# MOSDA: A Proposal for Multiple Objective Spiral Dynamics Algorithm

A. Azwan A.R., A.N.K. Nasir, Sha'akmal S., M. Sawal A.R.
*Faculty of Electrical & Electronics Engineering, Universiti Malaysia Pahang,*
*26600 Pekan, Pahang Malaysia.*
*ahmadazwan.ar@gmail.com*

***Abstract*—This paper proposed a multi-objective spiral dynamic algorithm (MOSDA) to solve multiple objectives problems. SDA is originally a single objective optimizer that inspired based on the spiral phenomena in nature. It has a good elitism strategy and has a simple structure. A method called "archive method" that is used in multi-objective particle swarm optimization (MOPSO) is adopted into SDA to develop its multi-objective (MO) type algorithm. Moreover, MOSDA is formulated by applying the widely-used concept of Pareto dominance to determine the movement of the particles and at the same time, the algorithm maintains the non-dominated solution in a setup global repository. These non-dominated solutions then will be used to guide other particles to move. The proposed algorithm is tested with several benchmark functions for multi-objective problems. Pareto front (PF) graphs are presented as the results of these tests. The accuracy and diversity of the produced PF are highly competitive compared to MOPSO.**

***Index Terms*—Metaheuristics; Multipoint Search; Multi-Objective Algorithm; Nature Inspired Computation; Spiral Dynamics.**

## I. INTRODUCTION

Recently, a multi-objective optimization technique is very important in solving problems that have several conflicting aims. This invites many scientists and researchers to involve in this area in order to solve real-world engineering design. From the study, many algorithms and methods are developed to deal with these problems. They also face challenges to provide an algorithm which has a low computation cost. Moreover, researchers also try hard to provide algorithms that can provide high accuracy and diversity PF solution.

Differs to single-objective type algorithm [1], MO type algorithm does not provide a single solution. Particle swarm optimization (PSO) [2] and genetic algorithm (GA) [3] are some of the good single-objective algorithms while PSO multi-objective version, MOPSO [4] is widely used algorithm used in many applications. To solve the multiple-conflicting problems, an optimum design solution can be obtained by adopting the concept of Pareto dominance. This concept provides the best solution set, which is defined as non-dominated solutions or Pareto-optimal solution.

As the fast convergence rate and accuracy [5] are important to provide good PF solution set, the metaheuristic elements are the best option to be adopted into MO algorithm. In the literature study, there are a lot of MO type algorithm. Other than MOPSO, the fast elitist non-dominated sorting genetic algorithm (NSGAII) [6] and multi-objective differential evolutionary (MODE) are some of the well-known MO algorithms. The goodness of metaheuristic techniques leads the researchers to develop numerous algorithm over the last decades.

Nowadays, spiral dynamics algorithm (SDA) [7] is one of the most recent introduced metaheuristic algorithm. This type of algorithm is a derivative-free or non-gradient based optimization algorithm. The algorithm has a simple structure that has good exploitation and high-speed computing time. Despite its great abilities, SDA remains one of the metaheuristic algorithms which is not extended to become a multiple-conflicting objectives problem solver as there is no such literature found.

For some explanation, MOPSO is initially proposed in 1999 by Moore et. al. [8]. This MOPSO then becomes the motivation to other researchers to extend this version of SOPSO. However, the version of Coello et. al. [4] is stand out as the main reference for MOPSO. MOPSO is also a complex structure algorithm that able to provide a diverse and accurate PF. However, it has a problem, in which the size of the archive increases very quickly. An archive that needs to be updated for each iteration will lead the computing cost to become higher.

In this paper, a new algorithm called "multi-objective spiral dynamics algorithm" is introduced. Our approach aims to make SDA capable to solve multiple-conflicting objectives problem. This paper describes the detail of the study which compares the performance of SDA and PSO after both of them are adopted with archive method. PF solution provided by both algorithms are also shown for comparison. The remaining section of this paper is organized as follows: Section II explains briefly about SDA, Section III explains archiving method in detail, Section IV describes MOSDA in detail, Section V explains the benchmark function setup, Section VI discusses the results. The conclusion and future works are explained in Section VII.

## II. SPIRAL DYNAMICS ALGORITHM

In 2010, Kenichi Tamura and Keiichiro Yasuda [9] introduced an algorithm in class of metaheuristic named spiral dynamics optimization algorithm (SDOA) based on an analogy of spiral phenomena in nature [10], [11], [17]. The methods inspired because of the movement of the particles in spiral steps generates logarithmic spirals seems to have a great strategy of solution searching which described as "diversification in the first half and intensification the second half" [11] and illustrated in Figure 1.

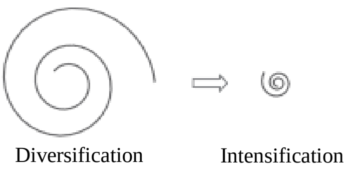


Figure 1: Logarithmic of spiral, ‘diversification and intensification illustration.

The algorithm has two parameters that required to be specified – first is convergence rate, r and second is rotation rate, $\boldsymbol{\theta}$ which important to specify the trajectory of the step movement. The spiral model search agents which defined as $\boldsymbol{x(k)} \in \mathbb{R}^{\boldsymbol{n}}$ converges to the arbitrary centre $\boldsymbol{x}^* \in \mathbb{R}^{\boldsymbol{n}}$ with a spiral logarithm is defined as follow [8]:

$$x(k+1) = S(r,\theta)x(k) - (S(r,\theta) - I)x^* \qquad (1)$$

where $\boldsymbol{x}^*$ is the centre of a spiral, $\boldsymbol{S_n}$ is a multiplication of radius, r and composition of rotational $\boldsymbol{n \times n}$ matrix, $\boldsymbol{R^n}$ based on the combination of all axes, $x$ is a coordinate location of a point and k is the iteration number.

Table 1
Parameters for SDA

| Symbols | Meaning |
|---|---|
| $\boldsymbol{\theta_{i,j}}$ | Angular displacement of search points. |
| $\boldsymbol{r}$ | Spiral radius. |
| $\boldsymbol{m \geq 2}$ | Number of search points |
| $\boldsymbol{k_{max}}$ | Maximum iteration. |
| $\boldsymbol{x_i(k)}$ | Position of $\mathrm{i_{th}}$ point in $\mathrm{k_{th}}$ generation. |
| $\mathbb{R}^{\boldsymbol{n}}$ | Composition of rotational n × n. |

```
Step 0: Preparation
Select the number of the search points $m \geq 2$, parameters $0 \leq \theta < 2\pi, 0 \leq r < 1$, of $\mathrm{S_n(r, \theta)}$ and maximum iteration number, $k_{max}$. Set $k = 0$.

Step 1: Initialization
Set initial points $x_i(0) \in R^n, i = 1,2,3, \dots, min$ the feasible region at random manner and center $x*$ as $x^* = x_{i_g}(0). i_g = arg\,min_i\ f\big(x_i(0)\big), i = 1,2,3, \dots., m.$

Step 2: Updating position, $\boldsymbol{x_i}$
Move the agent a step ahead by equation:
    $x_i(k+1) = S(r,\theta)x(k) - (S(r,\theta) - I)x^*$

Step 3: Updating center of spiral, $\boldsymbol{x*}$
$x^* = x_{i_g}(k+1), i_g = arg\,min_i\ f(\big(x_i(0)\big), i = 1,2,3, \dots., m.$

Step 4: Checking termination criterion
 If $k = k_{max}$ then terminate. Otherwise set $k = k + 1$, and return to step 2.
```

## III. Archive Method Description

To find the solution for multiple-conflicting problems, the Pareto ranking scheme which explained by David E. Goldberg [12] is applied to the SDA. This method also used for many algorithms such as GA [13] and PSO [2] in order to convert them as multi-objective problem (MOP) optimizer. Some of these MO type algorithms called respectively as NSGA/NSGAII/NSGAIII [14], [6], [15] and MOPSO [4]. The particles population characteristic is initialized at the beginning with “Best Position” criteria, which denotes the best experiences or the best fitness value obtained by them. These values will be used to store non-dominated solutions generated previously. Based on the technique inspired by Pareto Archive Evolution Strategy (PAES) [16], a global repository is set up. This repository is the storage where the particles will deposit its movement experience after each iteration. Global attraction mechanism will be combined with the previous found non-dominated solution that leads the convergence towards a globally non-dominated solution. The particles stored in the repository will be updated after each iteration. The best-required number of solution (i.e. 100 solutions will be stored in 100 of empty space in the repository) will be ranked, and particles that exceeded from the repository space will be deleted. The top-ranked solution will be plotted and from this, the Pareto front can be generated.

## IV. Proposed Multiple Objective Spiral Dynamics Algorithm (MOSDA)

It is important to define the features of SDA in a correct way to make new MO type SDA as an efficient MO algorithm. As explained before, in the standard operation of SDA, only one objective function could be optimized. MOSDA is the abbreviation, which is a derived from SO type SDA.

In this paper, although SDA needs to optimize more than one function, there is no specific modification need to be done to the original code of SDA. SDA is hybridized with archiving method used in MOPSO. The center of the spiral, x* is determined after the component of selecting a leader in MOPSO determine the non-dominated solution (NS) in the repository. This function also sorts the members in ascending order. The mutation and crossover function, which denoted in NSGAII, are also applied in this MOSDA in order to create more randomness in the search points. However, these two functions only applied to specific members of the generated population. These population, which combined with the previous members in the repository space, will be updated by determining their domination and only NS will be kept in the repository. It can be said that it is significantly important to save the NS set in the archive in order to plot the PF. The update will be done for each iteration when all the dominated set will be eliminated. From literature, there are a lot of setting for the sizing of the archive. In this paper, archive size is limited to only 50 NS. The algorithm will rank the previous NS, the new inserted NS and the excess number of NS then will be eliminated. The pseudocode to describe the MOSDA is as follows.

```
Step 0: Preparation
Select number of search points, $m \geq 2$, number of variable dimension, n, parameters $0 \leq \theta < 2\pi$, $0 < r < 1$ of $S_n(r,\theta)$, and maximum number of iteration, $\mathrm{k_{max}}$. Set $\mathrm{k = 0}$.

Step 1: Initialization
  i. Create members of a population, which present the particles that have initial points, $x_i(0) \in R^n, i =$
```

$1,2,3,\ldots.,m$ in the feasible region. These particles randomly spread.

ii. Calculate fitness value of each particle.

iii. Determine domination for each member of the population.

iv. Archive the non-dominated solution into repository.

v. Generate hypercubes based on the members of repository fitness.

**Step 2: Define spiral step center**

Select a leader from the repository member. This leader will be initial center for the spiral step, x*.

**Step 3: Move the particles a step ahead in spiral step**

i. Spiral equation to update position.

$$x(k+1) = S(r,\theta)x(k) - (S(r,\theta) - I)x^*$$

ii. Calculate fitness for each new location.

**Step 4: Apply Mutation to a new population**

Apply mutation to a random n$^{th}$ number of member in population.

**Step 5: Apply Crossover to a new population**

Apply cross over for first two and last two members of the current population.

**Step 6: Update repository member**

Combine previous repository member, SDOA population, crossover population and mutated population in repository space. Determine their domination. Keep the only non-dominated solution in the repository.

**Step 7: Display Pareto front**

**Step 8: Check termination criteria**

If $k = k_{max}$, then terminate, otherwise return to **Step 2**

## V. BENCHMARK FUNCTION EVALUATION

MOSDOA will be compared against a recently used MO algorithm which is MOPSO. As stated by the author of MOPSO, the parameters set will be 50 particles, a repository size for 100 particles and 7 division for the adaptive grid. This setting, which set up by the user will be also applied to MOSDA. In order to validate this new MOSDA, several benchmark test functions were tested, which taken from E. Zitzler et. al. [18] The MOSDA was tested with four different benchmark functions. The average time taken to iterate from beginning to a maximum number of iterations is computed.

### *A. Test Function 1*

The first test function used is the Schaffer's [18] benchmark function for MO algorithm.

Schaffer: minimize

$$\begin{cases} f_1(x) = \theta^2, \\ f_2(x) = (\theta - 2)^2 \end{cases} \tag{2}$$

$$\text{where: } -5 \le x \le 5$$

This function was evaluated by 30,000 times of function evaluation (FE) by these two algorithms. From this problem, theoretical solution, it has a Pareto front that continuous.

### *B. Test Function 2*

The second test performed with Fonseca's [18] benchmark function for MOA.

Fonseca: minimize

$$\begin{cases} f_1(x) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right) \\ f_2(x) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \end{cases} \tag{3}$$

$$\text{where: } -4 \le x \le 4, \quad 1 \le i \le n$$

This function was evaluated by 30,000 times of FE by these two algorithms. From this problem theoretical solution, it has a Pareto front that continuous.

### *C. Test Function 3*

The third benchmark function to perform the validation is Kursawe' [18] benchmark function for MO algorithm.

Kursawe: minimize

$$\begin{cases} f_1(x) = \sum_{i=1}^{2}\left[-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right] \\ f_2(x) = \sum_{i=1}^{3}\left[|x_i|^{0.8} + 5\sin(x_i^3)\right] \end{cases} \tag{4}$$

$$\text{where: } -5 \le x \le 5, \quad 1 \le i \le 3$$

This function was evaluated by 50,000 times of FE by these two algorithms. From this problem theoretical solution, it has a Pareto front that discontinuous.

### *D. Test Function 4*

The fourth benchmark function to perform the validation is Poloni' [18] benchmark function for MO algorithm.

Poloni: minimize

$$\begin{cases} f_1(x) = [1 + (A_1 - B_1(x,y))^2 + (A_2 - B_2(x,y))^2] \\ f_2(x) = (x - 5)^2 \end{cases}$$

where:

$$\begin{aligned} A_1 &= 0.5\sin(1) - 2\cos(1) + \sin(2) - 1.5\cos(2) \\ A_2 &= 1.5\sin(1) - \cos(1) + 2\sin(2) - 0.5\cos(2) \\ B_1(x,y) &= 0.5\sin(x) - 2\cos(x) + \sin(y) \\ &\quad - 1.5\cos(y) \\ B_2(x,y) &= 1.5\sin(x) - \cos(x) + 2\sin(y) \\ &\quad - 0.5\cos(y) \end{aligned} \tag{5}$$

$$\text{and } -\pi \le x,y \le \pi$$

This function was evaluated by 30,000 times of FE by these two algorithms. From this problem theoretical solution, it has a Pareto front that discontinuous.

### *E. Experimental setup*

The simulation will be performed on a PC with Intel(R) Core(TM) i5-4440 CPU processor which runs at 3.10GHz, 8Gb of RAM and a hard drive of 2Tb. The MOSDA was coded in MATLAB and several parts of the codes were programmed in C++ language. The operating system of the PC used was Windows 8.1 Pro 64bit. To compare the results, a fair evaluation must be performed, therefore all of the tests were running on the same specification of PC.

### *F. Number of function evaluation (NFE)*

NFE is defined as the number of evaluation of a fitness function. These numbers are not same as the number of iteration, as the fitness function can be evaluated more than once in a single iteration. The maximum number of fitness evaluation is more preferred than the maximum number of iteration because a single fitness evaluation provides some information about the problem. Thus, if the number of fitness evaluation is set limited, then the amount of information that can be provided by the algorithm for a problem is limited. This is the reasonable way on how to compare algorithms. So, for our study, it was to run for the same number of maximum fitness evaluation. The more times for the algorithm evaluating a problem, it provides more chances to come up with a better solution. The number of NFE can be seen in Table 2.

## VI. Discussion Of Result

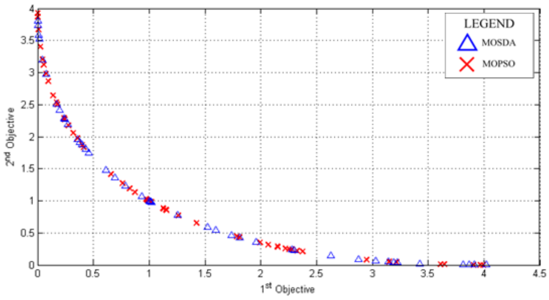
A number of particles are equivalent to the number of search points in SDA. The recommended number of particles is 50. A large number of particles can cause the slow computation speed of the algorithm.

The hypercubes also need to be determined. The unsuitable number of hypercubes cause the algorithm needs to be iterated at many more times, as it has to discover more division region. Size of repository represents the number of our desired number of non-dominated solution. It depends on the user on how many numbers of solution required. This might affect the diversity of the Pareto front a lot.

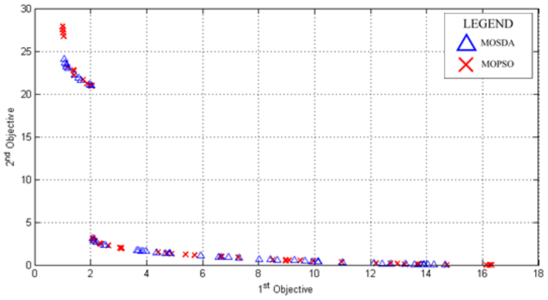
The proposed MOSDA which coded in MATLAB and task of optimization was executed using 25 independent runs. For all benchmark problems, the initial parameters for the MOSDA ($n_{population}, n_{grid}$ and $n_{repository}$) were set 50, 7, and 50 respectively. The maximum number of iteration varied for each problem. To evaluate, the maximum NFEs was taken as the stopping criterion. The NFEs set for each benchmark function can be seen in Table 2.

Table 2
Number of Function Evaluations (NFEs)

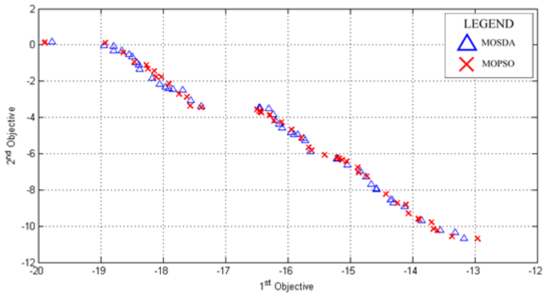
| Functions | NFEs |
|---|---|
| Schaffer | 30,000 times |
| Fonseca | 30,000 times |
| Kursawe | 100,000 times |
| Poloni | 30,000 times |



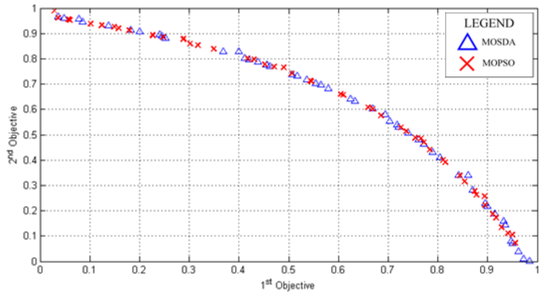
(a)



(b)



(c)



(d)

Figure 2: PF solution based on MOSDA and MOPSO algorithms. (a) Schaffer's function, (b) Fonseca's function, (c) Kursawe's function, (d) Poloni's function.

The plotted PF shown in Figure 2 show the comparison between MOSDA and MOPSO. The diversity and accuracy of the solution visibly distributed at the same level compared to PF plotted by MOPSO except for Kursawe. From the results of the MOSDA also, it can be concluded that the solution provided by the algorithms is comparable to MOPSO. As mentioned in Table 3, the time taken by MOPSO is clearly faster compared to MOSDA. This time depends on the number of function evaluation per iteration.

Table 3
Time of Iterations

| Function | Algorithm | |
|---|---|---|
| | MOSDA(secs) | MOPSO (secs) |
| Schaffer | 17.148 | **11.68** |
| Fonseca | 15.596 | **7.116** |
| Kursawe | 60.873 | **20.94** |
| Poloni | 15.838 | **7.529** |

Meanwhile, from Table 4, the result from the numerical analysis of the PF for both algorithm MOSDA and MOPSO is shown. The table indicates two parameters are measured from the PF: 1) generational distance (GD), which is defined as a criterion for the convergence between theoretical PF and produced PF [19]; and 2) metric of spacing (MOS) [19,20], which is defined as the distance of distributed non-dominates solution set along the PF. Both of two parameters are evaluated better when it getting smaller or approach to zero.

Table 4
GD and MOS of MOSDA and MOPSO.

| Parameters / Function | MOSDA | | MOPSO | |
|---|---|---|---|---|
| | GD | MOS | GD | MOS |
| Schaffer | **0.4216** | **2.53** | 4.8397 | 4.4355 |
| Fonseca | **0.074** | **0.071** | 0.6742 | 0.2317 |
| Kursawe | 6.6067 | **0.986** | **5.83** | 4.1426 |
| Poloni | 20.853 | **4.244** | **20.35** | 12.563 |

The variables increase the dimension of Cartesian vector space which its size needs to be defined according to the size of the variable. The problem from Kursawe's in this proposal, which the variables size is 3 lead the Cartesian vector space also need to in size of 3-dimension. This lead to the complex formulation of the Cartesian space. Hence, the computation will take longer than usual. Also for other three problems, the MOSDA is still the slower solution provider in finding the Pareto front solution. Although MOSDA has the simpler strategy to find the solution, MOSDA evaluates cost function 3 times per iteration compared to MOPSO (2 times per iteration only). This affects the time taken to produce a result. In term of the diversity of the solution, the MOSDA still cannot compete with MOPSO, but it is not too bad and still can be concluded as comparable to them. As a conclusion, for Kursawe's function test, MOSDA cannot truly find the solution exactly the same with its theoretical Pareto front, which this could be implemented for the future work.

## VII. Conclusion and Future Work

A new multi-objective spiral dynamics optimization algorithm (MOSDA) has been proposed for solving MOP. The single-objective problem solver Spiral dynamics algorithm has been modified to turn it into a MOP solution provider. In this paper, MOSDA has been tested with several benchmark functions. The result shows that the PF is comparable to MOPSO but MOSDA generates the PF slower. Even though SO SDA has simple strategy compared to SO PSO, in its MO-type version, MOSDA needs to evaluate function more times per iteration compared to MOPSO. The MOSDA which adopt archive method also still not compatible to operate the MO problem with more than two dimensions. Moreover, the performance PF solution has been evaluated by numerical analysis. For the future works, the MOSDA might be upgraded to hybridize it with new feature or element. This might lead MOSDA to be able to deal with up to 2 dimensions of the MO problem.

## Acknowledgment

This project has been supported by National Research Grant No. RDU160103 awarded by Ministry of Higher Education Malaysia and Electrical and Electronics Engineering Faculty, Universiti Malaysia Pahang, Malaysia.

## References

[1] Glover FW, Kochenberger GA (2003) *Handbook of metaheuristics*. Kluwer, Dordrecht

[2] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, pages 1942-1948, IEEE Press, Piscataway, NJ, 1995.

[3] K. F. Man, K. S. Tang and S. Kwong, "Genetic algorithms: concepts and applications [in engineering design]," in *IEEE Transactions on Industrial Electronics*, vol. 43, no. 5, pp. 519-534, Oct 1996.

[4] Coello, C.A.C., Lechuga, M.S.: MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In: *IEEE Congress on Evolutionary Computation*, Piscataway, New Jersey, pp. 1051–1056. IEEE Press, New York (2002)

[5] B. Suman, Study of self-stopping PDMOSA and performance measure in multiobjective optimization, *Comput. Chem. Eng*. 29 (2005) 1131–1147.

[6] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 2002;6(2):182–97.

[7] K. Tamura and K. Yasuda: Primary study of spiral dynamics inspired optimization, *IEEJ Transactions on Electrical and Electronic Engineering*, Vol. 6, No. S1, pp. S98–S100, 2011.

[8] Jacqueline Moore and Richard Chapman. *Application of particle swarm to multiobjective optimization*. Department of Computer Science and Software Engineering, Auburn University, 1999.

[9] K. Tamura and K. Yasuda: A Parameter Setting Method for Spiral Optimization from Stability Analysis of Dynamic Equilibrium Point, *SICE Journal of Control, Measurement, and System Integration*, Vol. 7, No. 3, pp. 173–182, May 2014

[10] K. Tamura and K. Yasuda: Primary study of spiral dynamics inspired optimization, *IEEJ Transactions on Electrical and Electronic Engineering*, Vol. 6, No. S1, pp. S98–S100, 2011.

[11] K. Tamura and K. Yasuda: Spiral dynamics inspired optimization, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 15, No. 8, pp. 1116–1122, 2011.

[12] David E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning*. Addison Wesley Publishing Company, Reading, Massachusetts, 1989.

[13] Dominik Gwiazda Tomasz. 2006. *Genetic Algorithms Reference*. Tomasz Gwiazda.

[14] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, UK: Wiley, 2001.

[15] Kalyanmoy Deb, Fellow, IEEE and Himanshu Jain, An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non Dominated Sorting Approach, Copyright (c) 2013 IEEE.

[16] Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation, 8(2):149 172,2000

[17] N.M.A Ghani, A.N.K Nasir & M.O. Tokhi, Optimization of fuzzy logic scaling parameters with spiral dynamic algorithm in controlling a stair climbing wheelchair : Ascending task", 2014 19th

[18] Eckart Zitzler, Kalyanmoy Deb and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithm: Empirical Results. *Evolutionary Computation*, 8(2):173-195, Summer 2000.

[19] Kaveh A, Laknejadi K (2011) A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization. *Expert Syst Appl* 38(12):15475–15488

[20] Coello CAC (2004) Handling multiple objectives with particle swarm optimization. *IEEE T Evolut comput* 8(3):256–27.