J.Mech.Cont. & Math. Sci., Vol.-7, No.-2, Januaryy (2013) Pages 1077-1093

BOX PUSHING USING HYBRID ABC-NSGAII ALGORITHM

BY

[1]Sudipta Ghosh, [2]Sudeshna Mukherjee and [3]Gopal Pal

[1]Deptt of Electronics & Communication Engineering,

Camellia School of Engineering &Technology,Barasat,Kolkata,West Bengal,India

[2]Cognizant Technology Solution Pvt Ltd,Saltlake,Sector-V,Kolkata,West Bengal,India

[3]Camellia Group of Engineering Colleges , Barasat ,Kolkata,West Bengal India

And IIM Kozhikode , Kerala, India

Abstract

In this paper, we present a novel method of path optimization using box pushing method and implementing ABC algorithm in combination with NSGAII Algorithm to achieve optimization. Here, in this case a Multi-Objective Function Optimization is carried out using Bees Colony Optimization and NSGAII Algorithm.

Keywords and phrases : box pushing, ABC algorithm, NSGAII, Bees Colony Optimization

বিমূর্ত সার *(Bengali version of the Abstract)*

র্সব্বোৎকৃষ্টতা অর্জনের জন্য NSGA II এ্যালগোরিদম্ - এর সঙ্গে ABC এ্যালগোরিদম্ - এর সমন্বয়কে কার্যকরী করে এবং বক্স পুশিং (Box Pushing) পদ্ধতিকে ব্যবহার করে পথের (Path) র্সব্বোৎকৃষ্টতার এক অভিনব পদ্ধতিকে এই পত্রে আমরা উপস্থাপন করেছি । এখানে এই ক্ষেত্রে বীজ্ কোলোনী র্সব্বোৎকৃষ্টতা (Bees Colony Optimization) এবং NSGA II এ্যালগোরিদম্‌কে ব্যবহার করে বহু বস্তুমুখী অপেক্ষকের র্সব্বোৎকৃষ্টতাকে সম্পাদন করেছি ।

1. INTRODUCTION

Box Pushing represents an interesting method of optimization where a box is rotated by applying torque on one of its edges thereby checking for any collisions with obstacles while being rotated. The utility of using multi objective function is that we can optimize both torque and path traversed at a same time. While rotating the box if no collision occurs then the position attained by the rotation is taken as a solution to optimization. However the rotation i.e. the direction of rotation is decided by the position of goal. The direction of rotation is such that it minimizes the distance traversed. Now, if there is a collision with obstacles while rotating through some angle, the course of rotation i.e. angle and direction of torque applied is changed and the next best-possible direction and angle of torque is taken into consideration. The former solution is not updated or appended to the solution list. In this way, a list of optimized solutions in terms of torque and path traversed is obtained which gives the optimized path for reaching the destination.

2. EXAMPLES OF BOX-PUSHING:

Two robots participating in moving the box are shown in figure below. in the figure, initial and final positions are marked by solid lines and intermediate posions by dashed lines. two steps followed are:

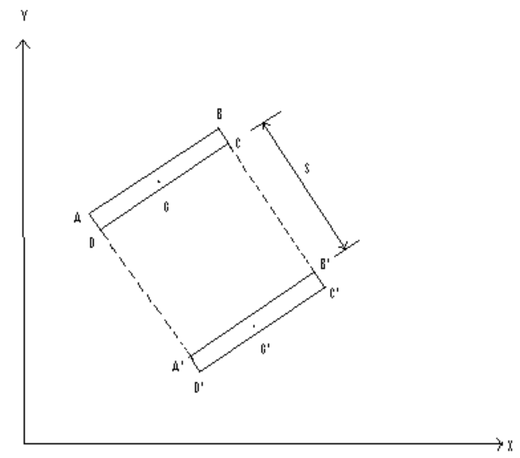
Step1: comprise of translation operation where one robot pushes the box and other one pulls it in order to move it along its width.

Step2: comprise of rotation operation where two robots apply equal and opposite forces to bring about rotation of the box about its centre.

3. MATHEMATICAL EXPRESSIONS FOR BOX PUSHING PROBLEM:

let G (x_g, y_g) be the centre of gravity and A (x_a, y_a), B (x_b, y_b), C (x_c, y_c) and D (x_d, y_d) be the 4 vertices of the box.

1. for translatory motion:



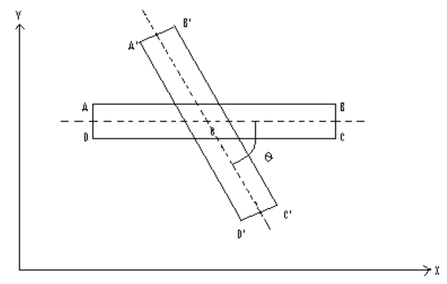
The following equations are derived:[11]

$Xg^{/} = Xg + S \cos \alpha \qquad Yg^{/} = Yg + S \sin \alpha$ (1)

The above equation is same for other co-ordinates.

$\alpha = \theta + 180^{o}$ (2)

2. for rotational:



The following equations are derived:[11]

$Xa^{/} = Xg(1 - \cos \theta) + Xa \cos \theta - (Yg - Ya) \sin \theta$ (3)
$Ya^{/} = Xg(1 - \cos \theta) + Ya \cos \theta - (Yg - Ya) \sin \theta$

Same equation is applied for other co-ordinates.

3. for combined translatory and rotational:

Co-ordinates after rotational movement:[1]

$$Xa^{/} = Xg(1 - \cos\theta) + Xa\cos\theta - (Yg - Ya)\sin\theta \quad (4)$$
$$Ya^{/} = Xg(1 - \cos\theta) + Ya\cos\theta - (Yg - Ya)\sin\theta$$

The other co-ordinates are updated using the same formulae.

Co-ordinates after translatory movement:[11]

$Xg^{/} = Xg + S\cos\alpha$ $\quad Yg^{/} = Yg + S\sin\alpha$ (5)

Same formula is applied for other co-ordinates.

4. OBJECTIVE FUNCTION VALUES:
   a. *Time objective:*

It is derived from

- $\acute{\omega} = T/J$ (6)
- $\theta(t) = \frac{1}{2}\acute{\omega}t^2$ (7)
- $s = \frac{1}{2}at^2$ (8)
- $F = ma$ (9)

.Where

J = Moment of Inertia ; ώ = Angular acceleration; a = Linear acceleration

From 7 $\theta(t) = \frac{1}{2}\acute{\omega}t^{2}$; From(6) $T = J$; $t = \sqrt{\frac{2\theta}{\acute{\omega}}}$

or, $\acute{\omega} = T/J$, $t = \sqrt{\frac{2\,\theta(t)}{T/J}}$ Say, t = t1, θ(t) = α(t), $t1 = \sqrt{\frac{2\,\alpha(t)}{T/J}}$

From (8), $s = 1/2a$, $t = \sqrt{}$ and From (9) , $F = m$, $t = \sqrt{\frac{2}{}}$

$s = d(t)$t), $t = \sqrt{\frac{2ms}{F}}$

F = F1t + F2t; T = t2, $t = \sqrt{\frac{2\,md(t)}{F1t+F2t}}$

From (8), $s = \frac{1}{2}at^2$, $t = \sqrt{\frac{2s}{a}}$

For constana value of 'a', $\sqrt{\frac{2}{a}} = Kt$; Or, $t3 = Kt\sqrt{s}$, $t3 \; \alpha \; Kt\sqrt{s}$

Total objective function

T = t1 + t2 + t3; $T = \sqrt{\frac{2\,\alpha(t)}{T/I}} + \sqrt{\frac{2m\; d(t)}{F1t+F2t}} + Kt\sqrt{s}$ (9a)

2. Energy Objective:

$T = F_{1r}d_1 + F_{2r}d_2$

where F_{1r} = force applied by 1st robot, F_{2r} = force applied by 2nd robot

$T = 2F_rd$ for $F_{r1}=F_{r2}=F_r$, $d_1=d_2=d$

- Energy for translation:

$E1 = (F_{1t} + F_{2t})d(t), \quad = 2F_{1t}d(t)$

- Energy for rotation:

$E2 = T\alpha(t), \quad = 2F_{1r}\, d_1\alpha(t)$

- Energy for transportation:

$E3 = K_eS$ where K_e is a constant.

$E = E1 + E2 + E3$ (9b)

5. ALGORITHMS:

In this paper, two algorithms are combined to obtain optimization:

1. Non Dominated Sorting Genetic Algorithm.
2. Bees Colony optimization Algorithm.

A. Non Dominated Sorting Genetic Algorithm:

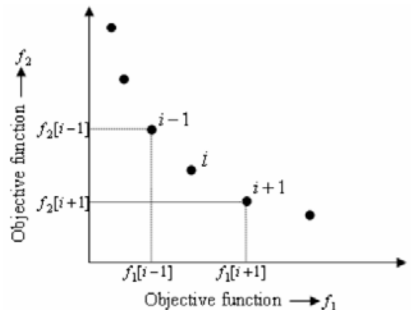
Due to its better spread of solutions and better convergence near the true *Pareto-optimal front*, [1]low computational requirements, elitism, and parameter optimizing, simple constraint handling strategy, it is widely used. Like many other evolutionary algorithms, in NSGA-II

also at first, an initial population called parent population *P0* (at time t=0) of size N is randomly generated. Then the population is sorted according to non-domination. Subsequent generations can be represented by discrete time steps: t = 1, 2, ...etc. After initialization, an iterative optimization process begins, where at the first step, using genetic operations i.e. binary tournament selection, recombination, and mutation operations child population *Qt* of the same size N, is generated from the parent population P*t*. Next, the parent and the child populations are combined to form the merged population *Rt* i.e. Rt = Pt U Qt , which is of size 2N. Then, the next population Pt+1 is constructed by choosing the best N solutions from the merged population Rt . Each solution is evaluated by using its rank as primary criterion and crowding distance as secondary.The ranking is done based on the non-domination. All the non-dominated solutions in the merged population are assigned rank 1. The rank 1 solution set is called front set F1. We now remove these solutions from the merged population, and again look for non-dominated solutions, if any, from the reduced merged population, and then assign rank 2 to these non-dominated solutions. The list of non dominated solutions thus obtained is called front set F2. In this way, rank is assigned to all the solutions. The members of the population Pt+1 are chosen from subsequent non dominated fronts in order of their ranking. Let F1 is the set, beyond which no other set can be accommodated. If by adding set F1 to Pt+1 , size of Pt+1 exceeds the population size then to select some solutions (N- Pt+1) from F1 , the set will be sorted based on the crowding distance, and the solutions with higher crowding distance are chosen. For maintaining good spread of solutions in the obtained set of solutions, the crowding distance concept has been introduced instead of choosing random solutions from *Fl* . Crowding distance of a solution is the sum of the difference between the function values of two adjacent solutions for all objectives i.e., to determine crowding distance of a solution, we have to sort the population according to each of the objective function value. Then, for each objective function, the solutions with maximum and minimum objective values are assigned infinite distance value, and the other intermediate solutions are assigned a distance value by taking the difference of the function value of their adjacent solutions.the following figure shows the solutions for two objective functions *f1* and *f2* . Then from the above discussion, the distance value of the *i*-th solution, for the 1st and the 2nd objective functions will be,[5]

$CR_{1\,distance}[i] = f_1[i+1] - f_1[i-1]$

$CR_{2\,distance}[i] = f_2[i-1] - f_2[i+1]$ respectively.

After calculating all the distance values for a solution, the crowding distance for the solution is obtained by taking the sum of the distance values corresponding to each objective functions. In this way, the crowding distances for all the solutions are obtained and according to the crowding distance, solutions from F1 are selected.[1]



Thus, by using ranking and crowding distance concept next population *Pt+1* is generated. This process is repeated fo

certain number of time steps, or until some acceptable solution has been found by the algorithm.

B. Artificial Bees Colony Optimization:

In ABC, the colony of artificial bees contains three groups of bees: employed bees associated with specific food sources, onlooker bees watching the dance of employed bees within the hive to choose a food source, and scout bees searching for food sources randomly. Both onlookers and scouts are also called unemployed bees. Initially, all food source positions are discovered by scout bees. Thereafter, the nectar of food sources are exploited by employed bees and onlooker bees, and this continual exploitation will ultimately cause them to become exhausted. Then, the employed bee which was exploiting the exhausted food source becomes a scout bee in search of further food sources once again. In other words, the employed bee whose food source has been exhausted becomes a scout bee. In ABC, the position of a food source represents a possible solution to the problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of employed bees is equal to the number of food sources (solutions) since each employed bee is associated with one and only one food source.

a. Employed Bees Phase

Employed bees search for new food sources ($\vec{v_m}$) having more nectar within the neighbourhood of the food source ($\vec{x_m}$) in their memory. They find a neighbour food source and then evaluate its profitability (fitness). For example, they can determine a neighbour food source $\vec{v_m}$ using the formula given by equation [2],[3]:

$$v_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki}) \quad (10)$$

where $\vec{x_k}$ is a randomly selected food source, i is a randomly chosen parameter index and ϕ_{mi} is a random number within the range $[-a, a]$. After producing the new food source $\vec{v_m}$, its fitness is calculated and a greedy selection is applied between $\vec{v_m}$ and $\vec{x_m}$.

The fitness value of the solution $fit_m(\vec{x_m})$, might be calculated for minimization problems using the following formula [2],[3]:

$$fit_m(\vec{x_m}) = \begin{Bmatrix} \frac{1}{1+f_m(\vec{x_m})} & \text{if } f_m(\vec{x_m}) \geq 0 \\ 1 + abs(f_m(\vec{x_m})) & \text{if } f_m(\vec{x_m}) < 0 \end{Bmatrix} \tag{11}$$

where $f_m(\vec{x_m})$ is the objective function value of solution $\vec{x_m}$.

a. *Onlooker Bees Phase*

Unemployed bees consist of two groups of bees: onlooker bees and scouts. Employed bees share their food source information with onlooker bees waiting in the hive and then onlooker bees probabilistically choose their food sources depending on this information. In ABC, an onlooker bee chooses a food source depending on the probability values calculated using the fitness values provided by employed bees. For this purpose, a fitness based selection technique can be used, such as the roulette wheel selection method (Goldberg, 1989).

The probability value p_m with which $\vec{x_m}$ is chosen by an onlooker bee can be calculated by using the expression given in equation [2],[3]

$$p_m = \frac{fit_m(\vec{x_m})}{\sum\limits_{m=1}^{SN} fit_m(\vec{x_m})} \tag{12}$$

After a food source $\vec{x_m}$ for an onlooker bee is probabilistically chosen, a neighbourhood source $\vec{v_m}$ is determined by using equation (10), and its fitness value is computed. As in the employed bees phase, a greedy selection is applied between $\vec{v_m}$ and $\vec{x_m}$. Hence, more onlookers are recruited to richer sources and positive feedback behaviour appears.

b. *Scout Bees Phase:*

The unemployed bees who choose their food sources randomly are called scouts. Employed bees whose solutions cannot be improved through a predetermined number of trials, specified by the user of the ABC algorithm and called “limit” or "abandonment criteria" herein, become scouts and their solutions are abandoned. Then, the converted scouts start to search for new solutions, randomly. For instance, if solution $\vec{x_m}$ has been abandoned, the new solution discovered by the scout who was the employed bee of $\vec{x_m}$ can be defined by(10). Hence those sources which are initially poor or have been made poor by exploitation are abandoned and negative feedback behaviour arises to balance the positive feedback.

Modifications:

Certain modifications were done in order to apply both NSGAII and ABC algorithms in unision to obtain an improved hybrid algorithm.

The hybrid algorithm is explained below:

1. The initial population of N solutions is initialized by using the scout bee's initialization formula. Each solution is a D dimensional vector. Box parameters are initialized as $x_{curnt} = x_c$ and $y_{curnt} = y_c$

2. Employed bee selects a solution from N no. of solutions. Out of the remaining N-1 solutions, another solution is selected randomly. And the same field of the previously selected solution is exchanged with that of the randomly selected solution. All other fields of the vector remain same. Thus we get an adapted vector from the original vector.

3. Now, the original and the adapted vector are compared and *m* unique food-sources are taken out of the N food-sources ($m<N$) using non-dominated sorting based on two objective functions stated in the equations (9a) and (9b) by employed bee. Rest N-*m* food-sources have 2(N-m) no of non-dominated solutions. The best (N-m) solutions are to be selected out of 2(N-*m)* solutions using crowding distance. Finally the set of solutions obtained is named as R_t.

4. After all employed bees complete the search process, they share the nectar information of the food sources (solutions) and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information from all employed bees and chooses a food source depending on the probability value associated with that food source calculated by expression stated in equation 12. where fi is the fitness value of the solution i evaluated by its employed bee, which is proportional to the nectar amount of the food source in the position i and N is the number of food sources which is equal to the number of employed bees.
5. After that, as in case of employed bee, onlooker bee produces a modification on the position (solution) in her memory using equation 1-8 and checks the nectar amount of the candidate source (solution). Providing that its fitness is better than that of the previous one, bee memorizes the new position and forgets the old one.
6. Memorize the best solution obtained so far. If any food source gets exhausted, or abandoned by employed bee then it's replaced with new solution and the parameters of the solution being randomly produced by scout bee.
7. Above mentioned steps (step 1 - 6) are repeated until the termination criteria i.e. ($|x_{cg}$-$x_{curnt}|$and$|y_{cg}$-$y_{curnt}|$) <β is met. Here β is an arbitrarily small number and (x_{cg} ,y_{cg}) is the co-ordinate of the centre of gravity of the box.

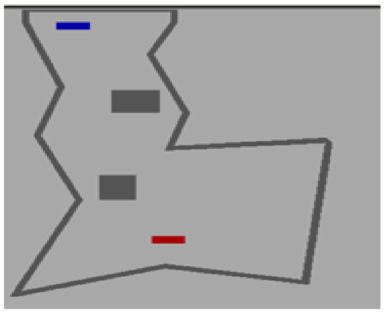
6. PSEUDOCODE:

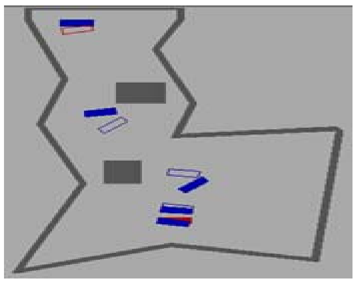
```
Input: Initial CG of the box $(x_c, y_c)$, final CG of the box
$(x_{cg}, y_{cg})$.
  Output:
  Forces applied by the robots(ForceA,ForceB), Average Energy, Average Time.
  Begin:

  Set for the box:
$x_{curr} \leftarrow x_c ; y_{curr} \leftarrow y_c$

Repeat
Call ABC-NSGAII $(x_{curr}, y_{curr})$

      Update $(x_{curr}, y_{curr})$ in each step using eqns. 1 to 5.
Until $(|x_{cg} - x_{curr}|$ and $|y_{cg} - y_{curr}|) < \xi$
```

```
/* $\xi$ is an arbitrarily small no.*/

  End

Procedure ABC-NSGAII ($x_{curr}, y_{curr}$)

Begin:

Initialize all the food sources (original populations).
Initialize problem parameters
Set Iteration to 1.
  Repeat
      For each employed bee
       {
         Produce a new solution $v_{mi}$ from equation (10).
         Create an adapted solution from original solution.
         Calculate the fitness value of each solution.
         Apply greedy selection process.
       }
      For each onlooker bee
       {
Select the food source to be modified based on its probability of being chosen as given in equation (12).
Produce new solution using the same equation (10).
Calculate its fitness value.
Apply greedy selection process.
         }
     Memorize the best solution obtained so far.
    If any food source gets exhausted, or abandoned by the employed bees then it's replaced with new solution, the parameters of the solution being randomly produced by the scout bees.
  Iteration incremented by 1.
Until acceptable solution is attained.

Return
```

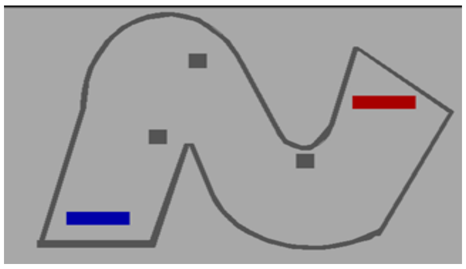
VII. Experiment And Computer simulation

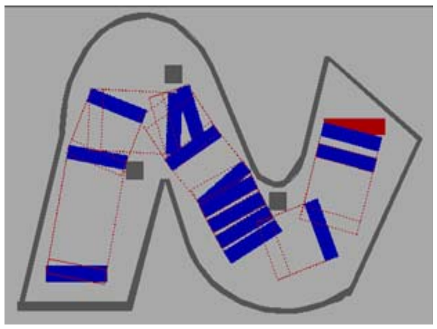
ARENA1 USING NSGAII



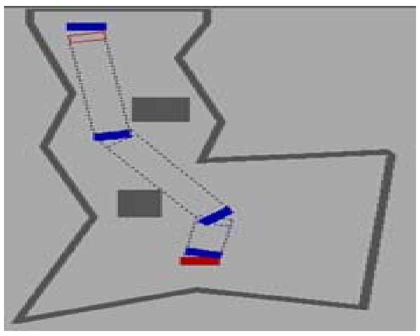


ARENA2 USING NSGA II





ARENA1 USING ABC-NSGAII



ARENA2 USING ABC-NSGA II

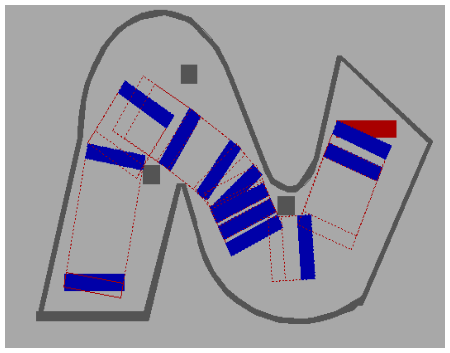


TABLE-1 FOR ARENA1
TURNING FORCES AND ANGLE OBTAINED IN BCO-NSGA II

Step	F_{1r}	F_{2r}	α	x_i	y_i
1	4.837599	4.161203	-0.162928	87.519548	30.00000
2	1.316298	0.534971	-0.503309	94.402448	142.879952
3	2.441566	0.181172	0.859329	152.502792	240.383225

TABLE-2 FOR ARENA1
FORCES FOR TRANSLATION, NEXT CENTRE OF GRAVITY POSITION, TIME AND ENERGY CONSUMPTION

Step	F_{1T}	X_c	Y_c	Time	Energy
1	2.660747	108.145629	140.620776	46.214202	405.216247
2	2.949634	174.861680	222.805832	44.682322	362.390612
3	2.240182	173.327822	281.966587	30.629678	137.001831

TABLE-3 FOR ARENA1
COMPARISON BETWEEN CURRENT AND PREVIOUS WORK

Method used	Total Time	Total Energy
BCO-NSGA II	121.526202	904.608690
NSGA II	170.184352	1698.317755

TABLE-4 FOR ARENA2
TURNING FORCES AND ANGLE OBTAINED IN BCO-NSGA II

STEP	**F_{1R}**	**F_{2R}**	**α**	**X_i**	**Y_i**
1	2.963915	13.436211	0.153859	216.072142	308.000000
2	14.365552	6.098516	0.167977	202.059193	175.462524
3	1.266601	0.169015	1.286237	210.769701	112.098426
4	0.258993	8.883624	1.042880	307.988367	176.040632
5	0.592326	12.197916	0.006447	328.527211	203.840576
6	1.359335	21.294524	-0.011449	340.152857	223.404154
7	0.601254	20.787651	0.001463	344.730793	243.813752
8	0.846957	5.912156	-0.027469	363.392290	256.354302
9	0.829737	0.124360	-1.101164	415.846034	248.545959
10	7.621267	1.206977	-1.145045	459.583100	246.207835
11	6.131635	6.865995	-0.003819	512.692990	179.098199

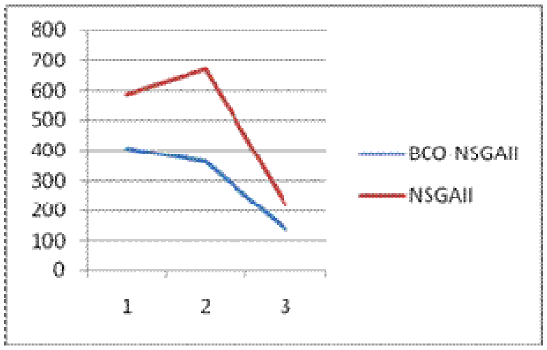
TABLE-5 FOR ARENA2
FORCES FOR TRANSLATION, NEXT CENTRE OF GRAVITY POSITION, TIME AND ENERGY CONSUMPTION

Step	F_{1T}	X_c	Y_c	Time	Energy
1	1.583060	215.232992	177.505572	142.852288	253.253323
2	6.125331	234.704148	120.078836	0.172409	480.000013
3	4.995232	309.292051	141.082992	2.556292	522.513420
4	3.703075	354.938309	189.731481	49.770547	163.361894
5	3.994147	365.678029	209.556059	37.819901	91.306847
6	6.006211	375.271957	226.788897	28.975282	122.391219
7	6.186300	385.010327	244.344984	28.489587	124.000188
8	3.256636	394.852086	260.975109	39.354098	67.829678
9	5.724099	461.138324	270.133974	2.777309	266.416754
10	18.025634	510.948578	178.439921	0.912427	1627.280414
11	2.488391	520.496553	52.294436	52.294436	69.289608

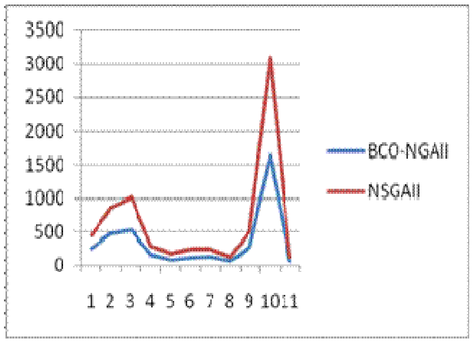
TABLE-6 FOR ARENA2
COMPARISON BETWEEN CURRENT AND PREVIOUS WORK

Method used	Total Time	Total energy
BCO-NSGA II	385.974578	3787.643357
NSGA II	542.046899	6188.945924

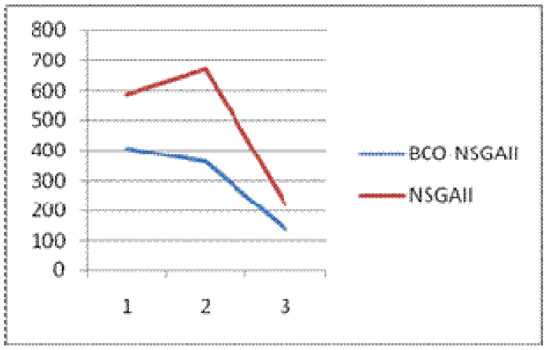
ENERGY COMPARISON GRAPH FOR ARENA 1

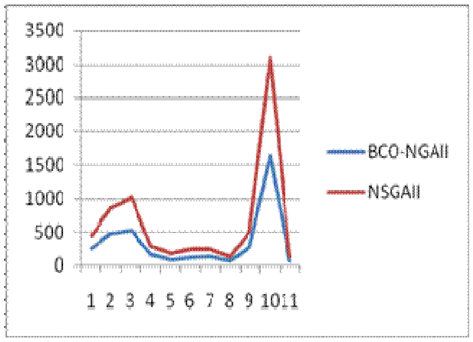


ENERGY COMPARISON GRAPH FOR ARENA 2



ENERGY COMPARISON GRAPH FOR ARENA 1 **ENERGY COMPARISON GRAPH FOR ARENA 2**





7. Conclusions

In this paper we have successfully developed a hybrid algorithm of ABC-NSGAII which is definitely a better approach to solve the box-pushing problem by two robots. The table showing the solutions clearly reveals the betterment. Further, the above algorithm was applied to all simulated test environments and it was found to work quite well in all such cases. Thus, the proposed scheme is a better approach to solve two-robot box-pushing problems.

References

1) Chakrabarty J., Konar A., Nagar A., Das S., "Rotation and translation selective Pareto optimal solution to the box-pushing problem by mobile robots using NSGA-II" IEEE CEC 2009

2) Karaboga Dervis, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, October, 2005

3) Karaboga D., Basturk B., On the Performance of Artificial Bee Colony Algorithm, received in revised form 9 January 2007; accepted 30 May 2007

4) Alatas Bilal, Chaotic Bee Colony Algorithm for Global Numerical Optimization

5) Deb K., Agarwal A. P. S., and Meyarivan T., "A fast and elitist multiobjective genetic Algorithm: NSGAII"

6) Kube C. R., and Zhang H., "The use of perceptual cues in multi-robot box pushing," in IEEE International Conference on Robotics and Automation, 1996, vol. 3, pp. 2085-2090

7) Yamada S., and Saito J., "Adaptive action selection without explicit communication for multi-robot box-pushing," in IEEE International Conference on Intelligent Robots and Systems, 1999, pp. 1444 -1449.

8) Chakraborty J., Konar A., Nagar A., Tawfik H., "A multi-objective Pareto-optimal solution to the box-pushing problem by mobile robots," Second UKSIM European Symposium on Computer Modeling and Simulation, pp.70-75, 2008.

9) Mataric M. J., Nilsson M., and Simsarian K. T., "*Cooperative multirobot box-pushing," In IEEE International Conference on Intelligent Robots and Systems*, 1995, vol. 3, pp.556-561.

10) Parker L.E., Tang F. , "Building multi-robot coalitions through automated task solution synthesis" *Proceedings of IEEE Vol.94*, No.7,July 2006.