An algorithm to solve multi-objective assignment problem by using discrete PSO decomposition

Waleed Mohammed Elaibi*
Department of Statistics
College of Administration and Economics
Mustansiriyah University
Iraq
waleed_m@uomustansiriyah.edu.iq

Faez Hassan Ali
Department of Mathematics
College of Science
Mustansiriyah University
Iraq
faezhassan@uomustansiriyah.edu.iq

Alaa Sabah Hameed
Ministry of Education
Baghdad
Iraq
alaasabah127@gmail.com

Hanan A. Cheachan
Department of Mathematics
College of Science
Mustansiriyah University
Iraq
hanomh@uomustansiriyah.edu.iq

Abstract. This paper presents an algorithm to solve a multi-objective assignment problem (MOAP) using discrete particle swarm optimization method under decomposition. The mathematical modeling of the algorithm has been derived to simulate the multi-objective assignment problem that considered as a non-linear function. The approach presented in this work highlights the optimal solution of every single objective function by decomposition the objective functions. To illustrate the algorithm a numerical examples are presented.

Keywords: assignment, multi-objective decision making, discrete PSO, decomposition.

1. Introduction

One of the significant problem in the mathematical problems is the assignment problem (AP). The classical type of the assignment problems (AP), in which the

*. Corresponding author

number of machines and jobs are equal, has studied in details and introduced to solve these problems. However, this problem move to the type of uncertain assignment due to the real life uncertainty. The goal of the AP is to assign a specific number of jobs to an equal and specific number of machines in order to minimize the total time required for execution all the jobs or the assignment total cost. The objectives in the multi-objective AP are discrete PSO decomposition. The classical type of the AP refers to a specific type of the linear programming problems (LP). The LP can be defined as one of the widely used tools for making decision for solving problems in real world. In the actual situation of the decision making problem, the critical concern is the decision problems include multiple criteria (objectives or attributes). However, several cases of the decision making of the real world come to happen in a specific environment where parameters, objectives, and constrains are not accurate Liu [1]. However, two issues are addressed through application of the PSO for solving the MOPs. The first one in the multi-objective PSO is the maintenance of archive for diversity and convergence balancing. The external elitist archive is applied for storing the non- dominated solution that obtained by using an algorithm and the others solutions obtained the filtered by a specific measurement of a quality, such as density.

Some MPSO proposals where the size of archive is unconstrained [1,3], the pre-fixed maximum size of an archive is commonly used because of the non-archive number. Furthermore, physical memory has a limited power. As a result, a suitable dominated solutions will evolve very rapidly, increasing the cost of computing of upgrading archiving. Filtering these (non-dominated) solutions with poorer quality steps is thus necessary. Several method for updating archive were introduced, for example, many MPSOs [4,17,20] use the crowding distance [11] to prune the archive, whereas Kernel density was first presented in [9] and utilized in MPSO [5]. For gbest updating in MPSO, a dynamic neighborhood approach is used and proposed in [21]. To pick pbest and gbest, a decomposition method was used29 on MPSOs [19]. To lead the search process, ranking approaches [12,16] are used to classify the best solutions. A good MPSO should be able to keep the diversity of the population. The external archive technique is commonly used for this reason in order to preserve the diversity of the received solutions. An external and alternative elitist archive holds the non-dominated alternatives discovered by an algorithm and updates the archive using a specific technique to ensure diversity. However, if a MOEA (e.g., MPSO) has trouble finding any Pareto optimal solutions, and these Pareto optimal solutions show an important role in preserving the diversity of archive, the archive's diversity cannot be sustained. To preserve variety, a decomposition technique has recently been used. MOEA/D (multi-objective evolutionary algorithm developed based on decomposition) is a representative algorithm that performs well when searching for a variety of non-dominated solutions for different types of MOPs [14]. MOEA/D breaks down the process of approximating the Pareto front (PF) into

a series of single-objective optimization of sub-problems by using conventional aggregation approaches.

MOEA/D substitutes an old solution with a new solution based on their values of the aggregation function if the aggregation function values are less than the former when evolving the solution package (i.e., the latter is closer to the optimal solutions than the former). In this paper we develop such an extension of the two phase method to solve multi- objective integer linear programmers

(MOIP) $Min\{z_1(x), \ldots, z_p(x) = Cx : x \in X\}$,

where $p \geq 2$ and $C \in R^{p*n}$. X denotes the set of feasible solutions of the problem and is defined by m, $X = \{x \in Zn : Ax \leq b; x \geq 0\}$, where $A \in Rm\_n$ and $b \in Rm$. We call Rn decision space($X \subset Rn$) and R^p objective space. $Y = \{Cx : X \in X\} \subset Rp$ is called the objective space feasible set or the outcome set.

Definition 1. A feasible solution $\overset{\frown}{x} \in X$ is efficient if no other feasible solution exists for $x \in X$ such that $z(x) \leq z(\overset{\frown}{x})/.Z(x)$ (is then called a non-dominated point. If $x; x' \in X$ are such that $z(x) \leq z(x')$ we say that x dominates x' and $z(x)$ dominates $z(x')$. If $z(x) = z(x'), x$ and x' are equivalent. Y_N is the non-dominated points set of all of Y and X_E is the efficient solutions set.

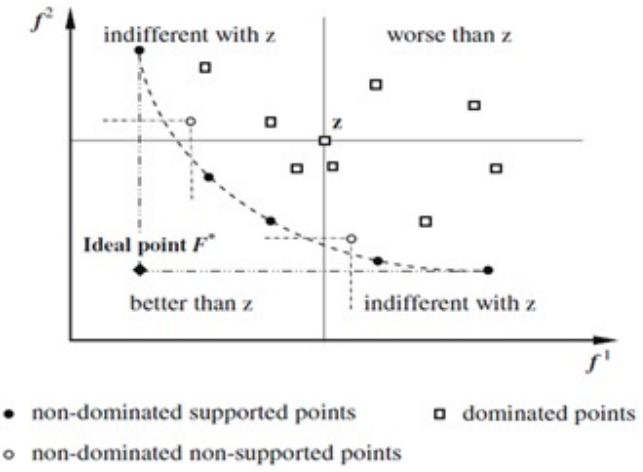

Fig. 1 Dominations in the Pareto sense in a bi-objective space

Definition 2 ([2])**.** A complete set defined by $X_E$is defined as a set of the efficient solutions at which all $x \in X \setminus X_E$ are either equivalent to or dominated by at least one $x' \in X_E$. In other words, There is at least one $x \in X_E$. for each non-dominated point $y \in Y_N$ such that $z(x) = y$.

In this work, the (AP) with the objectives is used to explain the developed methods and an algorithm to obtain the exact solution (finding a maximal or minimal complete set) of this problem is going to be proposed. In Pareto optimization (PO), the goal is to find a series of "efficient" solutions that are either

approximate or exact. Exact methods aim to solve a problem in the most efficient way possible, but their implementation on large real problems typically takes too long. Approximate approaches are used in functional applications to discover high-quality (but not necessarily optimal) alternatives in a short computational time. Approximate approaches are divided into two categories:

1. Heuristics: that are very specialized approximation approaches that take advantage of domain knowledge.

2. Meta-heuristics: Since they are not intended for a single problem, they have been effectively utilized to solve a broad variety of (combinatorial) optimization problems.

The multi-objective AP can then be solved in one of three ways: using exact methods where possible (see [3, 4], and [5]), approximating methods like those mentioned in [6,], or hybrid methods that incorporate the two preceding methods.

The below is the paper's structure: The mathematical modeling of the multi-objective AP is defined in section 2. The results and discussions are presented in Section 3. In sub-section 2.3, the optimization algorithm is presented. In sub-section 3.3, a numerical example is given to demonstrate the algorithm accuracy. Section 4 presents the conclusions and a few reflections of the suggested algorithm.

2. Mathematical model establishment of multi-objective AP

2.1 Multi-Objective Assignment Problem

This problem (MOP) is defined as follows.

$$Min(z) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^{k} x_{ij}, \quad k = 1, 2, 3, \ldots, n,$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad i = 1, 2, 3, \ldots, n,$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad j = 1, 2, 3, \ldots, n,$$

$$X_{ij} \geq 0, \quad \forall i, j,$$

where all coefficients of objective function c_{ij}^{k} are non-negative integers and $x = x_{11}, \ldots, x_{nn}$ are the decision variables matrix and equal 1 if the task i is assigned to agent j and 0 otherwise. The MOP can also be referred as a multi-objective perfect matching problem on a complete bipartite graph $G = (U \cup V; E)$ with $|U| = |V| = n$ so that any edge $i, j \in E$ links a vertex i in U to a vertex j in V with cost c_{ij}^{k} for any objective. A perfect matching in G is corresponding to a specific assignment in (MOAP). Let $X \subset \{0; 1\}n^2$ signify the feasible solutions set for (MOAP). To any solution $x \in X$ is associated a set $E(x)$ of n edges such that

an edge $i, j \in E$ of bipartite graph G is in $E(x)$ if and only if $xij = 1$. A solution is thus, consistently categorized by x or by $E(x)$. Finally, to any solution $x \in X$ is associated a criterion vector $z(x) = \{z_1(x), z_2(x), \ldots, z_n(x)\}$ which resembles to its image presented in the space of objective. The image set presented in the feasible solutions objective space is referred by $Z = \{z(x) : x \in X\}$.

2.2 Discrete particle swarm optimization

Position, fitness, and velocity values are all assigned to each particle. Particles benefit from historical knowledge gathered from the swarm community to travel across m-dimensional space. As a result, particles are more likely to fly into a better and stronger search field as evolution continues. Let NP stand for the swarm population size, which is given by $X_t = [X_{1t}, X_{2t}, \ldots, X_{NPt}]$. The following are the characteristics of each particle in the swarm population: The current situation was expressed by $X_i^t = [X_i^{1t}, X_i^{2t}, \ldots, X_i^{mt}]$, a current velocity represented as $V_i^t = [V_i^{1t}, V_i^{2t}, \ldots, V_i^{mt}]$ a current personal best position represented as $P_i^t = [P_i^{1t}, P_i^{2t}, \ldots, P_i^{mt}]$ and a current global best position characterized as $Gt = [g_{1t}, g_{2t}, \ldots, g_{mt}]$.

2.3 Decomposition property

Decomposition is a basic strategy in outdated multi-objective optimization. Though, it has not yet been commonly utilized in multi-objective evolutionary optimization. The decomposition idea has been used to a certain extent in several meta-heuristics for MOPs [7]–[11]. The traditional method of the mathematical programming techniques for solving MOPs used decomposition method extensively. Most MOEAs, on the other hand, regard a MOP as a whole and focus on dominance to assess solution consistency during their process of search. These algorithms could not be very successful at distributing solutions evenly around the PF.

There are some papers taking the composition approach as follow, Ref [13] took advantage of PSO as a quick, easy-to-implement, and a reliable method and specifies a smart methodology for particles updating while using decomposition to enhance and increase the solution diversity.

Since the original MOEA/D system was proposed, numerous works have been published in the literature aimed at: a) overcoming the shortcomings in components of design of the original MOEA/D, b) improving the efficiency of MOEA/D, c) presenting novel decomposition-based MOEAs, or d) adapting decomposition-based MOEAs for various types of problems. In 2016 the author [14] presents a broad survey of the decomposition- based multi-objective evolutionary algorithms anticipated in the last decade. The decomposition technique was not commonly used in the multi-objective optimization until Zhang and Li who presented the multi-objective evolutionary algorithm developed based on the principles of decomposition [15]. The proposed algorithm of Zhang and Li decompose the problem of the multi-objective optimization into a variety of

scalar sub- problems of optimization and optimizes them all at the same time using a specific evolutionary algorithm.

2.4 An introduction to PSO

In the optimization field, population-based stochastic method of the local search are a fairly recent paradigm. This family contains a variety of nature-inspired problem-solving methods that use metaphors as directions. The most well-known members are the genetic algorithms (GA), which seek solution spaces using the metaphor of genetic and evolutionary concepts of reproduction fitness selection for. Swarm Intelligence, which is described as "any attempt to design algorithms or distributed problem-solving devices inspired by the collective behavior of social insect colonies and other animal societies [19]", is motivated by the collective behavior of bird flocks, insect colonies, fish schools, and other societies of animals. The PSO approach, that used for solving optimization problems, is a relatively recent member of the Swarm Intelligence field. PSO is a recent evolutionary optimization strategy for optimizing the nonlinear continuous functions that was suggested by [20]. It takes its biological inspiration from the social contact metaphor and communication in a school of fish or flocks of birds. Among these communities, there is a leader that controls the whole swarm's activity. Every individual's movement is dependent on the leader and his own experience. Individuals (i.e. particles) in a PSO algorithm appear to obey the group leader, i.e. the one with the better results,since PSO is evolutionary in nature and population-based. In general, the model from which PSO is derived implies that each particle's action is a balance between its collective and individual memories-. The following are the rules that guide the PSO algorithm:

1. Any possible solution , a particle k in the swarm starts with a randomized position and velocity. The vectors denote the direction and velocity of particle k in the n- dimensional search space.$X_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ and $V_k = (V_{k1}, V_{k2}, \ldots, V_{kn})$, respectively, where $x_{kd}(d = 1, 2, ..., n)$ represents the location and $v_{kd}(d = 1, 2, ..., n)$ is the flying particle k velocity in the search space dth dimension.

2. Each particle k knows its position and the its objective function value. It also recalls the position at which it achieved its greatest achievement, which is $P_k^t = (P_{k1}^t, P_{k2}^t, ..., P_{kn}^t)$.

3. From every position, every particle can produce a neighborhood. It is therefore, a member of a certain neighborhood particles and recalls the particle (given by index g) which is in its best general position. The neighborhood will either be all the particles in case of the global neighborhood or a subset of particles in case of the local neighborhood.

The behavior of the particle is a compromise between the three possible options in each iteration t: following its exploration current pattern, returns to its best prior position, returns towards the all particles best historic value.

The following equations in the current iteration of the algorithm implement this compromise:

$$v_{kd}^{t+1} = wv_{kd}^{t} + c_1 r_1 (p_{kd}^{t} + x_{kd}^{t}) + c_2 r_2 (p_{gd}^{t} + x_{gd}^{t} \tag{1}$$

$$x_{kd}^{t+1} = x_{kd}^{t} + v_{kd}^{t+1} \tag{2}$$

where the symbol w is a constant value, called the inertia weight, chosen by the user for controlling the relationship between the previous velocities and the current one. The symbol c1 is the weight factor introduced to the attraction of the current particle's previous best position and the symbol c2 is the weight given to the previous best neighborhood particle location attraction. The symbols r1 and r2 in [0, 1] are random variables distributed uniformly.

2.5 The proposed algorithm (MOPSO/D) for assignment problems

In this section we are presented our new method in details as follows:

2.5.1 Multi-objective Particle Swarm Optimization (MOPSO)

In this section, the state-of-the-art PSO algorithm are examined. Two main categories of algorithms are distinguished, based on two methods: techniques that independently exploit each objective function, and schemes based on Pareto. The distinction is largely made for purposes of presentation and is not restrictive, because some algorithms combine characteristics from both methods. The exposition of the methods for each category is based on a chronological ordering.

The difficulty in determining an optimal solution of a multi-objective problem lies with the potential disagreements and contradict between the optimal solutions for the individual goals. The best solution for one purpose of a specific objective may be the worst. The resolution of these disagreements is essential to finding a multi-objective "optimal" solution. Two methods are available to resolve the multi-objective. First, the multi-objective problem would be converted into one objective problem (single problem). This was often done by weightily assembling all the objectives. However, with this technique, there is a problem of adjusting and changing the weighting factor. The second approach is mainly based on the Pareto optimality concept , which provides an ideal set of solutions rather than an optimum one. The reason for many solutions' optimalities is that with respect to all objective functions no one of them can be better than anyone else. PO is better suited to solving multi-objective problems compared to classical algorithms [12].

The (MOPSO/D) was proposed by Peng and Zhang[16]. This method utilizes the MOEA/D framework but substitutes the genetic operators (mutating and crossover) with the classical PSO inertia flight equations. MOPSO/D uses a turbulence operator (or mutation) and uses special archiving strategy (which is based on c-dominance [17]) for nondominated solutions storing initiated during

the search. In [18] they introduce a new decomposition- based multi-objective particle swarm optimizer (dMOPSO) for continuous and unconstrained MOPs. Further, the paper sought to resolve the multi-target optimum active power dispatch issue with PSO and MOPSO performances. In the IEEE Standard 30-bus system of 6 generators, the proposed technique was tested and validated. The aim is to address the multi-objective issue of the dispatch of economic power; four targets are fuel costs and effect on the environment due to emissions caused by SO2, CO2 and NOx.
MOPSO/D Procedure
Set $k := 0$ and velocity=0
Randomly initialize Point P_i for n. population;
Calculate the fitness values of initial Population: $f(P)$;
Personal best =POP;
$Zi = \min\{f(p) \setminus p \in POP(t)\}, 1 \leq i \leq m.$
Find the non-dominated solutions and initialized the archive with them (Decomposition and fitness calculation) First, by calculating the fitness value for each solution in POP(t) by crowding distance, split the POP(t) into N class solutions. Then pick some better POP(t) solutions and place them in a temporary pop population of N dimensions. Our work mostly uses the binary selection of tournament
WHILE (the termination conditions are not met)
1) PSO Steps
$p_{gd}^t = SelectLeader(POP)$
$v_{kd}^{t+1} = wv_{kd}^t + c_1r_1(p_{kd}^t + x_{kd}^t) + c_2r_2(p_{gd}^t + x_{gd}^t)$
$P_{kd}^{t+1} = P_{kd}^t + v_{kd}^{t+1}$
$P_{kd}^{new} = P_{kd}^{t+1} + 0.5 * (p_{gd}^t - p_{kd}^t)$
End
End
(Solution update) For each j = 1,..., m, if $\{Zj > (p) \setminus p \in 0\}$, then update $\{Zj > (p) \setminus p \in 0\}$. Firstly, categorize the 180 solutions of POP(k) $\cup$ O by, then pick N of the best solutions through the updated strategy of Section 3.4 and plug them into POP$(t + 1)$.
Update Global best
Update Personal best
Set $k := k + 1$;
END WHILE

3. Results and discussions

In this section, we are compares Multi-objective Partial Swarm Optimization with Decomposition (MOPSOD) practically with a stochastic meat-heuristic search algorithm,Non-Sorting Genetic Algorithm two (NSGAII), Non-Sorting Genetic Algorithm three (NSGAIII, (MOPSO), Strength Parito Evolutionary

Algorithm two (SPEA2), and (PESAII) to determine how MOPSOD is positioned against other heuristic search algorithms. In addition, through tables 1 and 2, we can say that the presented algorithm is highly efficient relative to other algorithms, and we can see this through the results that were shown or shown by statistical measures.

In this manner, standard benchmark test functions (i.e., Table 4.1) (Tarasewich and Mcmullen, 2002) presented in detail. Through the Table (4.2), i.e., we can see the performance of combine SOGS-BAT with other algorithms.

Table 1: Table (1): Comparative between algorithms by using Inverted Generational distance when (M=2)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	2	5	2.7279e-2 (0.00e+0) =	2.7279e-2 (0.00e+0) =	2.7279e-2 (0.00e+0) =	2.7279e-2 (0.00e+0) =	2.7279e-2 (0.00e+0)
	2	10	1.1224e-1 (9.01e-4) =	1.1205e-1 (4.25e-4) =	1.1216e-1 (4.08e-4) =	1.1235e-1 (5.86e-4) =	1.1202e-1 (8.18e-4)
	2	15	2.4625e-1 (1.07e-3) =	2.4583e-1 (3.07e-4) =	2.4577e-1 (3.09e-4) =	2.4721e-1 (1.08e-3) =	2.4554e-1 (1.18e-3)
	2	20	4.3854e-1 (1.43e-3) =	4.3860e-1 (8.38e-4) =	4.3850e-1 (2.10e-4) =	4.3838e-1 (1.57e-3) =	4.3683e-1 (1.55e-3)
	2	25	6.7901e-1 (1.90e-3) =	6.7874e-1 (4.62e-4) =	6.7815e-1 (1.18e-3) =	6.8099e-1 (1.29e-3) =	6.7802e-1 (6.82e-4)
MOAP	2	50	2.6702e+0 (1.61e-3) =	2.6692e+0 (3.09e-3) =	2.6709e+0 (5.16e-3) =	2.6742e+0 (1.61e-3) =	2.6703e+0 (1.48e-3)
	2	100	1.0620e+1 (3.63e-3) =	1.0620e+1 (2.40e-3) =	1.0621e+1 (1.58e-3) =	1.0625e+1 (9.49e-3) =	1.0616e+1 (2.80e-3)
	2	150	2.3881e+1 (4.80e-3) =	2.3893e+1 (1.17e-2) =	2.3887e+1 (5.83e-3) =	2.3901e+1 (1.28e-2) =	2.3886e+1 (3.97e-3)
	2	200	4.2363e+1 (6.19e-3) =	4.2369e+1 (1.97e-2) =	4.2378e+1 (1.39e-2) =	4.2405e+1 (1.91e-2) =	4.2363e+1 (1.06e-2)
	2	250	6.6130e+1 (1.18e-2) =	6.6122e+1 (1.23e-2) =	6.6138e+1 (4.00e-3) =	6.6131e+1 (2.26e-2) =	6.6117e+1 (4.80e-3)
	2	500	2.6455e+2 (1.26e-2) =	2.6451e+2 (4.14e-2) =	2.6459e+2 (3.28e-2) =	2.6461e+2 (3.95e-2) =	2.6454e+2 (1.88e-2)
	2	1000	1.0578e+3 (5.05e-2) =	1.0577e+3 (6.86e-2) =	1.0577e+3 (5.07e-2) =	1.0579e+3 (1.08e-1) =	1.0577e+3 (4.78e-2)
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 2: Table (2): Comparative between algorithms by using Inverted Generational distance when (M=3)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	3	5	3.3670e-2 (2.65e-4) =	3.4041e-2 (1.50e-4) =	3.3663e-2 (2.53e-4) =	3.3954e-2 (4.16e-4) =	3.3954e-2 (3.81e-4)
	3	10	1.2964e-1 (2.01e-4) =	1.3004e-1 (3.31e-4) =	1.2963e-1 (1.92e-4) =	1.2965e-1 (1.10e-3) =	1.2957e-1 (3.86e-4)
	3	15	2.9836e-1 (6.33e-4) =	2.9833e-1 (5.39e-4) =	2.9809e-1 (7.14e-4) =	2.9812e-1 (4.89e-4) =	2.9786e-1 (4.40e-4)
	3	20	5.2566e-1 (6.02e-4) =	5.2534e-1 (9.77e-4) =	5.2584e-1 (4.93e-4) =	5.2613e-1 (9.49e-4) =	5.2588e-1 (2.42e-4)
	3	25	8.1345e-1 (1.06e-3) =	8.1383e-1 (9.24e-4) =	8.1370e-1 (2.55e-4) =	8.1317e-1 (2.90e-4) =	8.1473e-1 (1.60e-3)
MOAP	3	50	3.2815e+0 (1.28e-3) =	3.2805e+0 (1.85e-3) =	3.2783e+0 (2.94e-3) =	3.2794e+0 (1.75e-3) =	3.2771e+0 (2.28e-3)
	3	100	1.2967e+1 (1.31e-3) =	1.2964e+1 (5.19e-3) =	1.2961e+1 (3.54e-3) =	1.2963e+1 (7.57e-3) =	1.2968e+1 (2.43e-3)
	3	150	2.9157e+1 (2.91e-3) =	2.9164e+1 (1.13e-2) =	2.9156e+1 (7.85e-3) =	2.9163e+1 (5.51e-3) =	2.9160e+1 (9.66e-3)
	3	200	5.1850e+1 (5.19e-3) =	5.1852e+1 (3.70e-3) =	5.1841e+1 (3.90e-3) =	5.1837e+1 (3.40e-3) =	5.1848e+1 (1.01e-2)
	3	250	8.1023e+1 (5.43e-3) =	8.1035e+1 (1.69e-2) =	8.1036e+1 (7.65e-3) =	8.1033e+1 (8.63e-3) =	8.1022e+1 (1.19e-2)
	3	500	3.2382e+2 (4.45e-2) =	3.2383e+2 (2.36e-2) =	3.2383e+2 (7.39e-3) =	3.2381e+2 (1.28e-2) =	3.2383e+2 (2.12e-2)
	3	1000	1.2954e+3 (1.23e-2) =	1.2954e+3 (4.17e-2) =	1.2954e+3 (2.37e-2) =	1.2954e+3 (1.31e-2) =	1.2954e+3 (2.75e-2)
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 3: Table (3): Comparative between algorithms by using Inverted Generational distance when (M=4)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	4	5	4.0560e-2 (3.80e-5) =	4.0625e-2 (7.56e-5) =	4.0544e-2 (4.84e-5) =	4.0583e-2 (5.24e-5)	4.0639e-2 (4.05e-5) =
	4	10	1.5867e-1 (1.87e-4) =	1.5883e-1 (2.28e-4) =	1.5869e-1 (1.43e-4) =	1.5874e-1 (2.24e-4)	1.5876e-1 (4.30e-4) =
	4	15	3.3823e-1 (7.17e-4) =	3.3827e-1 (7.90e-4) =	3.3829e-1 (6.88e-4) =	3.3854e-1 (2.67e-4)	3.3873e-1 (4.10e-4) =
	4	20	6.0579e-1 (1.47e-4) =	6.0506e-1 (1.01e-3) =	6.0519e-1 (3.28e-4) =	6.0564e-1 (1.57e-4)	6.0529e-1 (4.34e-4) =
	4	25	9.3145e-1 (9.35e-4) =	9.3236e-1 (1.23e-3) =	9.3069e-1 (1.27e-3) =	9.3207e-1 (1.26e-3)	9.3255e-1 (1.04e-3) =
MOAP	4	50	3.7235e+0 (1.41e-3) =	3.7213e+0 (1.72e-3) =	3.7242e+0 (9.34e-4) =	3.7240e+0 (3.21e-3)	3.7213e+0 (8.79e-4) =
	4	100	1.5054e+1 (3.90e-3) =	1.5053e+1 (5.78e-3) =	1.5052e+1 (4.99e-3) =	1.5052e+1 (2.31e-3)	1.5049e+1 (4.86e-3) =
	4	150	3.3670e+1 (3.20e-3) =	3.3670e+1 (4.04e-3) =	3.3667e+1 (7.48e-3) =	3.3665e+1 (5.07e-3)	3.3667e+1 (1.08e-2) =
	4	200	5.9860e+1 (9.82e-3) =	5.9863e+1 (1.74e-2) =	5.9864e+1 (8.16e-3) =	5.9865e+1 (2.08e-3)	5.9871e+1 (6.53e-3) =
	4	250	9.3560e+1 (7.52e-3) =	9.3562e+1 (1.53e-2) =	9.3558e+1 (9.84e-4) =	9.3557e+1 (1.94e-2)	9.3556e+1 (5.35e-3) =
	4	500	3.7398e+2 (3.44e-3) =	3.7397e+2 (1.49e-2) =	3.7398e+2 (1.97e-2) =	3.7398e+2 (1.49e-2)	3.7397e+2 (2.50e-2) =
	4	1000	1.4949e+3 (4.76e-2) =	1.4949e+3 (2.36e-2) =	1.4948e+3 (2.71e-2) =	1.4949e+3 (2.58e-2)	1.4948e+3 (7.21e-2) =
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 4: Table (4): Comparative between algorithms by using Inverted Generational distance when (M=5)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	5	5	4.3292e-2 (2.61e-4)	4.3199e-2 (4.92e-4) =	4.3210e-2 (3.35e-4) =	4.3242e-2 (3.87e-4) =	4.3541e-2 (1.05e-4) =
	5	10	1.7763e-1 (1.62e-4)	1.7790e-1 (5.55e-4) =	1.7784e-1 (1.39e-4) =	1.7774e-1 (3.83e-4) =	1.7740e-1 (2.69e-4) =
	5	15	3.8214e-1 (4.88e-4)	3.8245e-1 (3.41e-4) =	3.8196e-1 (2.80e-4) =	3.8166e-1 (6.22e-4) =	3.8201e-1 (5.16e-4) =
	5	20	6.7409e-1 (2.98e-4)	6.7531e-1 (9.57e-4) =	6.7460e-1 (1.21e-3) =	6.7356e-1 (5.86e-4) =	6.7435e-1 (6.04e-4) =
	5	25	1.0559e+0 (9.35e-4)	1.0567e+0 (1.18e-3) =	1.0558e+0 (6.62e-4) =	1.0558e+0 (5.97e-4) =	1.0548e+0 (4.12e-4) =
	5	50	4.1736e+0 (2.24e-4)	4.1759e+0 (9.58e-4) =	4.1750e+0 (1.62e-3) =	4.1743e+0 (6.88e-4) =	4.1727e+0 (2.68e-3) =
	5	100	1.6814e+1 (9.21e-4)	1.6815e+1 (1.57e-3) =	1.6813e+1 (4.65e-4) =	1.6812e+1 (2.68e-3) =	1.6808e+1 (3.70e-3) =
	5	150	3.7685e+1 (4.56e-3)	3.7691e+1 (4.36e-3) =	3.7683e+1 (6.86e-3) =	3.7689e+1 (2.83e-3) =	3.7692e+1 (2.17e-3) =
MOAP	5	200	6.6930e+1 (5.35e-3)	6.6930e+1 (5.38e-4) =	6.6927e+1 (2.82e-3) =	6.6931e+1 (1.09e-2) =	6.6924e+1 (5.36e-3) =
	5	250	1.0466e+2 (4.05e-3)	1.0465e+2 (8.90e-3) =	1.0466e+2 (1.24e-2) =	1.0465e+2 (6.49e-3) =	1.0466e+2 (1.38e-2) =
	5	500	4.1820e+2 (7.32e-3)	4.1819e+2 (1.56e-2) =	4.1820e+2 (1.13e-2) =	4.1820e+2 (3.05e-3) =	4.1819e+2 (7.92e-3) =
	5	1000	1.6714e+3 (4.46e-2)	1.6714e+3 (4.84e-2) =	1.6714e+3 (4.44e-2) =	1.6714e+3 (1.76e-2) =	1.6714e+3 (2.70e-2) =
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 5: Table (5): Comparative between algorithms by using Inverted Generational distance when (M=2)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	2	5	3.9214e-3 (1.85e-6) =	3.9159e-3 (1.92e-6) =	3.9140e-3 (6.13e-19) =	3.9140e-3 (0.00e+0) =	3.9140e-3 (0.00e+0) =
	2	10	3.4271e-2 (2.71e-2) =	2.6908e-2 (1.00e-2) =	2.1268e-2 (8.52e-3) =	1.5943e-2 (7.68e-5) =	1.5943e-2 (7.68e-5) =
	2	15	5.0951e-2 (1.38e-2) =	7.8397e-2 (1.10e-2) =	8.7615e-2 (4.99e-2) =	5.0517e-2 (5.67e-3) =	5.0517e-2 (5.67e-3) =
	2	20	1.7530e-1 (3.71e-2) =	1.3909e-1 (4.43e-2) =	1.9605e-1 (1.01e-1) =	7.2658e-2 (1.97e-3) =	7.2658e-2 (1.97e-3) =
	2	25	2.3433e-1 (2.69e-2) =	2.6261e-1 (3.90e-2) =	1.9509e-1 (1.72e-2) =	1.4350e-1 (2.34e-2) =	1.4350e-1 (2.34e-2) =
MOAP	2	50	1.0685e+0 (4.15e-1) =	8.8944e-1 (1.76e-1) =	9.6106e-1 (1.25e-1) =	8.0803e-1 (1.81e-1) =	8.0803e-1 (1.81e-1) =
	2	100	3.9906e+0 (7.75e-1) =	3.7113e+0 (9.68e-1) =	3.7715e+0 (2.37e-1) =	3.4943e+0 (2.28e-1) =	3.4943e+0 (2.28e-1) =
	2	150	7.1281e+0 (4.69e-1) =	1.0267e+1 (3.11e+0) =	9.1569e+0 (2.43e+0) =	6.7541e+0 (8.92e-1) =	6.7541e+0 (8.92e-1) =
	2	200	1.4412e+1 (4.95e-1) =	1.7113e+1 (3.66e+0) =	1.4070e+1 (1.59e+0) =	1.9078e+1 (9.55e+0) =	1.9078e+1 (9.55e+0) =
	2	250	2.5003e+1 (4.02e+0) =	2.6166e+1 (6.02e+0) =	2.5468e+1 (6.81e+0) =	1.9986e+1 (9.15e-1) =	1.9986e+1 (9.15e-1) =
	2	500	1.3446e+2 (1.73e+1) =	9.2404e+1 (2.41e+1) =	1.2981e+2 (1.99e+1) =	8.9370e+1 (1.66e+1) =	8.9370e+1 (1.66e+1) =
	2	1000	4.0703e+2 (5.72e+1) =	3.7493e+2 (5.65e+1) =	3.6326e+2 (6.39e+1) =	4.1923e+2 (6.12e+1) =	4.1923e+2 (6.12e+1) =
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 6: Table (6): Comparative between algorithms by using Inverted Generational distance when (M=3)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	3	5	8.7638e-3 (1.80e-3) =	9.5212e-3 (9.04e-4)	8.6174e-3 (1.51e-3) =	1.0016e-2 (1.74e-3) =	8.8364e-3 (1.60e-3) =
	3	10	3.8275e-2 (1.87e-3) =	3.7414e-2 (5.53e-3)	3.9525e-2 (2.90e-3) =	4.6431e-2 (1.67e-2) =	3.5299e-2 (3.66e-3) =
	3	15	7.9435e-2 (2.31e-2) =	6.7059e-2 (3.34e-3)	8.2517e-2 (1.36e-2) =	7.5517e-2 (7.25e-3) =	8.5704e-2 (1.69e-2) =
	3	20	1.4546e-1 (1.19e-2) =	1.3876e-1 (1.24e-2)	1.5256e-1 (6.46e-3) =	1.5567e-1 (2.76e-2) =	1.2827e-1 (1.81e-2) =
MOAP	3	25	2.2999e-1 (2.77e-2) =	2.1102e-1 (3.25e-2)	2.2502e-1 (1.82e-2) =	2.0215e-1 (9.33e-3) =	2.0230e-1 (7.35e-3) =
	3	50	9.8449e-1 (9.74e-2) =	8.3464e-1 (1.87e-1)	9.9841e-1 (1.86e-1) =	1.0157e+0 (1.58e-1) =	9.4627e-1 (1.75e-1) =
	3	100	4.8561e+0 (1.23e+0) =	4.0878e+0 (6.52e-1)	3.8836e+0 (6.12e-1) =	3.8586e+0 (6.27e-1) =	3.3895e+0 (6.80e-2) =
	3	150	6.9263e+0 (8.50e-1) =	7.7971e+0 (8.15e-1)	7.2409e+0 (1.23e+0) =	9.0822e+0 (1.11e+0) =	7.6053e+0 (1.30e+0) =
	3	200	1.4584e+1 (1.73e+0) =	1.6442e+1 (8.25e-1)	1.4504e+1 (3.55e+0) =	1.5546e+1 (1.54e+0) =	1.4802e+1 (7.23e-1) =
	3	250	2.2191e+1 (2.40e+0) =	2.1704e+1 (3.87e+0)	2.4310e+1 (2.29e+0) =	2.8338e+1 (9.28e+0) =	2.1335e+1 (2.82e+0) =
	3	500	9.3920e+1 (1.78e+1) =	8.9006e+1 (1.17e+1)	8.0480e+1 (5.38e+0) =	8.6549e+1 (9.03e+0) =	8.0611e+1 (6.14e+0) =
	3	1000	3.1176e+2 (1.35e+1) =	3.4100e+2 (4.50e+1)	3.2783e+2 (3.53e+1) =	3.6301e+2 (5.30e+1) =	3.4669e+2 (1.24e+1) =
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 7: Table (7): Comparative between algorithms by using Inverted Generational distance when (M=4)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	3	5	8.7638e-3 (1.80e-3) =	9.5212e-3 (9.04e-4)	8.6174e-3 (1.51e-3) =	1.0016e-2 (1.74e-3) =	8.8364e-3 (1.60e-3) =
	3	10	3.8275e-2 (1.87e-3) =	3.7414e-2 (5.53e-3)	3.9525e-2 (2.90e-3) =	4.6431e-2 (1.67e-2) =	3.5299e-2 (3.66e-3) =
	3	15	7.9435e-2 (2.31e-2) =	6.7059e-2 (3.34e-3)	8.2517e-2 (1.36e-2) =	7.5517e-2 (7.25e-3) =	8.5704e-2 (1.69e-2) =
	3	20	1.4546e-1 (1.19e-2) =	1.3876e-1 (1.24e-2)	1.5256e-1 (6.46e-3) =	1.5567e-1 (2.76e-2) =	1.2827e-1 (1.81e-2) =
MOAP	3	25	2.2999e-1 (2.77e-2) =	2.1102e-1 (3.25e-2)	2.2502e-1 (1.82e-2) =	2.0215e-1 (9.33e-3) =	2.0230e-1 (7.35e-3) =
	3	50	9.8449e-1 (9.74e-2) =	8.3464e-1 (1.87e-1)	9.9841e-1 (1.86e-1) =	1.0157e+0 (1.58e-1) =	9.4627e-1 (1.75e-1) =
	3	100	4.8561e+0 (1.23e+0) =	4.0878e+0 (6.52e-1)	3.8836e+0 (6.12e-1) =	3.8586e+0 (6.27e-1) =	3.3895e+0 (6.80e-2) =
	3	150	6.9263e+0 (8.50e-1) =	7.7971e+0 (8.15e-1)	7.2409e+0 (1.23e+0) =	9.0822e+0 (1.11e+0) =	7.6053e+0 (1.30e+0) =
	3	200	1.4584e+1 (1.73e+0) =	1.6442e+1 (8.25e-1)	1.4504e+1 (3.55e+0) =	1.5546e+1 (1.54e+0) =	1.4802e+1 (7.23e-1) =
	3	250	2.2191e+1 (2.40e+0) =	2.1704e+1 (3.87e+0)	2.4310e+1 (2.29e+0) =	2.8338e+1 (9.28e+0) =	2.1335e+1 (2.82e+0) =
	3	500	9.3920e+1 (1.78e+1) =	8.9006e+1 (1.17e+1)	8.0480e+1 (5.38e+0) =	8.6549e+1 (9.03e+0) =	8.0611e+1 (6.14e+0) =
	3	1000	3.1176e+2 (1.35e+1) =	3.4100e+2 (4.50e+1)	3.2783e+2 (3.53e+1) =	3.6301e+2 (5.30e+1) =	3.4669e+2 (1.24e+1) =
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

Table 8: Table (8): Comparative between algorithms by using Inverted Generational distance when (M=5)

Problem	M	D	NSGAII	NSGAIII	SPEA2	PESAII	MOPSODH
	5	5	6.2201e-3 (6.76e-6) =	7.0160e-3 (7.96e-4) =	6.2108e-3 (1.02e-5) =	6.2102e-3 (1.30e-5) =	6.5660e-3 (3.04e-4)
	5	10	2.7052e-2 (1.57e-3) =	2.9719e-2 (2.46e-4) =	2.7929e-2 (1.69e-3) =	2.7169e-2 (3.07e-3) =	3.0579e-2 (9.69e-4)
	5	15	6.2890e-2 (8.50e-3) =	5.9254e-2 (1.25e-3) =	5.7259e-2 (1.93e-3) =	5.7566e-2 (2.98e-3) =	6.6935e-2 (1.84e-3)
	5	20	1.0039e-1 (8.16e-3) =	1.1888e-1 (6.60e-3) =	1.1657e-1 (2.71e-2) =	1.0128e-1 (1.76e-3) =	1.0129e-1 (6.52e-3)
	5	25	1.6449e-1 (1.30e-2) =	1.7703e-1 (1.45e-2) =	1.6654e-1 (2.09e-2) =	1.6048e-1 (1.19e-2) =	1.9133e-1 (3.95e-2)
MOAP	5	50	6.4229e-1 (3.75e-2) =	7.9225e-1 (1.14e-1) =	6.8370e-1 (1.00e-1) =	7.2307e-1 (1.48e-1) =	6.8890e-1 (7.35e-2)
	5	100	2.8454e+0 (2.09e-1) =	2.7929e+0 (2.05e-1) =	2.3871e+0 (1.41e-2) =	2.5909e+0 (1.96e-1) =	3.0128e+0 (5.23e-1)
	5	150	6.0505e+0 (6.39e-1) =	5.8471e+0 (1.79e-1) =	6.0516e+0 (2.88e-1) =	5.8859e+0 (8.65e-1) =	5.9409e+0 (5.43e-1)
	5	200	1.0677e+1 (7.21e-1) =	1.1026e+1 (8.88e-1) =	1.0766e+1 (1.26e+0) =	1.4437e+1 (3.57e+0) =	1.0498e+1 (1.19e+0)
	5	250	1.9098e+1 (2.74e+0) =	1.8985e+1 (1.23e+0) =	1.7538e+1 (1.23e+0) =	1.7088e+1 (2.00e+0) =	1.7290e+1 (1.37e-1)
	5	500	6.6312e+1 (6.85e+0) =	6.9702e+1 (4.77e+0) =	6.9473e+1 (2.39e+0) =	6.7029e+1 (4.64e+0) =	6.4727e+1 (5.31e+0)
	5	1000	2.5046e+2 (1.83e+1) =	2.7881e+2 (7.76e+0) =	2.5666e+2 (1.37e+1) =	2.7149e+2 (2.08e+1) =	2.5650e+2 (2.66e+1)
+/-/=			0/0/12	0/0/12	0/0/12	0/0/12	

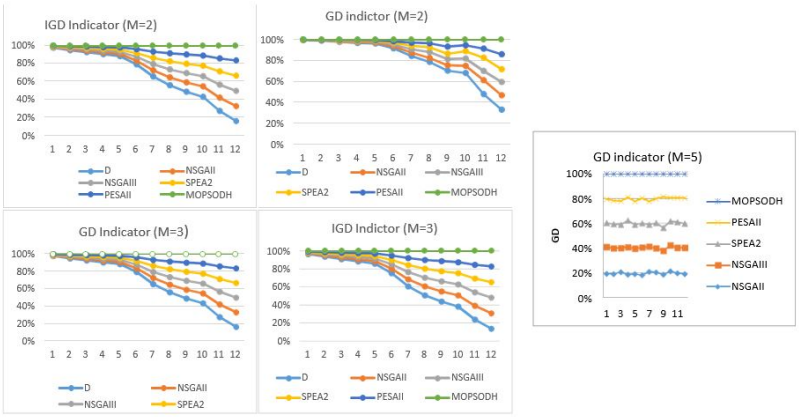

Figure 1 Shows the Priority of the Proposed Method (MOPSOD) with Others When M=2 and 3 |

4. Conclusion

This article suggests a decompose strategy-based MOPSO (MOPSO-D), in which MOPs are broken down into several subproblems of scalar optimization, and each one of them is optimized solely by using the information introduced from its several neighborhood subproblems in a single execution (run). Two performance metrics (γ and Δ), both show that MOPSO-D is highly competitive and even more effective than selected MOPSOs. The Pareto fronts also demonstrate that the MOPSO-D is able in comparison to the chosen MOPSOs to make comparatively better distributed Pareto fronts.

For further studies, it is attractive to extend the proposed algorithm and apply the technique to other evolutionary algorithms similar to Differential Evolution (DE). It is also important to expand our approach for resolution of additional performance criteria or even multi-objective parallel machines that plan problems. In addition, the presented approach can be applied for more complex machine environments to examine its validity and performance.

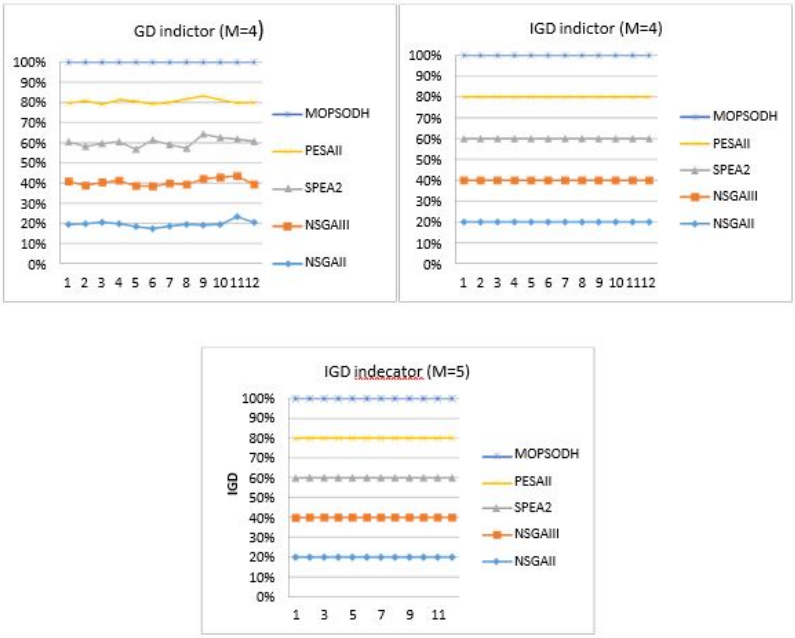

Figure 2 Shows the Priority of the Proposed Method (MOPSO/D) with Others When M=4and 5

References

[1] S.P. Gao, S.A. Liu, *Two Phase fuzzy algorithms for multi-objective transportation problem,* The Journal of Fuzzy Mathematics, 12 (2004), 147-155.

[2] P. Hansen, *Bicriterion path problems*, in: G. Fandel, T. Gal (Eds.), Multiple Criteria Decision Making Theory and Application, in: Lecture Notes in Economics and Mathematical Systems, vol. 177, Springer Verlag, Berlin, 1979, 109-127.

[3] H.L. Hatia, R. Malhotra, M.C. Puri, *Bi-criteria assignment problem,* Oper. Res., 19 (1982), 84–96.

[4] Hanan Ali Chachan, Faez Hassan Ali, *Using non-dominated sorting partical swarm optimization algorithm II for Bi-objective flow shop scheduling problem,* Iraqi Journal of Science, 62 (2021), 275-288.

[5] Hanan A. Chachan, *Strength pareto evolutionary algorithm II for multi-objective Knapsack problem,* International Journal of Psychosocial Rehabilitation Vol.24,Issue 01.2020,ISSN:1475-7192.

[6] Hanan A. Chachan, Manal H. Ibrhim, Hussam A.A. Mohammed, *Scheduling of single machine with release date to minimize multiple objective functions,* IOP Conf. Series, Materiasls Science and Engineering, 571 (2019).

[7] L. Paquete, T. Stützle, *A two-phase local search for the bi-objective traveling salesman problem,* in Proc. Evol. Multi-Criterion Optimization, 2003, 479–493.

[8] E. J. Hughes, *Multiple single objective pareto sampling,* in Proc. Congr. Evol. Comput., Canberra, Australia, 2003, 2678–2684.

[9] Y. Jin, T. Okabe, B. Sendhoff, *Adapting weighted aggregation for multiobjective evolutionary strategies,* in Evolutionary Multi-criterion Optimization, Springer, 2001, 1993, LNCS, 96–110.

[10] H. Ishibuchi, T. Murata, *Multi-objective genetic local search algorithm and its application to flow shop scheduling,* IEEE Trans. Syst., Man, Cybern., 28 (1998), 392-403.

[11] A. Jaszkiewicz, *On the performance of multiple-objective genetic local search on the 0/1 knapsack problem-A comparative experiment,* IEEE Trans. Evol. Comput., 6 (2002), 402–412.

[12] E. Mezura-Montes, J.I. Flores-Mendoza, *Improved particle swarm optimization in constrained numerical search spaces,* Springer Berlin Heidelberg, 193 (2009), 299–332.

[13] Al Moubayed, Noura, Andrei Petrovski, John McCall, *A novel smart multiobjective particle swarm optimisation using decomposition,* International Conference on Parallel Problem Solving from Nature, Springer, Berlin, Heidelberg, 2010.

[14] Trivedi, Anupam, et al., *A survey of multi-objective evolutionary algorithms based on decomposition,* IEEE Transactions on Evolutionary Computation, 21.3 (2016), 440-462.

[15] Q. Zhang, H. Li, *MOEA/D: A multi=objective evolutionary algorithm based on decomposition,* Evolutionary Computation, IEEE Transactions on, 11 (2007), 712-731.

[16] W. Peng, Q. Zhang. *A decomposition-based multi-objective particle swarm optimization algorithm for continuous optimization problems,* In IEEE International Conference on Granular Computing, 2008, 534-537.

[17] M. Laumanns, L. Thiele, E. Zitzler, K. Deb. *Archiving with Guaranteed convergence and diversity in multi-objective optimization,* In W. L. et al., editor, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002), 439–447, San Francisco, California, July 2002, Morgan Kaufmann Publishers.

[18] Zapotecas Martínez, Saúl, Carlos A. Coello Coello. *A multi-objective particle swarm optimizer based on decomposition,* Proceedings of the 13th annual conference on Genetic and evolutionary computation, ACM, 2011.

[19] E. Bonabeau, Dorigo, M., Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems,* Santa Fe Institute Studies in the Science of Complexity, New York, NY: Oxford University Press.

[20] J. Kennedy, R.C. Eberhard, *Particle swarm optimization,* In: Proceedings of IEEE International Conference on Neural Networks (1942–1948), Piscataway, NJ, USA, 1995.

[21] Ivan Belik, Kurt Jörnsten, *A new Semi-Lagrangean Relaxation for the k-cardinality assignment problem,* Journal of Information and Optimization Sciences, 37 (2016), 75-100.

[22] Kuo-Jen Hu, *Fuzzy goal programming technique for solving flexible assignment problem in PCB assembly line,* Journal of Information and Optimization Sciences, 38 (2017), 423-442.

Accepted: April 27, 2021