TELKOMNIKA, Vol.16, No.4, August 2018, pp. 1712~1723
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v16i4.7765

Optimizing Laying Hen Diet using Multi-swarm Particle Swarm Optimization

Gusti Ahmad Fanshuri Alfarisy*[1], Wayan Firdaus Mahmudy[2], Muhammad Halim Natsir[3]
[1,2]Faculty of Computer Science, Universitas Brawijaya, Malang, Indonesia
[3]Faculty of Animal Husbandry, Universitas Brawijaya, Malang, Indonesia
*Corresponding author, e-mail: gusti.alfarisy@gmail.com[1], wayanfm@ub.ac.id[2], emhanatsir@ub.ac.id[3]

Abstract

Formulating animal diet by accounting fluctuating cost, nutrient requirement, balanced amino acids, and maximum composition simultaneously is a difficult and complex task. Manual formulation and Linear Programming encounter difficulty to solve this problem. Furthermore, the complexity of laying hen diet problem is change through ingredient choices. Thus, an advanced technique to enhance formula quality is a vital necessity. This paper proposes the Multi-Swarm Particle Swarm Optimization (MSPSO) to enhance the diversity of particles and prevent premature convergence in PSO. MSPSO work cooperatively and competitively to optimize laying hen diet and produce improved and stable formula than Genetic Algorithm, Hybridization of Adaptive Genetic Algorithm and Simulated Annealing, and Standard Particle Swarm Optimization with less time complexity. In addition, swarm size, iteration, and inertia weight parameters are investigated and show that swarm size of 50 for each sub-swarm, total iteration of 16,000, and inertia weight of 6.0 should be used as a good parameter for MSPSO to optimize laying hen diet.

Keywords*: Feed formulation, PSO, Multi-swarm particle swarm optimization, Good parameters*

1. Introduction

The feed given to laying hen on daily basis is made of several feeds that are mixed through a formula that consists of composition of feed ingredients. The formula has to meet nutrient requirements of laying hen depending on the age. Furthermore, each feed ingredients have a maximum composition, different digestible amino acids related to the lysine nutrient, and fluctuating cost that increases the complexity of formulation.

Formulating appropriate composition of feeds using manual method such as Trial and Error, Pearson Square, and Algebraic method have many limitations. They are not accounting the cost of feed and the formula likely could not be found when accounting a lot of nutrients and feeds [1]. While in Nigeria, the feed takes approximately 65-80% of all cost production [2]. Linear Programming (LP) is the commonly employed in optimizing diet formulation [3–5] and successfully reducing the cost. However, as a constraint in requirement more restricts and no violation is allowed, LP could not provide the feasible formula. The LP then extended to Goal Programming (GP) to solve an issue of the infeasible formula and utilize multiple objectives like meal quality that could be obtained [6]. Furthermore, several methods have been proposed to formulate an optimum animal diet using mathematical approach [7], evolutionary, and hybridization approach [1].

The evolutionary approach shows promising results in feed formulation problem. The formula obtained from Genetic Algorithm (GA) has a lower cost than LP [8]. GA also employed for solving aquaculture diet [9] and extended by hybridizing roulette wheel and binary tournament selection in selection phase that produces more feasible formulae in shrimp diet than roulette wheel and queen bee selection [10]. The cost also reduced by enhancing the GA using adaptive approach on its parameter and by employing Simulated Annealing with GA [11]. Adaptive nature of Evolution Strategies (ES) with LP initialization can also be employed and show improved formula than conventional ES [12].

In addition, Swarm Intelligence approach like Particle Swarm Optimization (PSO) produce better formula than LP and GA in terms of time complexity and optimum solution [13].

Received November 9, 2017; Revised January 31, 2018; Accepted June 21, 2018

The process of PSO to find optimum formula does not require variation operators such as recombination, mutation, and selection phase which lead PSO to produce more rapid solution than GA. The movement process in PSO relies upon better solution found so far and the memory of best solution of all candidate solution using a simple equation which gives some advantageous. However, PSO easily trapped into local optima when problem dimension is high [14] and when dealing with the complex multi-modal problem [15]. Since the most used formula only estimating total amino acids in the feed. Whereas, the obtained formula often excess digestible amino acids of poultry [16]. The complexity of problem would increase by accounting digestible amino acids. Thus, the more advanced technique is required to enhance PSO to produce better and stable formula with less time complexity. Furthermore, there is a lack of study of PSO parameters in finding the optimum formula. It could be used for software developer as parameters references and improve PSO ability to reach global optima since PSO parameter is problem dependent [17].

The diversity of particle position should be expanded in order to prevent premature convergence. It can be obtained by using a multi-swarm approach which is a technique to use more than one swarm that corporate each other. This approach is proven as robust algorithm to solve a complex multi-modal problem and to prevent premature convergence [18]. The multi-swarm approach also has been applied to rainfall forecasting problem and show better results [19].Therefore, this study emphasizes on the improvement of PSO through the multi-swarm, take into account the balanced amino acids that considering the amount of digestible amino acids, and determination of good swarm size, iterations, and inertia weight of proposed method.

2. Standard Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an algorithm inspired from swarm behaviour of animals like birds and fish to find a food [20]. It has been applied in economic dispatch problem [21,22], software effort estimation [23], cost forecasting [24], and in designing artificial neural network [25]. The birds movement is influenced by other birds. In other words, the particle's position will change based on a velocity that depends on last velocity, cognitive movement, and social movement. Let say that a particle *i* represent the candidate solution that have D-dimensional problem and each dimension *j* have particular velocity and position at time *t* denoted as $v_{i,j}(t)$ and $x_{i,j}(t)$ respectively.

The cognitive movement is obtained from distance between current position and personal best position ($pbest_{i,j}$) accelerated by cognitive coefficient (c_1) and random real value between [0,1] (r_1). While the social movement is obtained from distance between current position and global best position ($gbest_j$) accelerated by social coefficient (c_2) and random real value between [0,1] (r_2). The inertia weight denoted as *w* is added later to control both movements [26].

$$v_{i,j}(t+1) = w \,.\, v_{i,j}(t) + c_1 \,.\, r_1 \left(pbest_{i,j}(t) - x_{i,j}(t)\right) + c_2 \,.\, r_2 \left(gbest_j(t) - x_{i,j}(t)\right) \quad (1)$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \quad (2)$$

The next velocity and position are updated by equation 1 and 2 respectively. As long the particle move, it has personal best position memory that updated by equation 3. While the best of all particle of all memory's movement is kept in *gbest* that updated by equation 4.

$$pbest_{i,j}(t+1) = \begin{cases} x_{i,j}(t+1), fitness\left(x_{i,j}(t+1)\right) > fitness\left(pbest_{i,j}(t)\right) \\ x_{i,j}(t), fitness\left(x_{i,j}(t+1)\right) \leq fitness\left(pbest_{i,j}(t)\right) \end{cases} \quad (3)$$

$$gbest_j(t+1) = \begin{cases} x_{i,j}(t+1), fitness\left(x_{i,j}(t+1)\right) > f\ itness\left(gbest_j(t)\right) \\ x_{i,j}(t), fitness\left(x_{i,j}(t+1)\right) \leq fitness\left(gbest_j(t)\right) \end{cases} \quad (4)$$

3. Proposed Multi-Swarm Particle Swarm Optimization

The proposed model of Multi-Swarm Particle Swarm Optimization (MSPSO) is depicted in Figure 1 where the square object represents the sub-swarm. Swarm in PSO is divided into 4 sub-swarm which are A, B, C, and D. The movement process of a particle of PSO in each sub-swarm is enhanced with a probability of bisection method using equation 5 and 6. Since particle only shares information outside of itself using global best position, it will be advantageous of a particle to get information from other better particle. Using equation 5, the particle can move based on another best personal position using bisection. *K* is a random integer number in [1, N], *N* is the swarm size of each sub-swarm. This movement is performed when the probability is satisfied called *Bp* that shown in equation 6. The value of *Bp* is linearly decreasing through time, *maxBp* is the maximum bisection probability, *minBp* is the minimum bisection probability, and *maxIteration* is the total iteration in migration phase. In MSPSO, each sub-swarm have different *maxBp* which are 0.2, 0.3, 0.4, and 0.5 with the same *minBp* of 0.01.

$$x_{i,j}(t+1) = \frac{x_{i,j}(t) + pbest_{k,j}}{2} \quad (5)$$

$$Bp = (maxBp - minBp)\, x\, \frac{maxIteration - t}{maxIteration} + minBp \quad (6)$$

In particular period, global best position in each sub-swarm is migrated to neighbor sub-swarm as shown in Figure 1. That movement process takes 80% of defined total iterations called as a migration phase. While the rest 20% of defined total iterations, all sub-swarm are aggregated into one swarm and all particle move using standard PSO movement process that uses the best global best position of all sub-swarm. This movement is called as an aggregation phase.

The proportion of iteration for both phases shows that migration phase takes longer run than aggregation phase. This intended to make particles focus on finding the good solution by cooperatively migrating the global best position and competitively using movement in its sub swarm with different maxBp parameters. Finally, in order to ensure the particles convergence and to work cooperatively for all particles, the aggregation phase is performed with 20% of total iterations.

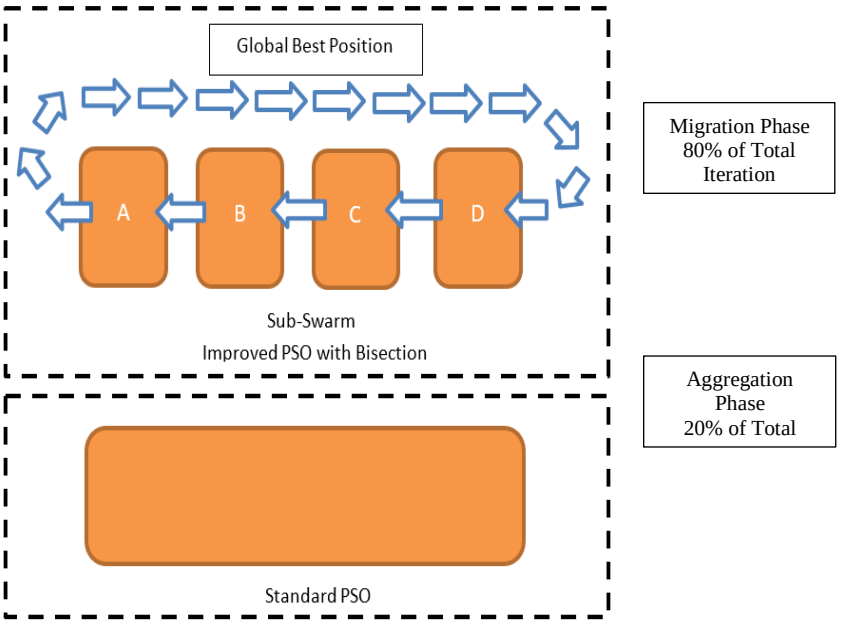

Figure 1. Proposed MSPSO Model to Optimize Laying Hen Diet

4. MSPSO Application to Optimize Laying Hen Diet

The laying hen diet formula consists of a composition of feed ingredients in percentage form. The amount of feed intake on daily basis easily obtained by converting the formula to weight form. All composition have to satisfy hard constraint and soft constraint. The hard constraint is the sum of all composition that have to be 100%. While the soft constraints are non-amino acids constraint, balanced amino acids constraint, and composition constraint.

4.1. Total Ingredient Constraint

The total percentage of each feed ingredient has to be 100% that serve as the hard constraint. Thus, the amount of feed in percentage form have to satisfy the hard constraint and no penalty is given. Let $\bar{X}_i$ is the vector position of a particle i that contains each percentage of feed ingredient that is described in equation 7 and the hard constraint is described in equation 8. $x_{i,j}$ is the percentage value of particle i at feed ingredient j and D is the total dimension of the problem which is the total feed ingredients.

$$\bar{X}_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,j}, \dots, x_{i,D}\} \quad (7)$$

$$\sum_{j=1}^{D} x_{i,j} = 100 \quad (8)$$

During movement process, the total amount of ingredients likely not equal to 100% precisely. Thus, the repair process is a necessity for a particle to satisfy the hard constraint. The amount of ingredient composition is repaired using equation 9. However, the equation can be used only when all composition has a positive value. During the initial movement, it is highly likely that the amount of ingredient is negative which equation 9 incapable to be employed and a negative value is given to total negative amount found in a particle. The assessment of a particle was discussed in detail in chapter 4.4.

$$x_{i,j} = \frac{x_{i,j}}{\sum_{j=1}^{D} x_{i,j}} \; x \; 100\% \quad (9)$$

4.1. Non-Amino Acids Constraint

The nutrient requirement of laying hen change through age phases such as starter, grower, or layer. It has minimum and maximum boundary that should be satisfied by each nutrient in feed. The amount of non-amino acids nutrient is sufficient if it is in-between the boundary. Accounted non-amino acids in this study are Metabolizable Energy (ME), Crude Protein (CP), Crude Fat (F), Crude Fiber (CF), Calcium (Ca), Phosphorus (P), Sodium (Na), Potassium (K), Chlorine (Cl), Manganese (Mn), and Zinc (Zn).

Let b be the particular nutrient and total amount of b denoted as tN_b that is obtained by using equation 10. $x_{i,j,b}$ is the amount of b in ingredient j at particle i in one Kilogram. The constraint then defined in equation 11. min and max in b is the minimum and maximum boundary.

$$tN_b(\bar{X}) = \sum_{j=1}^{D} \frac{x_{i,j}}{100} \times x_{i,j,b} \quad (10)$$

$$b_{min} \leq tN_b(\bar{X}) \leq b_{max} \quad (11)$$

4.2. Amino Acids Constraint

An amount of digestible Amino Acids (AA) nutrients depends on the amount of Lysine (*Lis*) in a feed. Even though the amino acids is excessive in a feed, the laying hen only takes the digestible amount of it. Thus, the extension of equation 10 is necessary to assess the digestible AA in a feed. The accounted AA in this study are Arginine (Arg), Cysteine (Cys), Glycine (Gly), Histidine (His), Isoleucine (Isol), Leucine (Leu), Lysine (Lis), Methionine (Met), Phenylalanine (Fenil), Threonine (Thre), Tryptophan (Trip), Tyrosine (Tir), and Valine (Val).

The amount of particular AA in a feed is relative to the fulfillment of *Lis*. If the amount of *Lis* in a feed is less than the minimum requirement, then the particular digestible AA is calculated using equation 12. The minimum requirement of particular nutrient *c* and *Lis* denoted as c_{min} and Lis_{min} respectively. Otherwise, if the amount of *Lis* in a feed is satisfied or

excessive, then the amount of other AA is the same with total nutrient in a feed (tN_b) regardless the satisfaction of the requirement of AA. Therefore, The amount of digestible AA under certain condition is defined in equation 13 and the constraint is defined in equation 14.

$$D_c(\bar{X}) = \frac{c_{min}}{Lis_{min}} \times tN_{Lys}(\bar{X}) \tag{12}$$

$$dN_c(\bar{X}) = \begin{cases} D_c(\bar{X}), & tN_{Lys}(\bar{X}) < Lys_{min} \wedge tN_c(\bar{X}) \geq D_c(\bar{X}) \\ tN_c(\bar{X}), & tN_{Lys}(\bar{X}) \geq Lys_{min} \\ tN_c(\bar{X}), & tN_{Lys}(\bar{X}) < Lys_{min} \wedge tN_c(\bar{X}) < D_c(\bar{X}) \end{cases} \tag{13}$$

$$C_{min} \leq dN_c(\bar{X}) \tag{14}$$

4.3. Maximum Ingredient Constraint

Each position *j* in particle *i* that denoted as $x_{i,j}$ have maximum composition boundary that is recommended by the experts. If $x_{i,j,max}$ shows the maximum limit of ingredient *j*, then the maximum ingredient constraint that should be satisfied is defined in equation 15.

$$0 \leq x_{i,j} \leq x_{i,j,max} \tag{15}$$

4.4. Fitness Function

The fitness function is a function to evaluate a particle with certain assessment or objective. The higher the fitness value, the higher the quality of candidate solution. The fitness value can be obtained from the inverse of the summation of the total cost, a penalty of the nutrient or non-amino acids, a penalty of amino acids, and penalty of a maximum ingredient which respectively denoted as $tCost$, pN, pAA, and pMI. Furthermore, the negative position in a particle highly likely to be found in the first movement. If the negative position is repaired immediately in the same iteration, the valuable information would be a loss for another particle. In this study, the total negative position becomes the fitness value of a particle which denoted as tNP. Therefore, the fitness function is defined in equation 16 and the total negative position is defined in equation 17.

$$F(\bar{X}) = \begin{cases} \frac{1}{nCost(\bar{X})+pN(\bar{X})+pAA(\bar{X})+pMI(\bar{X})}, & \forall x \in \bar{X}: x \geq 0 \\ tNP(\bar{X}), & \exists x \in \bar{X}: x < 0 \end{cases} \tag{16}$$

$$tNP(\bar{X}) = \sum_{j=1}^{D} \begin{cases} x_{i,j}, & x_{i,j} < 0 \\ 0, & x_{i,j} \geq 0 \end{cases} \tag{17}$$

When minimizing cost as well as another objective like nutrient and maximum ingredient penalty, the number should be not too far from other. For instance, In 100 Kg. The cost of feed could be IDR 400,000 or more. This number is too far from the penalty value. Thus, the normalized cost is performed to total cost in respect to all objectives as shown in Equation (18) and the total cost is shown in equation 19. The minimum cost from all feed ingridients in one kilogram is denoted as *minCost* while the maximum cost is denoted as *maxCost*. For instance, ingredient A=IDR 5,000, ingredient B=IDR 2,000, and ingredient C=IDR 3,000. Then the *minCost* would be 2,000 while *maxCost* would be 5,000.

$$nCost(\bar{X}) = \frac{tCost(\bar{X}) - 100\ .minCost}{100\ .maxCost - 100\ .minCost} \tag{18}$$

$$tCost(\bar{X}) = \sum_{j=1}^{D} x_{i,j}\ x\ c_j \tag{19}$$

The penalty of non-amino acids is given when the amount of particular nutrient *b* violates the constraints. Let *B* is the number of non-amino acids and $b \subset B$. For each accounted *b*, the penalty is summed which is shown in equation 20. b_{min} and b_{max} are a minimum and a maximum requirement of nutrient *b* respectively.

$$pN(\bar{X}) = \sum_{b\in B} \begin{cases} tN_b(\bar{X}) - b_{max}, & b_{max} < tN_b(\bar{X}) \\ b_{min} - tN_b(\bar{X}), & b_{min} > tN_b(\bar{X}) \\ 0, & b_{min} \leq tN_b(\bar{X}) \leq b_{max} \end{cases} \quad (20)$$

While the penalty of amino acids is defined in Equation (21). Let *C* be the number of amino acids, *c* is the particular nutrient of amino acids, and $c \subset C$. c_{min} and c_{max} are a minimum and a maximum requirement of nutrient *c* respectively.

$$pAA(\bar{X}) = \sum_{c\in C} \begin{cases} c_{min} - dN_c(\bar{X}), & dN_c(\bar{X}) < c_{min} \\ 0, & dN_c(\bar{X}) \geq c_{min} \end{cases} \quad (21)$$

The penalty from maximum ingredient constraint is obtained from Equation (22). The maximum limit of feed *j* is denoted as $x_{i,j,max}$.

$$pMI(\bar{X}) = \sum_{j=1}^{D} \begin{cases} x_{i,j} - x_{i,j,max}, & x_{i,j,max} < x_{i,j} \\ 0, & x_{i,j,max} \geq x_{i,j} \end{cases} \quad (22)$$

5. Results and Discussion

Analysis of good parameters is performed first before comparison to other approach such as Genetic Algorithm, Standard PSO, and Hybridization Adaptive GA with Simulated Annealing. In the analysis of swarm size and total iteration, w=0.6, c1=1.8, and c2=2.1 is used as initial parameter. The test formulae for experimentation of swarm size, iterations, and inertia weight is defined in Table 1. While the test formulae for comparison to other algorithms is defined in Table 2 (Please visit http://blog.ub.ac.id/alfarisy/2018/01/31/ingredients-dataset-of-poultry-diet-optimizing-laying-hen-diet-using-multi-swarm-particle-swarm-optimization/ for more detail).

Table 1. Test Formulae for Good Parameters

Formula	Feed Ingredient
11A	Bran, Corn Bran, Wheat, Yellow Corn, Menir , Soybean Meal, Coconut Meal, Peanut Meal, Foka, MBM, Bone Flour
12A	Bran, Wheat, Menir, Sorghum, Coconut Meal, Hidrolisis I. Rumen, MBM, Quill Flour, Meat Flour, Blood Flour, Bone Flour, Fish Oil
13A	Corn Bran, Wheat, Yellow Corn, Menir, Pollard, Rubber Seed Meal, Fish Flour (Herring), Fish Flour (Menhaden), Meat Flour, Clamshell, Bone Flour, Fish Oil, Coconut Oil
14A	Bran, Corn Bran, Wheat, Yellow Corn, Menir, Pollard, Coconut Meal, Skim Milk, Fish Flour (Ancovetta), Fish Flour (Menhaden), Meat Flour, Chalk, Clamshell, Bone Flour
15A	Bran, Corn Bran, Wheat, Yellow Corn, Cotton Seed Meal, Rubber Seed Meal, Soybean Meal, Coconut Meal, Snail Flour, Clamshell, Blood Flour, Lamtoro Flour, Chalk, Bone Flour, Fish Oil

Table 2. Test Formulae for Comparison

Formula	Feed Ingredient
11B	Corn Bran, Wheat, Yellow Corn, Menir, Pollard,9, Coconut Meal, Fish Flour (Ancovetta), Fish Flour (Herring), Fish Flour (Menhaden), Bone Flour
12B	Bran, Corn Bran, Yellow Corn, Sorghum, Coconut Meal, Hidrolisis I. Rumen, MBM, Fish Flour (Ancovetta), Quill Flour, Blood Flour, Bone Flour, Fish Oil
13B	Bran, Corn Bran, Wheat, Yellow Corn, Pollard,8, Coconut Meal, MBM, Fish Flour (Herring), Lamtoro Flour, Clamshell, Fish Oil, Coconut Oil
14B	Bran, Corn Bran, Wheat, Sorghum, Cotton Seed Meal, Coconut Meal, Peanut Meal, MBM, Fish Flour (Ancovetta), Meat Flour, Blood Flour, Chalk, Clamshell, Coconut Oil
15B	Bran, Corn Bran, Wheat, Pollard, Sorghum, Cotton Seed Meal, Soybean Meal, Coconut Meal, Hidrolisis I. Rumen, MBM, Quill Flour, Lamtoro Flour, Clamshell, Bone Flour, Fish Oil

5.1. Swarm Size

The swarm size significantly affects the performance of PSO. A large number of swarm size may increase time complexity of PSO while a small number of swarm size snares a particle into local optima region [21]. An effect of the increment of swarm size is carried out through simulation by gradually increase swarm size from 5 to 100 by 5 step and the obtained result is shown in Figure 2.

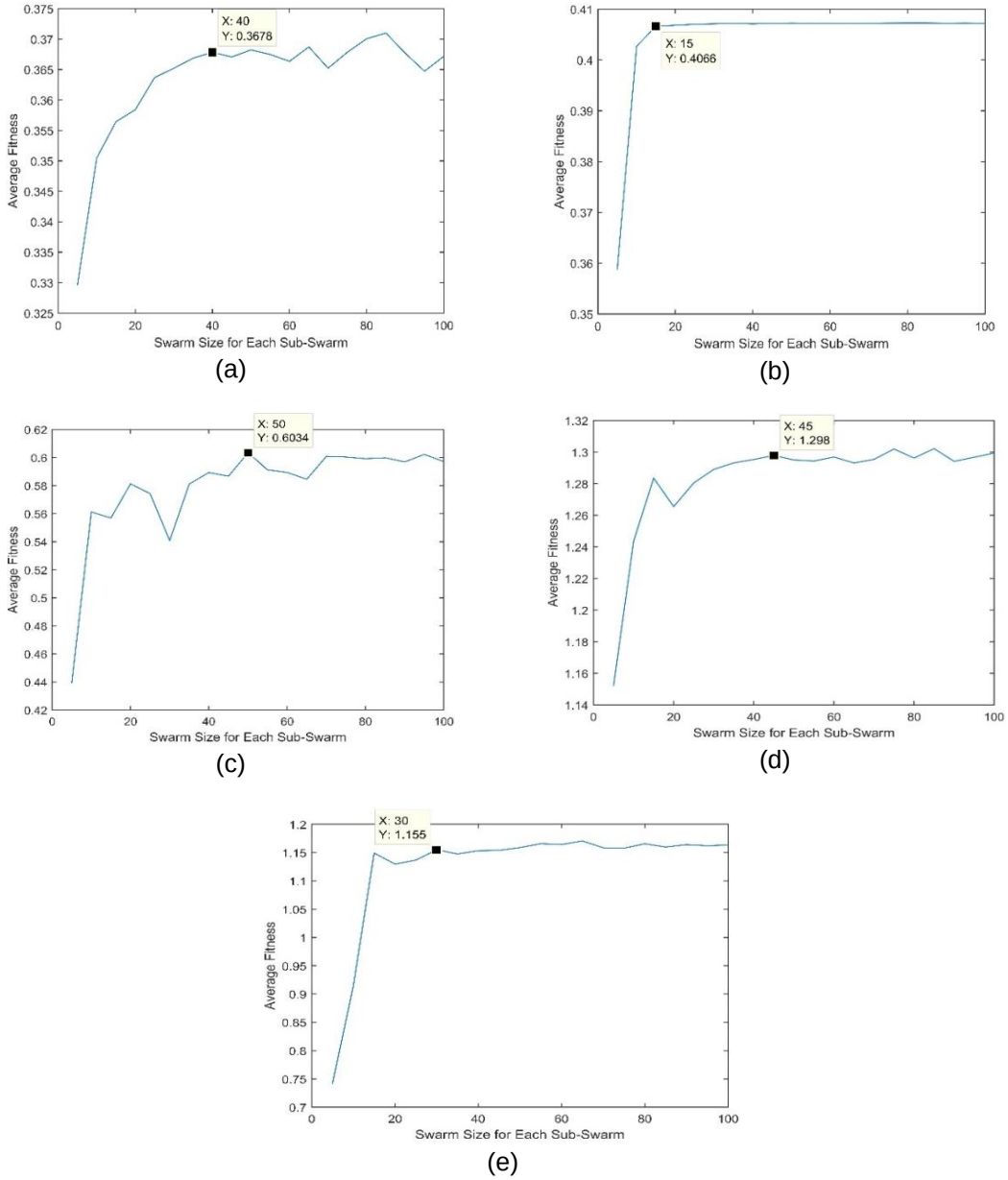

Figure 2. The effect of swarm size for each sub-swarm in formula 11A (a), 12A (b), 13A (c), 14A (d), and 15A (e)

The optimum swarm size for each sub-swarm is determined by pick out the number that give little bit significant improvement less than it and does not give any siginificant improvement higher than it. In formula 11A, 40 swarm size is the optimum value as shown in Figure 2. if we increase the swarm size above it, it does not give any serious impact to the average fitness. While for other formula, 12A = 15, 13A = 50, 14A = 45, and 15A = 30 swarm size.

The simulation results show us that different choice of ingredients need different optimum swarm size. It is a lot of task to determine each optimum swarm size for each ingredients combination. Futhermore, the number of problem dimension is not associate with the number of the swarm size. 15A which is 15 different ingredient choices need less swarm size than 13A which is 13 different ingredient choices. Therefore, swarm size should be determined for all possible choices from 5 simulation results. Eventhough it is not optimum for every formula.

In determining good swarm size, the determined value should not aggravate the performance for others. For instance, if we choose 15 as a good swarm size, then it is not the optimum value for 11A which is shown lower average fitness. While, if we choose the highest optimum swarm size like 50, all formulae would produce high average fitness even it is not optimum in terms of time complexity. Since it is a lot of task to analyze every ingredient choices and considering fluctuating price, the good value is determined by the small sample of simulation. Therefore, the good choice of swarm size should be above 50.

5.2. Iterations

The results of optimal iterations for all formula is depicted in Figure 3. We tune the number of iterations from 1,000 until 20,000 and if the optimal iteration does not seem obvious to be determined as an optimal value then we increase the iteration to 30,000 by 1,000 steps. PSO is run 10 times and average fitness is calculated due to stochastic optimization and for fair analysis. The optimum iteration for each formula is drawn from the average fitness for each iteration.

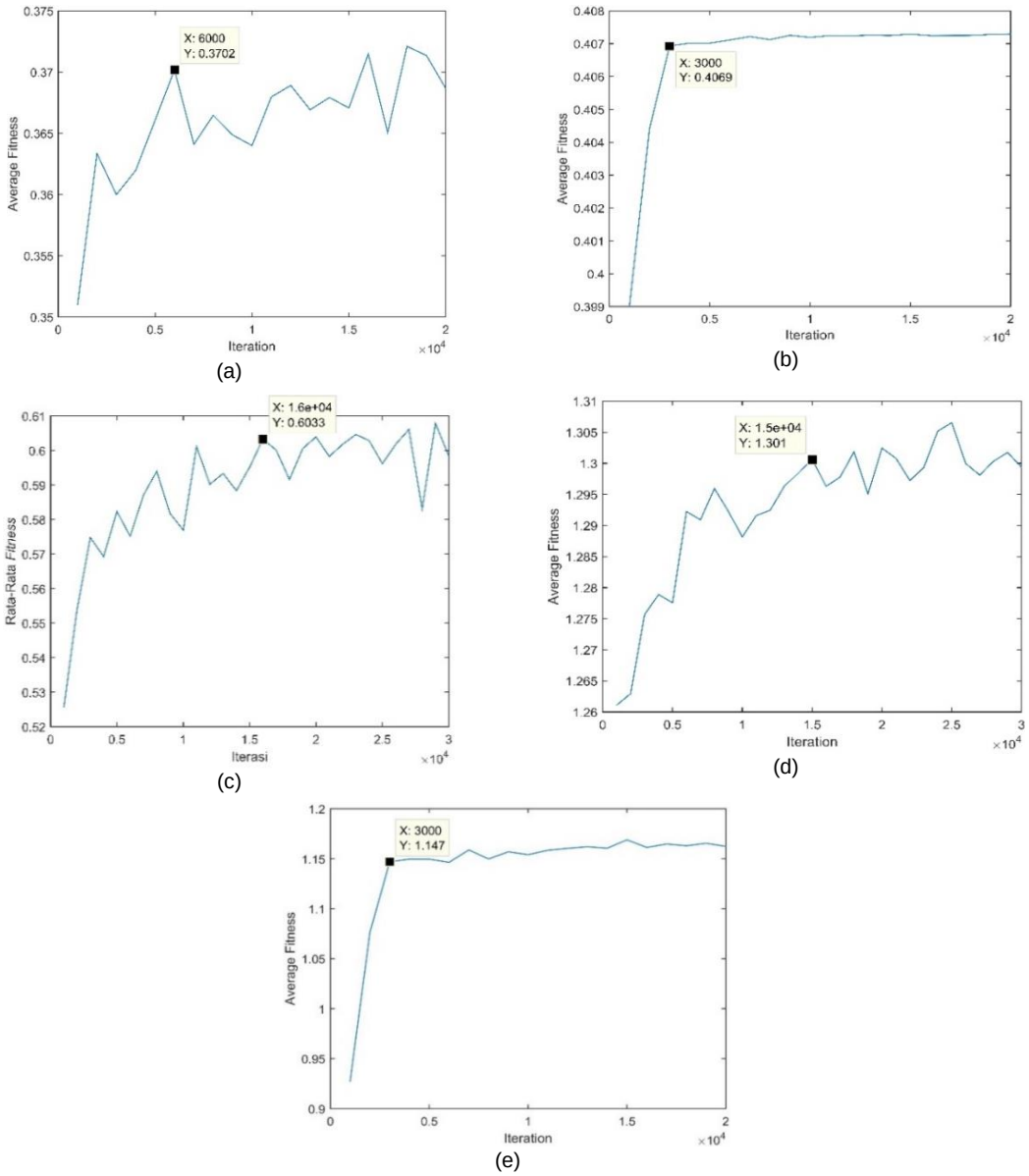

Figure 3. The effect of iterations in formula 11A (a), 12A (b), 13A (c), 14A (d), and 15A (e)

The optimum iteration is determined by observing the impact to average fitness and PSO duration. In formula 11A, 6,000 iterations are adequate to produce good solution since significant improvement is not found above 6,000 as well as other formulae. In formula 12A, require the minimum iteration of 3,000. In formula 13A, require the minimum iteration of 16,000. In formula 14A, require the minimum iteration of 15,000. While in formula 15A, require the minimum iteration of 3,000. The all optimum iteration differs through the formula. In this problem, no association between a number of dimensions and optimum iteration.

As shown in Figure 3, There are no optimum iterations for all ingredients choices. It becomes difficult task to find all optimum iteration for all ingredient combinations. We should determine the good iterations that notice whether it decrease the average fitness in another formula or not. In this case, the highest optimum iteration is safe to choose since it does not sacrifice the average fitness even though it may increase time complexity for others. Thus, the number of iteration above 16,000 should be chosen as good iteration.

5.3. Inertia Weight

The results of good inertia weight for all formula is depicted in Figure 4. We tune the inertia weight from 0,1 and gradually increase the value by 0,1 until 1,0. For each inertia weight, PSO run 10 times and average fitness is calculated. The good inertia weight for each formula is drawn from the obtained result.

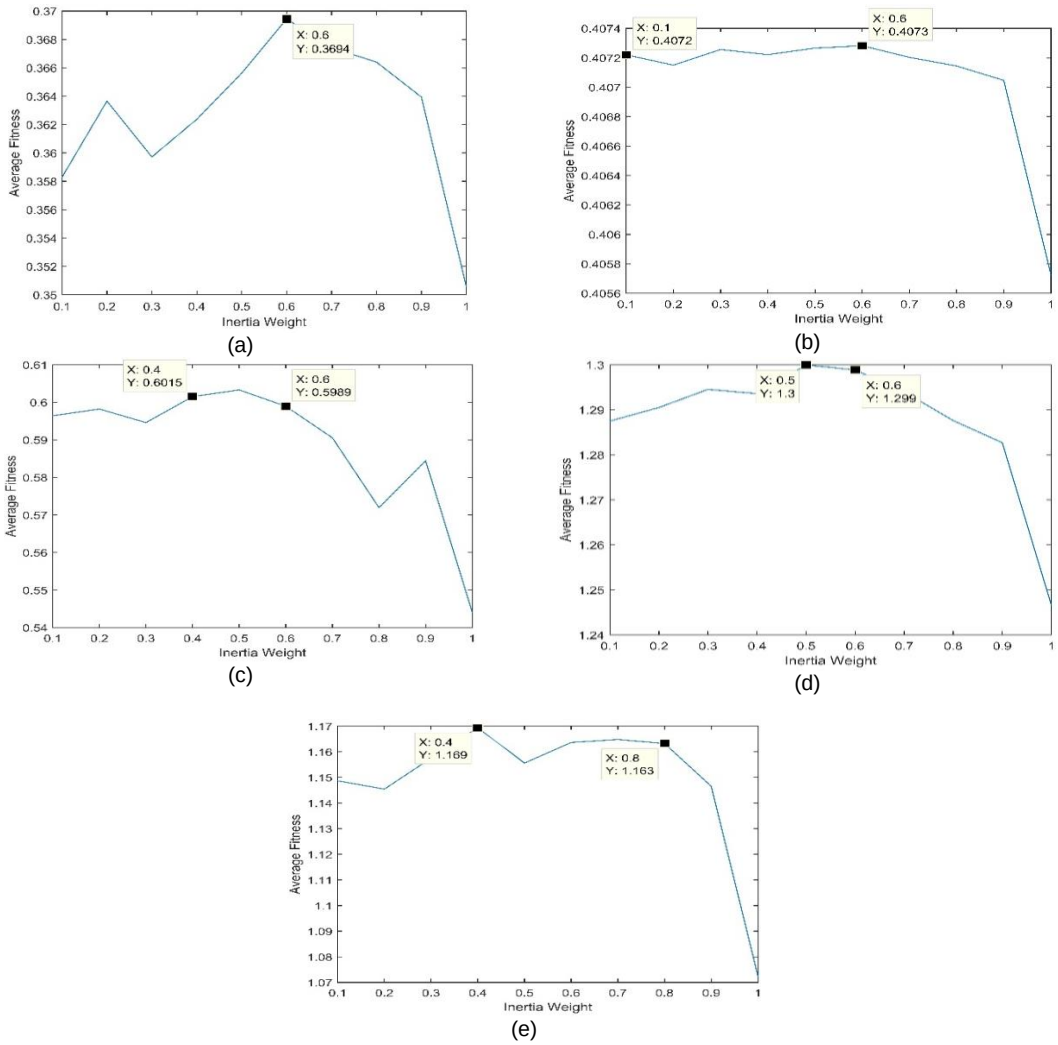

Figure 4. The effect of inertia weight in formula 11A (a), 12A (b), 13A (c), 14A (d), and 15A (e)

As depicted in Figure 4, The high value of inertia weight, particularly inertia weight of 1.0, would reduce the average fitness on all formula. In formula 11A, increasing inertia weight from 0.1 to 0.6 would increase average fitness gradually and above 0.6 would reduce the average fitness. In this case, 0.6 should be the good inertia weight for 11A. While in formula 12A, the inertia weight from 0.1 to 0.6 does not give any significant improvement and inertia weight above 0.6 would reduce the average fitness. Thus, in formula 12A, the inertia weight between 0.1 to 0.6 should be a good inertia weight to choose.

The determination of the good inertia weight for other formulae also use the same observation. In formula 13A, the inertia weight between 0.4 and 0.6 can produce higher average fitness than below 0.4 and above 0.6. In formula 14A, the good inertia weight should be between 0.5 and 0.6. While in formula 15A, the good inertia weight should be between 0.4 and 0.6. The good inertia weight that can produce a better solution for all formula is about 0.6 as depicted in Figure 4. Inertia weight of 0.6 is suitable in formula A11, A12, A13, A14, and A15. Therefore, 0.6 is chosen as a good inertia weight that is used in next experimentation.

5.4. Comparison

MSPSO is tested with other algorithms in order to perceive how robust of this proposed algorithm. The comparison algorithms are Genetic Algorithm (GA), Hybridization of Adaptive Genetic Algorithm and Simulated Annealing (AGASA), and standard Particle Swarm Optimization (PSO).

The obtained results of average fitness, average penalty, average cost, average time, and average standard deviation for all algorithms and formulae are shown in Table 3. All algorithms use the same swarm size/population size and total iteration/generation. They were run ten times and the average of the results was calculated. In GA, the crossover rate of 0.7 and mutation rate of 0.3 are used in this experimentation. While in AGASA, crossover rate of 0.6, mutation rate of 0.4, SA iteration of 10, and temperature decrease rate of 0.75 are used as shown as a good parameter in AGASA to optimize poultry diet [11].

Table 3. Comparison Results of GA, AGASA, PSO, and MSPSO for all formulae

Formula	Algorithm	Average Fitness	Average Total Penalty of Nutrient	Average cost (IDR/100Kg.)	Average Time (seconds)	Average Standard Deviation
B11	GA	0.778741	1.000656589	**740,567.87**	140.2455	0.049634
	AGASA	0.794305	0.969401818	745,331.83	478.4577	0.029883
	PSO	0.86739	0.83925435	790,760.30	61.04272	0.04715
	MSPSO	**0.922858**	**0.770553304**	784,254.23	**54.14518**	**0.009419**
B12	GA	0.131164	4.837896555	**474,864.53**	168.7403	0.013303
	AGASA	0.16641	0.987496691	818,293.01	521.4085	0.009502
	PSO	0.176263	2.416125052	885,897.58	**53.68276**	0.006955
	MSPSO	**0.183378**	**0.578135752**	1,028,150.25	56.72956	**0.000827**
B13	GA	0.985869	0.987496691	818,293.01	164.1121	0.116667
	AGASA	1.289382	0.734292391	885,897.58	318.4472	0.032245
	PSO	1.378856	0.677241754	1,028,150.25	78.95179	0.038345
	MSPSO	**1.415733**	**0.654207681**	**789,554.65**	**58.522**	**0.019207**
B14	GA	0.355566	2.416125052	**885,897.58**	189.6768	0.027065
	AGASA	0.40628	2.043839734	925,324.45	420.3396	0.032158
	PSO	0.467193	1.678281028	995,170.41	80.58469	0.02634
	MSPSO	**0.513508**	**1.456611067**	1,038,229.90	**64.84849**	**0.007197**
B15	GA	1.585272	0.578135752	1,028,150.25	167.421	0.077493
	AGASA	1.884987	0.481274575	961,762.33	336.9133	0.04799
	PSO	1.993432	0.455753393	913,981.35	86.50326	0.059692
	MSPSO	**2.068289**	**0.44004207**	**872,777.03**	**75.26807**	**0.026161**

As shown in Table 3, MSPSO produces the highest average fitness, the lowest average total penalty, and the lowest average standard deviation than other algorithms that shown in bold type value. MSPSO produce the lowest average cost on 2 formulae (B13 and B15). In case of B12 and B14, MSPSO produces the highest average cost than others. However, this result is associated with the average lowest penalty that may increase the cost.

GA and AGASA produce lower average fitness than PSO dan MSPSO. It shows us that particle swarm optimization approach is better than genetic algorithm approach in terms of

average fitness or solution quality. In term of time complexity, almost on all formulae (except B12), MSPSO require the lowest time than other algorithms. Furthermore, MSPSO can produce the most stable formula as shown in average standard deviation results. It demonstrates us that MSPSO as effective and efficient algorithm to optimize laying hen diet.

The results prove MSPSO as a robust algorithm to optimize laying hen diet. The diversity of particles is enhanced by cooperative sub-swarms that attracted on its own global best position which contributes towards diverse search trajectory and produce bettter solution. Furthermore, hypothetically speaking, MSPSO employs several sub-swarm that may operate parallelly in operating system level through multi-core processor. Thus, it requires less time than PSO which only employ one swarm. However, further study is necessary to understand empirically or theoretically how multi-swarm can produce less time than one swarm in PSO.

6. Conclusion

This study presents the optimization of feed composition for laying hen through MSPSO that accounting the balanced amino acids constraint. Furthermore, good swarm size, iteration and inertia weight were analyzed through several simulations on five different formulae.

The experimental results show that MSPSO as an effective and efficient approach to optimize laying hen diet. In parameter experimentation, MSPSO requires different optimum swarm size for each sub-swarm, iteration, and inertia weight for different ingredient choices. The good swarm size and iteration which are 50 and 16,000 respectively are determined by the maximum value found in a small sample. The inertia weight of 6.0 is found as good parameter choice on all sample. These parameter values should be used to optimize poultry diet using MSPSO. While in comparative experimentation, MSPSO produces best and stable solution quality than other algorithms. Furthermore, almost on all formulae, MSPSO require less time than others. Therefore, MSPSO is proven as a robust approach to optimize laying hen diet.

In the future study, free-parameter PSO variants may beneficial to solve the animal diet problem since the parameter is not necessarily defined explicitly that overcame the issue with a difference of the optimum parameter through ingredient choices. In addition, The time complexity could be reduced by shrinking the dimension when feed composition is near to zero which means that the ingredients are not used in the formula and considered not necessary to be computed in further PSO movement.

References

[1] Rahman RA, Ang C, Ramli R. Investigating Feed Mix Problem Approaches : An Overview and Potential Solution. *World Academy of Science, Engineering and Technology*. 2010; 46(10): 424–32.
[2] Oladoja MA, Olusanya TP. Impact of private feed formulation and production as a tool for poverty alleviation among poultry farmers in Ogun state, Nigeria. *International Journal of Poultry Science*. 2009; 8(10): 1006–10.
[3] Olugbenga SO, Abayomi OO, Oluseye AA, Taiwo AT. Nutrition & Food Optimized Nutrients Diet Formulation of Broiler Poultry Rations in Nigeria Using Linear Programming. *Journal of Nutrition & Food Sciences*. 2015; 1–6.
[4] Tozer PR. Least-Cost Ration Formulations for Holstein Dairy Heifers By Using Linear and Stochastic Programming. *Journal of Dairy Science*. 2016; 83(3): 443–51.
[5] Oladokun V, Johnson a. Feed formulation problem in Nigerian poultry farms: a mathematical programming approach. *American Journal of Scientific and Industrial Research*. 2012; 3(1): 14–20.
[6] Babić Z, Perić T. Optimization of livestock feed blend by use of goal programming. *International Journal of Production Economics.* 2011; 130(2): 218–23.
[7] Saxena P, Khanna N. Animal feed formulation: Mathematical programming techniques. In: *CAB Reviews: Perspectives in Agriculture, Veterinary Science, Nutrition and Natural Resources*. 2014.
[8] Akif Şahman M, Çunkaş M, Inal Ş, Inal F, Coşkun B, Taşkiran U. Cost optimization of feed mixes by genetic algorithms. *Advances in Engineering Software*. 2009; 40(10): 965–74.
[9] Rahman RA, Ramli R, Jamari Z, Ku-Mahamud KR. *Evolutionary Algorithm Approach For Solving Animal Diet Formulation*. Proceedings of the 5th Intl. Conference on Computing and Informatics, Istanbul. 2015; 274–279.
[10] Abd Rahman R, Ramli R, Jamari Z, Ku-Mahamud KR. Evolutionary algorithm with roulette-tournament selection for solving aquaculture diet formulation. *Mathematical Problems in Engineering*. 2016; 2016.
[11] Wijayaningrum VN, Mahmudy WF, Natsir MH. Optimization of Poultry Feed Composition using Hybrid Adaptive Genetic Algorithm and Simulated Annealing. *Journal of Telecommunication, Electronic and Computer Engineering*. 2017; 9(2): 1–5.

[12] Fatyanosa TN, Utaminingrum F, Data M. Linear Programming Initialization Method of Evolution Strategies for Beef Cattle Feed Optimization. *Journal of Telecommunication, Electronic and Computer Engineering*. In Press. 2018.
[13] Altun AA, Şahman MA. Cost optimization of mixed feeds with the particle swarm optimization method. *Neural Computing and Applications*. 2011; 22(2): 383–90.
[14] Arasomwan MA, Adewumi AO. Improved particle swarm optimization with a collective local Unimodal search for continuous optimization problems. *The Scientific World Journal*. 2014; 2014.
[15] Qin Q, Cheng S, Zhang Q, Li L, Shi Y. Biomimicry of parasitic behavior in a coevolutionary particle swarm optimization algorithm for global optimization. *Applied Soft Computing*. 2015; 32: 224–40.
[16] Tahir M, Pesti GM. Comparison of ingredient usage and formula costs in poultry feeds using different amino acid digestibility databases. *Journal of Applied Poultry Research.* 2012;21(3):693–705.
[17] Van Den Bergh F, Engelbrecht AP. A study of particle swarm optimization particle trajectories. *Information Sciences*. 2006; 176(8): 937–71.
[18] Jie J, Wang W, Liu C, Hou B. *Multi-swarm particle swarm optimization based on mixed search behavior*. Proceedings of the 5th IEEE Conference on Industrial Electronics and Applicationsis. Taichung. 2010; 605–10.
[19] Utomo MCC, Mahmudy WF, Anam S. Determining the Neuron Weights of Fuzzy Neural Networks Using Multi-Populations Particle Swarm Optimization for Rainfall Forecasting. *Journal of Telecommunication, Electronic and Computer Engineering*. 2017; 9(2): 37–42.
[20] Eberhart R, Kennedy J. *A new optimizer using particle swarm theory*. MHS'95 Proceedings of the Sixth International Symposium on Micro Machine and Human Science. 1995; 39–43.
[21] Ab Ghani MR, Hussein ST, Mohd Amin NH, Jano Z, Sutikno T. Dynamic Economic Dispatch Problems: PSO Approach. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*. 2017; 15(1): 48.
[22] Chandram K, Subrahmanyam N, Sydulu M. *Secant method combined with PSO for economic dispatch with prohibited operating zones*. Proceedings of the IEEE/PES Power Systems Conference and Exposition (PSCE). Convention Center Seattle. 2009;
[23] Novitasari D, Cholissodin I, Mahmudy WF. Hybridizing PSO with SA for optimizing SVR applied to software effort estimation. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*. 2016; 14(1): 245–53.
[24] Lu Y, Niu D, Li B, Yu M. Cost Forecasting Model of Transmission Project based on the PSO-BP Method. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*. 2014; 12(4): 773.
[25] Garro BA, Vázquez RA. Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms Beatriz. *Computational Intelligence and Neuroscience*. 2015; 2015.
[26] Shi Y, Eberhart R. *A modified particle swarm optimizer*. IEEE International Conference on Evolutionary Computation Proceedings. 1998; 69–73.