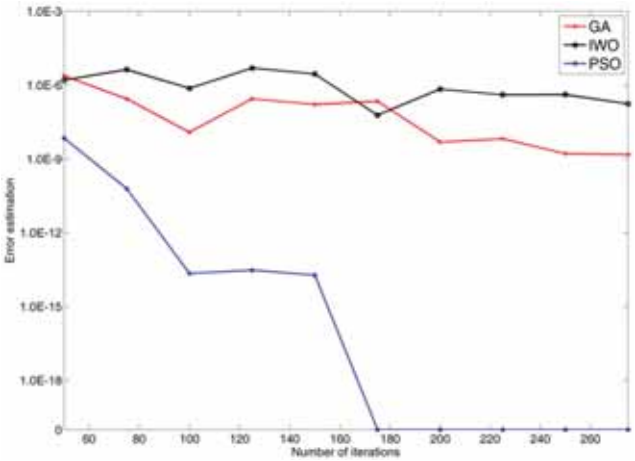

Fig. 9: Error estimation for Example 2 by using test # 3.

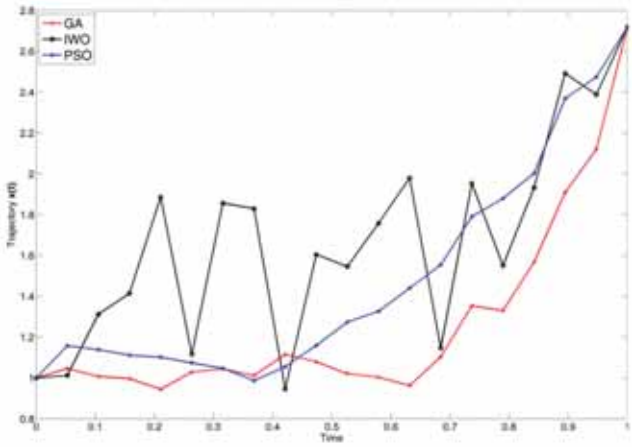

Fig. 10: Trajectory functions for Example 2.

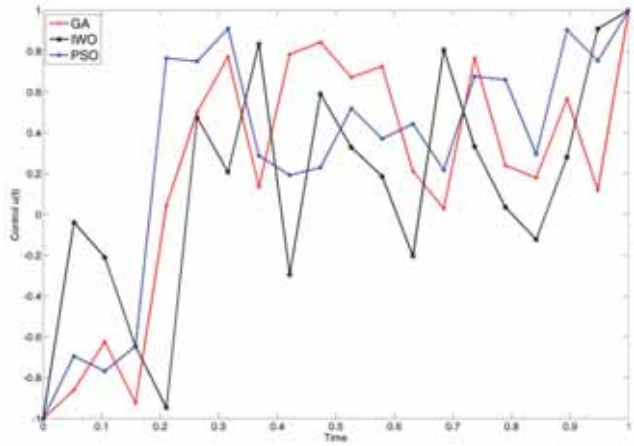

Fig. 11: Control functions for Example 2.

Example 3.:

In the third example we consider an OCP of minimizing

$$I(u(.)) = \frac{1}{2}\int_0^1 u^2(t)dt,$$

subject to

$$\dot{x}_1(t) = x_2,$$

$$\dot{x}_2(t) = -x_2(t) + u(t),$$

with boundary conditions

$$x_1(0) = 1,\ x_2(0) = 1,$$

$$x_1(1) = 0,\ x_2(1) = 0,$$

where the control function u is bounded as $|u| \le 18$.

Almost, one can claim that achievement to the approximate solutions of this OCP by other numerical schemes, is very complicated. As prvious examples, the comparison of the error estimations for the given tests which are achieved by applying different algorithms can be observed in Figures 12-17. The exact and approximate optimal trajectories $x_1(.)$ and $x_2(.)$ and approximate optimal control are shown in Figures 18-20, respectively, in case the number of nodes 20, population size 20 and the number of iterations 275.

Table 7: Error estimation x_1 for Example 3 by using test # 1.

Number of nodes	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	1.1377e-010	1.1326	3.9136e-006
10	1.6632e-012	1.1293	3.4325e-007
15	6.3591e-013	1.4076	3.8105e-007
20	5.3388e-014	1.4897	6.7850e-007
25	5.6765e-010	1.4740	1.8650e-006
30	6.5968e-011	1.4924	9.3929e-007
35	1.5069e-011	1.4880	7.2935e-008
40	2.6154e-009	1.4596	1.7768e-007
45	1.1966e-012	1.4956	2.8186e-006
50	6.6811e-007	1.5071	5.7506e-007

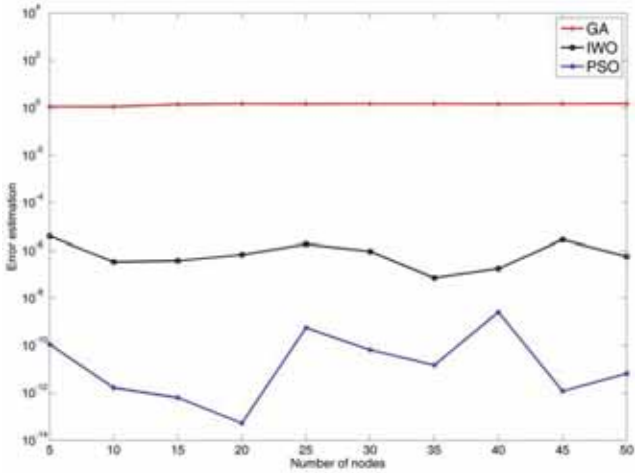

Fig. 12: Error estimation x_1 for Example 3 by using test # 1 .

Table 8: Error estimation x_2 for Example 3 by using test # 1.

Number of nodes	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	5.6676e-011	3.6076e-005	1.9721e-006
10	5.0471e-013	9.8679e-005	2.3314e-006
15	5.3768e-013	1.0565e-007	3.7624e-007
20	4.4365e-013	2.9034e-007	2.7904e-007
25	6.5277e-011	8.7620e-008	2.6160e-008
30	1.6996e-010	9.6752e-006	6.2609e-008
35	2.8577e-013	0.0325	5.5424e-007
40	1.1751e-010	0.0040	5.5424e-007
45	1.4683e-011	0.1963	3.1384e-008
50	1.3089e-005	0.1793	3.7717e-008

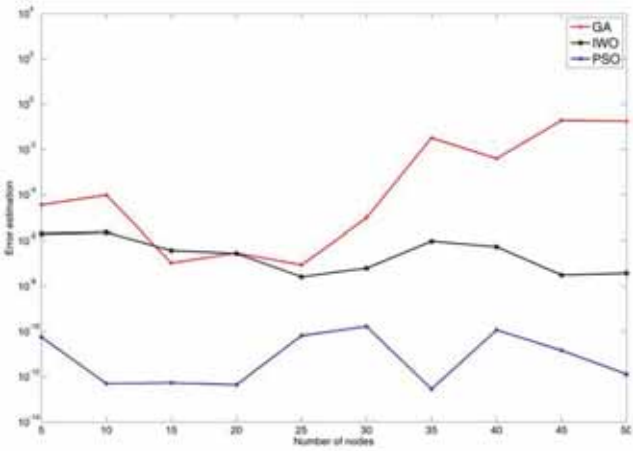

Fig. 13: Error estimation $x_2$for Example 3 by using test # 1.

Table 9: Error estimation x_1 for Example 3 by using test # 2.

Population size	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	1.6127e-007	1.5112	2.0865e-007
10	1.8493e-009	1.5951	5.8744e-006
15	4.4269e-013	1.5477	9.4962e-008
20	5.3388e-014	1.4897	6.7850e-007
25	4.4008e-008	1.4974	8.1491e-007
30	5.0410e-012	1.5089	3.8449e-007
35	1.5991e-013	1.4212	6.2129e-007
40	8.6863e-013	1.3082	1.1204e-007
45	5.1683e-013	1.1164	5.4255e-007
50	1.2567e-013	1.0752	1.9204e-007

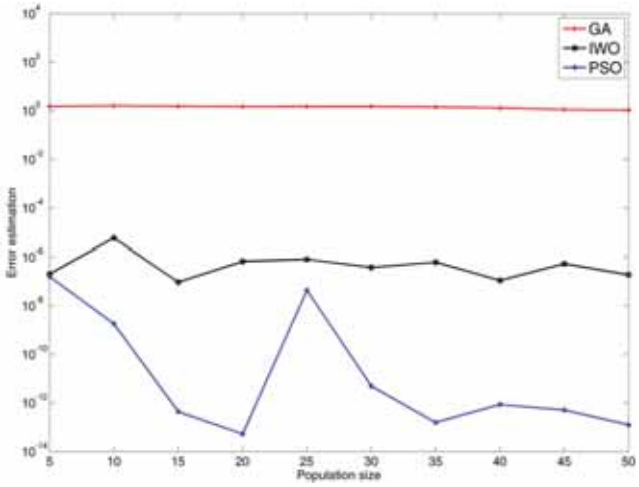

Fig. 14: Error estimation x_1 for Example 3 by using test # 2.

Table 10: Error estimation x_2 for Example 3 by using test # 2.

Population size	PSO Algorithm	Genetic Algorithm	IWO Algorithm
5	7.4368e-010	0.1638	3.7375e-006
10	9.1376e-010	0.1014	4.6774e-007
15	5.0930e-012	0.1606	2.0588e-006
20	4.4365e-013	2.9034e-007	2.7904e-007
25	2.7851e-009	8.5487e-007	1.9865e-007
30	2.5327e-011	4.5149e-006	6.5141e-007
35	3.9496e-013	5.6749e-005	8.7108e-007
40	1.8286e-012	4.9436e-005	2.0274e-007
45	4.1056e-013	6.2449e-004	5.0884e-007
50	6.6058e-015	9.4009e-004	7.8810e-007

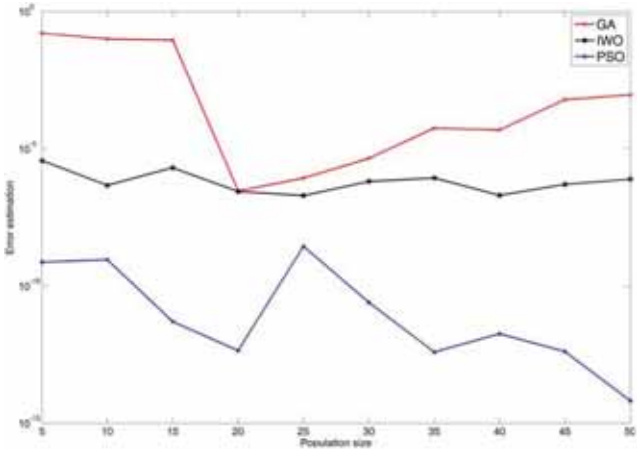

Fig. 15: Error estimation x_2 for Example 3 by using test # 2.

Table 11: Error estimation x_2 for Example 3 by using test # 3.

Number of iterations	PSO Algorithm	Genetic Algorithm	IWO Algorithm
50	6.4691e-007	1.5547	1.1291e-006
75	2.8760e-010	1.3579	3.1599e-006
100	5.3388e-014	1.4897	6.7850e-007
125	4.7653e-015	1.4525	2.5197e-007
150	4.8572e-017	1.4256	2.3863e-007
175	5.5511e-017	1.3808	7.3007e-007
200	3.4694e-017	1.5035	6.0131e-007
225	2.0817e-017	1.3899	3.5396e-007
250	6.9389e-018	1.4062	2.7320e-007
275	3.4694e-018	1.3766	1.2072e-008

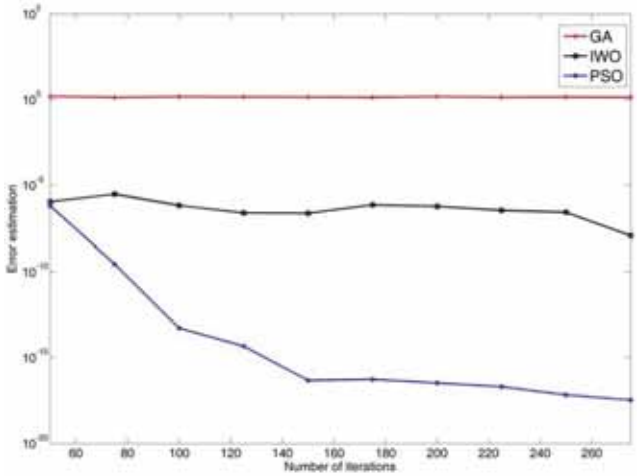

Fig. 16: Error estimation x_1 for Example 3 by using test # 3 .

Table 12: Error estimation x_2 for Example 3 by using test # 3.

Number of iterations	PSO Algorithm	Genetic Algorithm	IWO Algorithm
50	1.9478e-008	1.0165e-004	3.4126e-006
75	3.6311e-012	2.9403e-005	4.2610e-007
100	4.4365e-013	2.9034e-007	2.7904e-007
125	5.0349e-014	7.9919e-007	8.2350e-007
150	3.3307e-016	1.3255e-006	3.3721e-007
175	0	5.4163e-008	1.2792e-007
200	0	2.0800e-007	2.8889e-007
225	0	6.5848e-008	4.6569e-007
25	0	2.8084e-007	2.2777e-007
275	0	2.7651e-007	7.3965e-007

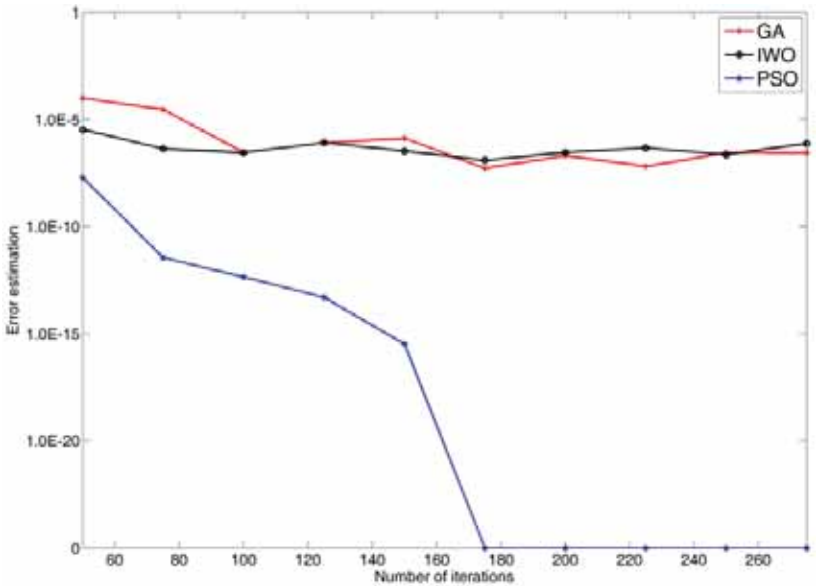

Fig. 17: Error estimation x_2 for Example 3 by using test # 3.

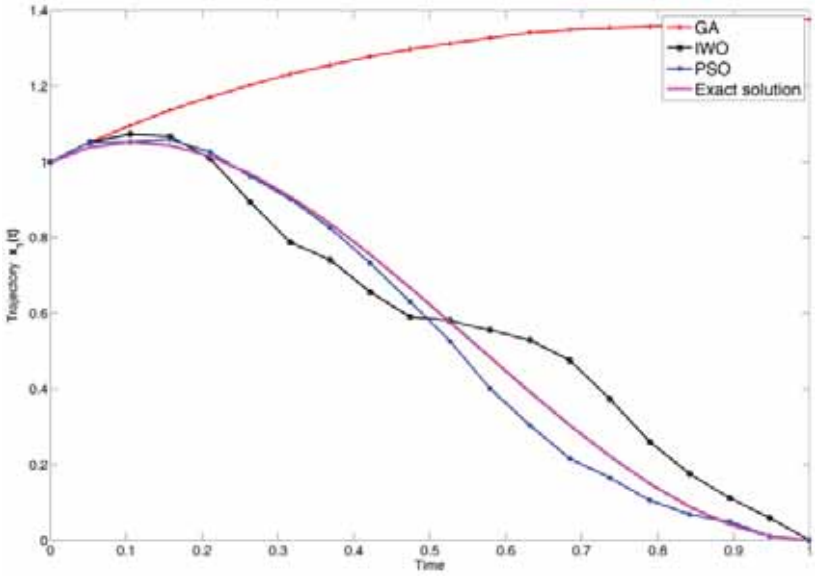

Fig. 18: Trajectory function x_1 for Example 3.

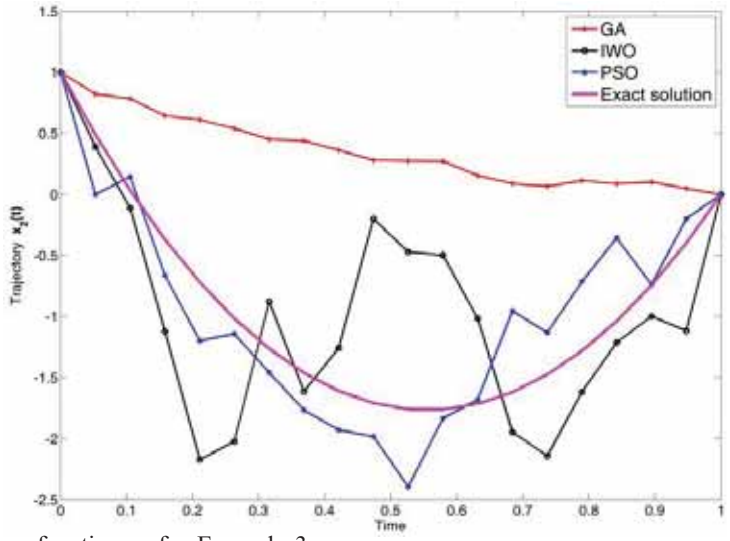

Fig. 19: Trajectory function x_2 for Example 3.

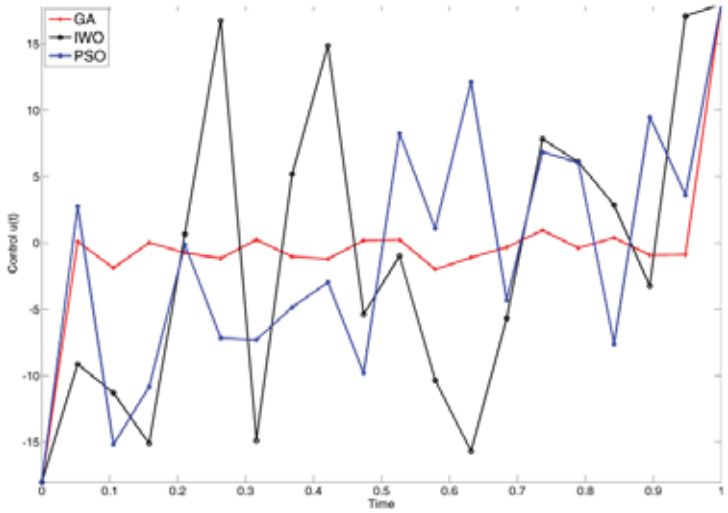

Fig. 20: Control functions for Example 3.

Conclusions:

In this paper we tried to apply the benefits of two evolutionary algorithms, PSO and IWO, to obtain approximate solution of optimal control problems. To this means, we proposed an special discretization of control state and then change the procedure of this algorithms to obtain the best solution. Numerical results show the accuracy of the method in final conditions. Of course success of PSO, in comparison to two other algorithms is obvious. Comparisons show that the number of iterations, populations size and the number of iterations effect on the complexity of methods but the nonlinearity of the objective and system have no serious effect on the procedure of applying this approaches. In the case of large discrete optimal control problems, the given approach may be implemented on parallel computers to save the computational time.

REFERENCES

Dsidri, J.A., S. Peigin and S. Timchenko, 1999. Application of Genetic Algorithms to the Solution of the Space Vehicle Reentry Trajectory Optimization Problem. French National Inst. for Research in Computer Science and Control (INRIA) Research Rept. No. 3843, Sophia Antipolis, France.

Borzabadi, A.H. and H.H. Mehne, 2009. Ant Colony Optimization for Optimal Control Problems. Journal of Information and Computing Science, 4(4): 259-264.

Jones, K.O., 2006. Comparison of genetic algorithms and particle swarm optimization for fermentation feed prole determination. Int. Conf. on Computer Systems and Tecnologies.

Kennedy, J. and R.C. Eberhart, 1995. Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, 1942-1948.

Mehrabian, A.R. and C. Lucas, 2006. A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics, 1: 355–366.

Fard, O.F. and A.H. Borzabadi, 2007. Optimal Control Problem, Quasi-Assignment Problem and Genetic Algorithm. Transactions on Engineering, Computing and Technology, 19: 422 - 424.

Schmidt, W.H., 2006. Numerical Methods for Optimal Control Problems with ODE or Integral Equations. Lecture Notes in Computer Science, 255 - 262.

Shi, Y., 2004. Particle Swarm Optimization. IEEE Neural Networks Society.

Wuerl, A., T. Crain and E. Braden, 2003. Genetic Algorithms and Calculus of Variations Based Trajectory Optimization Technique. Journal of Spacecraft and Rockets, 40(6): 882-888.