Journal of Theoretical and Applied Information Technology
31st December 2019. Vol.97. No 24
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# LEARNING PATH MODEL BASED ON COGNITIVE CLASSIFICATION USING HYBRID DISCRETE PARTICLE SWARM OPTIMIZATION

**[1,2]EKO SUBIYANTORO, [3]AHMAD ASHARI*, [4]SUPRAPTO**

[1]Department of Technology Informasi, PPPPTK-BOE VEDC,

Malang, Indonesia

[2,3,4]Department of Computer Sciences and Electronics, Universitas Gadjah Mada,
Yogyakarta, Indonesia

E-mail: [1]ekovedc@gmail.com, [3]ashari@ugm.ac.id*, [4]sprapto@ugm.ac.id

## ABSTRACT

Revised Bloom's Taxonomy (RBT) brings up taxonomic tables which are interrelations between cognitive processes and knowledge. Taxonomic tables can measure the depth and breadth of learning goals to be achieved. The variety of characteristics of students' abilities in a class has always been a problem that is often faced by a teacher. Unfortunately, cognitive classification to develop student knowledge towards Higher Order Thinking Skills has not been used to plan the learning path model. The purpose of this study is to determine the learning path recommendation that is appropriate to students' cognitive abilities based on the revised Bloom Taxonomy and ontology learning objects. The cognitive classification of students uses the Learning Vector Quantization (LVQ) method to get three cognitive classes (Cognitive Low, Cognitive Medium, and Cognitive High). Whereas to determine the learning path using the Hybrid Discrete Particle Swarm Optimization (HDPSO) method to overcome combinatorial problems, namely the learning object ontology with discrete PSO that is controlled by cognitive classes using binary PSO. The determination of the learning path is based on testing the RBT connection quality between LO and the ontology of a subject controlled by the student's cognitive class. The RBT cognitive classification results of the developed model can identify student cognitive with very high accuracy through determining the appropriate learning rate on the LVQ network. While the Hybrid Discrete Particle Swarm Optimization (HDPSO) method applied can overcome combinatorial problems more practically and regularly in determining the learning path. Experimental studies show that the models and techniques presented are suitable for finding a learning path that fits a student's cognitive class.

**Keywords:** *RBT, learning object, ontology, learning path, HDPSO*

## 1. INTRODUCTION

Benjamin S. Bloom, in 1949, put forward his idea of the distribution of cognitive taxonomy to facilitate the process of compiling questions so that they have the same learning objectives. Revised Bloom's Taxonomy (RBT) is proposed in general to look ahead and respond to the demands of the development of the educational community, including on how children develop and learn and how teachers prepare to teach materials. Bloom's taxonomy has only one dimension, while RBT has two dimensions namely cognitive processes and knowledge. Interrelations between cognitive processes and knowledge are called taxonomic tables. The cognitive process dimension (the columns in the table) contains six categories, namely remembering (C1), understanding (C2), applying (C3), analyzing (C4), evaluating (C5), and creating (C6). Understanding is a more complex level than remembering; applying has more complex levels of cognitive than understanding, and so on. Whereas the dimension of knowledge (ie, the rows in the table) contains four categories, namely factual, conceptual, procedural, and metacognitive [1]. These six levels are a series of levels of human thought. Based on these levels, it can be seen that thinking to remember is the lowest level of thinking (lower) while the highest level of thinking (higher) is to create.

Higher Order Thinking Skills (HOTS) [2][3] is a student thinking activity that involves a high level of cognitive level from Bloom's taxonomy of

* Corresponding author's

thinking including (C4) analyzing, (C5) evaluating and (C6) creating [4]. HOTS activities sharpen students' skills in seeking knowledge in inductive and deductive reasoning to think of answers or identify and explore scientific examinations of existing facts [5]. Students can process information and make the right and fast decisions in the present. Students need to develop logical thinking and reasoning based on facts.

The heterogeneity of most classes is often cited as a cause of difficulties. It is common to find students with very different levels of knowledge, motivation, commitment, and learning rhythm. Therefore, it is difficult for teachers to follow an approach that is suitable for each student. To reach all students, teachers often design lectures and activities in the classroom. To improve this situation personal support and guidance are needed, so that individual needs and difficulties can be overcome. However, given the number of students owned by the class, it is not easy for a teacher to handle the diversity of levels and student needs at all times [6].

Curriculum sequencing (CS) is a technique to provide students in planning the most appropriate sequence of learning tasks individually [7]. CS not only helps students determine the most appropriate learning path but also enable teachers to organize program structure, create content or learning object, and make improvement [8]. The purpose of CS is to replace the structure of rigid, general learning methods, and one suitable model set by the teacher or pedagogical team becomes a more flexible and personalized learning path. So that individualization of teaching materials is challenged in choosing the right LO and making LO sequences that are easy to learn . This suitability of learning paths and students' cognitive abilities will produce an optimal result.

Many studies in the CS domain had already applied evolution algorithm (EA) approach include using genetic algorithms, namely pedagogic sequence determination through approaches to matching keywords and difficulty levels [9], pedagogic sequence determination by minimizing the average difference between the level of compatibility of learning objects and participant satisfaction level [10], and pedagogical sequence genetic algorithms through calculating distance in LO [11]. Contrast to the EA method, the swarm intelligence approach emphasizes more on cooperation than competition [12]. In supporting cooperation concept, each agent has equipped with a simple ability to learn from experiences and communicate with fellow agents. The metaheuristic method based on the swarm intelligence concept is Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO).

This research proposes a learning path model based on Revised Bloom's Taxonomy cognitive classification and a learning object ontology. Learning Vector Quantization (LVQ) method is used to classify cognitive into three cognitive classes (Cognitive Low (CL), Cognitive Medium (CM), and Cognitive High (CH). Whereas Hybrid Discrete Particle Swarm Optimization (HDPSO) is used to overcome combinatorial problems, namely ontology learning objects with Discrete PSO controlled by cognitive classes using Binary PSO. The determination of the learning path is based on testing the RBT connection quality between LO and the ontology of a subject controlled by cognitive class.

## 2. LITERATURE REVIEW

### 2.1 Learning Vector Quantization

*Learning Vector Quantization (LVQ)* is a guided Artificial Neural Network (ANN) that uses competitive learning methods which were first developed by Kohonen in 1996 [13]. The Learning Vector Quantization is a learning method in the supervised competitive layer. The competitive layer will automatically learn to classify input vectors [14]. The classes obtained as a result of this competitive layer depend only on the distance between the input vectors. In case several input vectors have a very close distance then the input vectors will be grouped in the same class.

In order to understand the LVQ techniques its should be borne in mind that the closest weight vector $w_j$ to a pattern $X$ may be associated with a node $j$, that has the wrong class label for $X$. This followss because the initial node labels are based only on their most frequent class use and are therefore not always reliable[14]. The procedure for updating $w_j$ is given by,

$$\Delta w_j = \begin{cases} \alpha(x - w_j) & : \text{if } x \text{ classified correctly} \\ -\alpha(x - w_j) & \text{if } x \text{ classified incorrectly} \end{cases} \quad (1)$$

The negative sign equation (1) in the misclassification makes the weight vector move from the clustercontaining x, which, on average, tends to make weight vector move away from class boundaries.

LVQ is directed to determine the output unit that best matches the target of the input vector by shifting the position of the representative vector. If the training data vector $X$ is grouped together with the vector of the winning $Wc$, then the representative vector is shifted closer to the training vector with equation (2)

$$W_c(t+1) = W_c(t) + \alpha(t)[X(t) - W_c(t)] \quad (2)$$

Conversely, if the training data vector $X$, grouped is not the same as the vector of the winning $Wc$, then the representative vector is shifted away from the training vector with equation (3)

$$W_c(t+1) = W_c(t) - \alpha(t)[X(t) - W_c(t)] \quad (3)$$

where $\alpha$ is learning rate, $W_c$ is the position of the representative vector when $t$;$X$ is the position of the input vector when $t$.

The method used to calculate vector distance in LVQ networks is the euclidean distance used in the learning process to achieve accurate classification [15], as a basic rule of competition as in equation (4)

$$D_j = \|\mathbf{x} - \mathbf{w}_j\| = \sqrt{\sum_i (\mathbf{x}_i - \mathbf{w}_{ij})^2} \quad (4)$$

where $D_j$ is the euclidean distance at time $j$, $X_i$ is input vector $i$, and $w_{ij}$ is weight vector $i$ for output unit $j$. The basic LVQ algorithm is described in Figure 1.

```
Step 1 :
    Initialize reference vectors $w_i$
    Initialize learning rate $\alpha$
Step 2 :
    While stopping is false , do Steps 3-7
        Step 3 :
            For each training input vector x, do Steps 4-5
                Step 4 :
                    Compute j using Euclidean distance,
                        $D_{(j)} = \sum (w_{ij} - x_i)^2$
                    Find j when $D_{(j)}$ is minimum
                Step 5 :
                    Update $w_j$ as follows:
                        If $T = c_j$ , then
                            $w_{j(new)} = w_{j(old)} + \alpha[x - w_{j(old)}]$
                        If $T \neq c_j$ , then
                            $w_{j(new)} = w_{j(old)} - \alpha[x - w_{j(old)}]$
Step 6 :
    Reduce the learning rate $\alpha$
Step 7 :
    Test for the stopping condition
```

*Figure 1: LVQ Algorithm*

## 2.2 Particle Swarm Optimization

Inspired by bird group social behavior, Dr. Eberhart and Dr. Kennedy developed Particle Swarm Optimization (PSO) is a population-based stochastic optimization technique in 1995[16]. The PSO algorithm works based on particles in the population that work together to solve existing problems disregarding the physical position [17][18]. The PSO algorithm combines local and global search methods that balance exploration (ability to conduct investigations in different areas of the search area to get the best optimal value) and exploitation (ability to concentrate around the search area for fix solution).

The similarity of PSO and GA is that the system starts with a population formed from random solutions, then the system seeks optimization through random generation changes. Each particle holds traces of position in the search space as the interpretation of the best solution (*fitness*) that had been achieved.

There are three stages in the basic algorithm of PSO, namely generation of position and velocity of particles, velocity updates and position updates. First Step, position $x_i^t$ and velocity $v_i^t$ from a collection of particles randomly generated using the upper limit ($xmax$) and the lower limit ($xmin$) of the variable design shown in (5) and (6),

$$x_0^t = x_{min} + rand(x_{max} - x_{min}) \quad (5)$$
$$v_0^t = x_{min} + rand(x_{max} - x_{min}) \quad (6)$$

The second step is to update the latest speed ($v_{i+1}$) on each particle at time t + 1 based on the previous speed ($v_i$) and the two best positions that have been searched ( $P_{best}$ and $G_{best}$). The update velocity formulation includes several random parameters, inertia factor ($w$), self-confidence $c_1$), swarm confidence ($c_2$) shown in (7),

$$v_{i,j}^{t+1} = wv_{i,j}^t + c_1 r1(Pbest_{i,j}^t - x_{i,j}^t) + c_2 r2(Gbest_{g,j}^t - x_{i,j}^t) \quad (7)$$

The third step is to update the particle position ($x_i^{t+1}$) based on its velocity ($v_i^{t+1}$). The alteration of particle position is hoped to gain optimal solution. The update of the particle position is shown in (8),

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (8)$$

## 2.3 Binary Particle Swarm Optimization

In 2007 Engelbrecht developed PSO to operate in binary search space because real-value domains can be converted into binary-value domains [19]. The proposed algorithm is called the binary PSO algorithm (BPSO) where particles represent positions in binary space and the vector position of particles can take binary values of 0 or 1 where $x_{ij} \in \{0,1\}$. In this case, it maps from the binary space of the dimension $B^n$ (eg long string bits n) to real numbers $f = B^n \to R$ (where f is a fitness function and R is a set of real numbers). That means the position of the particle must belong to $B^n$ to be calculated by the fitness function [20]. In BPSO, particle velocity $v_{ij}^t$ is connected to the possibility that the position of particle $x_{ij}^t$ is 0 or 1. Equation (8) for updating particle velocity is still used in BPSO. Next, the $S_{ij}^t$ sigmoid function shown in equation (10) is used to update the particle velocity.

$$S_{ij}^t = \frac{1}{1 + e^{-v_{ij}^{t+1}}} \quad (10)$$

The position of the particle $x_{ij}^t$ is updated by equation (11),where $u_{ij}^t$ is a random number chosen from the distribution (0, 1) and $S_{ij}^t$ sigmoid function.

$$x_{ij}^t = \begin{cases} 1 \text{ if } u_{ij}^t < S_{ij}^t \\ 0 \text{ if } u_{ij}^t \geq S_{ij}^t \end{cases} \quad (11)$$

## 2.4 Discrete Particle Swarm Optimization

In 2000, Clerc modified the PSO algorithm which was formulated by Kennedy and Eberhart [21]. Clerc modified the representation of the position of the particles, the shape of the velocity produced by the particles and the effect of velocity on the position of the particles. The expectation of these modifications is to be applied to problems with discrete models especially combinatorial types [22]

$$v_i^{t+1} = c_1 v_i^t \oplus c_2 \left( \left( Pbest_i^t + \frac{1}{2}\left(Gbest_g^t - Pbest_i^t\right) \right) - x_i^t \right) \quad (12)$$

The framework of PSO for discrete optimization problems proposed by Goldbarg[23][24]is shown in Figure 2.

```
Procedure Discrete_PSO
/* Define initial probabilities for particles' moves:*/
pr1 ← a1 /*to follow its own way*/
pr2 ← a2 /*to go towards Pbest*/
pr3 ← a3 /*to go towards Gbest*/
/* a1+ a2+ a3=1 */
Initializa the population of particles
do
    for each particle i
        $value_i$ ← Evaluate($x_i$)
        if $f(value(x_i) < f(value(Pbest_i)$ then
            $Pbest_i \leftarrow x_i$
        if $f(value(x_i) < f(value(Pbest_i)$ then
            $Gbest_i \leftarrow x_i$
    end
    for each particle i
        $velocity_i$ ← define_velocity($pr_1, pr_2, pr_3$)
        $x_i$ ← update($x_i$, $velocity_i$)
    end
  /* Update probabilities*/
  $pr_1 = pr_1 \times 0.95$;
  $pr_2 = pr_2 \times 1.01$;
                $pr_3 = 1 - (pr_1 + pr_1)$;
while ( a stop creterion is not satisfied )
```
*Figure 2: Pseudo-code of DPSO*

The coefficients $c_1$ and $c_2$ have the same meaning stated previously and the signal $\oplus$ represents a composition. In initial applications of the proposed approach, only one of the three primitive moves is associated with each particle of the swarm at each iteration step. Thus, $c_1, c_2 \in \{0,1\}$ and $c_1 + c_2 = 1$ in equation (12). The assignment is done randomly. Initial probabilities are associated with each possible move and, during the execution, these probabilities are updated. Initially, a high value is set to $pr_1$, the probability of particle $i$ to follow its own way, a lower value is set to $pr_2$, the probability of particle $i$ goes towards $P_{best}$ and the lowest value is associated with the third option, to go towards $G_{best}$. The algorithm utilizes the concept of social neighborhood and the $G_{best}$ of all particles is associated with the best current solution, $G_{best}$. The initial values set to $pr_1$, $pr_2$, and $pr_3$ are 0.9, 0.05 and 0.05, respectively. As the algorithm runs, $pr_1$ is decreased and the other probabilities are increased. At the final iterations, the highest value is associated with the option of going towards $G_{best}$ and the lowest probability is associated with the first move option.

## 3. RESEARCH METHOD

### 3.1 Proposed Architecture

The model used in this study consists of three components, cognitive classification with LVQ, learning object ontology based on RBT, and hybrid discrete particle swarm optimization. The general architecture of the proposed model can be seen in Figure 3.

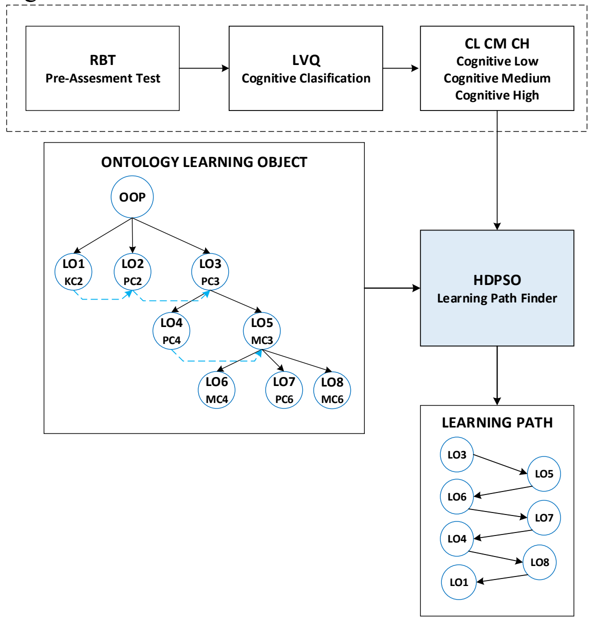


*Figure 3: The Architecture of The proposed Model*

The Hybrid Discrete Particle Swarm Optimization algorithm is applied to overcome combinatorial problems in a more practical and orderly manner in determining learning pathways. Determination of the order of learning objects through LO ontology based on cognitive classes from the learning vector quantization method and using RBT to assess connection quality. The expected result is that each student gets a recommendation for a learning path that is appropriate to their cognitive level.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

### 3.2 Cognitive Classification With LVQ

The classification structure and function model using the Learning Vector Quantization network proposed in this study are presented Figure 4. and Table 1. The model structures under development are; a) Making questions based on revised bloom taxonomy, b) Cognitive classification using LVQ,

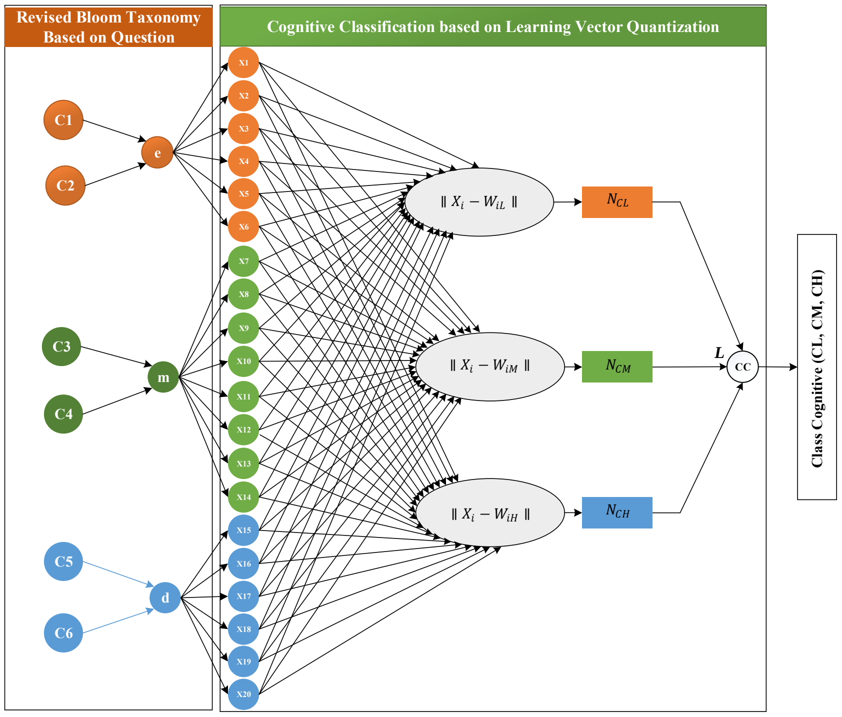


*Figure 4: Cognitive Classification RBT*

*Table 1: Notation of Classifier Cognitive Sturctuture*

| Notation | Description |
|---|---|
| $e$ | The number of easy category questions |
| $m$ | The Number of medium category questions |
| $d$ | The number of difficult category questions |
| $X$ | Input vector $X_i = (X_1, X_2, \dots X_{20})$ |
| $W_L$ | Weight vector to cognitive low $W_L = (W_{1L}, W_{2L}, \dots W_{20L})$ |
| $W_M$ | Weight vector to cognitive medium $W_M = (W_{1M}, W_{2M}, \dots W_{20M})$ |
| $W_H$ | Weight vector to cognitive high $W_H = (W_{1H}, W_{2H}, \dots W_{20H})$ |
| $N_{CL}$ | Cognitive low value |
| $N_{CM}$ | Cognitive medium value |
| $N_{CH}$ | Cognitive high value |
| $L$ | Cognitive class classification |
| $CC$ | Class Cognitive |
| $CL$ | Class Cognitive Low |
| $CM$ | Class Cognitive Medium |
| $CH$ | Class Cognitive High |

Criteria's ratio for easy, medium, and difficult questions is 3: 4: 3. The difficulty level of the question corresponds to Bloom's cognitive taxonomy hierarchy. Easy question categories are developed based on the level of cognitive ability to remember (C1) and understand (C2). The category intermediate question is developed based on the level of cognitive ability to apply (C3) and analyze (C4). Whereas, difficult question categories are developed based on the level of cognitive ability to evaluate (C5) and create (C6).

The amount of questions is determined by appointed reference, for example, A is a cognitive set that contains all forms of tests in an assessment. The number of cognitive groups in the form of tests is $N = \{ e, m, d \}$. Where $e, m$, dan $d$ are the category of test questions, e is the number of easy category questions, m is the number of medium category questions, and d is the number of difficult problem categories, shown in equation (13).

$$C1, C2 \leftrightarrow e \,;\, C3, C4 \leftrightarrow m \,; C5, C6 \leftrightarrow d$$

$$n = 0.3e \times 0.4m \times 0.3 \qquad (13)$$

LVQ is used to classify twenty cognitive data on RBT. The input vector is X. The weight vectors are $W_L$, $W_M$, and $W_H$. The results of LVQ are three groups of cognitive data types, namely; CL, CM, and CH. In determining values of $N_{CL}$, $N_{CM}$, and $N_{CH}$ are by calculating the distance between the input vector with the weight vector shown in equations (14) (15) and (16).

$$e \leftrightarrow X_1 \dots X_6 \,, m \leftrightarrow X_7 \dots X_{14} \,, d \leftrightarrow X_{15} \dots X_{20} \,, W_L \leftrightarrow N_{CL}$$

$$N_{CL} = \sqrt{\sum_i (X_i - W_{iL})^2} \quad (14)$$

In equation (14), $N_{CL}$ is the value of cognitive low. If $n$ in equation (13) is 20, then the input vector which is six cognitive levels in RBT can be written with $X_i = (X_1, X_2, X_3, \dots X_{20})$ and the weight vector $W_L$ is a weight vector that connects each neuron in the input layer to the first neuron in the output $W_{iL} = \left(W_{11}, W_{21}, W_{31}, \dots, W_{20(1)}\right)$.

$$e \leftrightarrow X_1 \dots X_6, m \leftrightarrow X_7 \dots X_{14}, d \leftrightarrow X_{15} \dots X_{20}, W_L \leftrightarrow N_{CM}$$

$$N_{CM} = \sqrt{\sum_i (X_i - W_{iM})^2} \quad (15)$$

In equation (15), $N_{CM}$ is the value of cognitive medium. If $n$ in equation (13) is 20, then the input vector which is six cognitive levels in RBT can be written with $X_i = (X_1, X_2, X_3, \dots X_{20})$ and the weight vector $W_M$ is a weight vector that connects each neuron in the input layer to the first neuron in the output $W_{iM} = \left(W_{12}, W_{22}, W_{32}, \dots, W_{20(2)}\right)$.

$$e \leftrightarrow X_1 \dots X_6, m \leftrightarrow X_7 \dots X_{14}, d \leftrightarrow X_{15} \dots X_{20}, W_L \leftrightarrow N_{CH}$$

$$N_{CH} = \sqrt{\sum_i (X_i - W_{iH})^2} \quad (16)$$

In equation (16), $N_{CH}$ is the value of cognitive high. If $n$ in equation (13) is 20, then the input vector which is six cognitive levels in RBT can be written with $X_i = (X_1, X_2, X_3, \dots X_{20})$ and the weight vector $W_H$ is a weight vector that connects each neuron in the input layer to the first neuron in the output $W_{iH} = \left(W_{13}, W_{23}, W_{33}, \dots, W_{20(3)}\right)$.

Some researchers use LVQ-based optimum method [18] [19]. L is a classification CC's optimum conditions. There are three definitions of L's optimum conditions probabilities, namely; 1) Cognitive Low, 2) Cognitive Medium, and 3) Cognitive High as shown in equation (17) (18)

$$L = arg\ min \parallel \{N_{CL}, N_{CM}, N_{CH}\} \parallel, \quad (17)$$

$$CC = \begin{cases} CL, if \ \ L = N_{CL} \\ CM, if\ L = N_{CM} \\ CH, if\ L = N_{CH}. \end{cases} \quad (18)$$

L is CL if the low cognitive value is smaller than medium cognitive value and smaller than high cognitive value as shown in equation (17) thus CC is CL as in equation (18). *L* is considered a CM if the medium cognitive value is smaller than the cognitive low and smaller than cognitive high, then *CC* is *CM*. Whereas *L* is considered as CH if the value of cognitive high is smaller than the cognitive medium value and smaller than cognitive low value, then *CC* is *CH*. Hence the consequence of the value of cognitive distances (C1, C2, C3, C4, C5, C6) considerably influences the results of cognitive classification.

## 3.3 Learning Object Mapping RBT

Learning activities often involve both lower order and higher order thinking abilities that include ways of thinking concrete and abstract knowledge. The dimensions of cognitive processes are a continuum in increasing cognitive complexity from low-level thinking skills to higher thinking skills. According to Krathwohl[1], in identifying nineteen specific cognitive processes to clarify the scope of six classification categories. Concept map of analyzing the depth and breadth of learning objectives is shown in Table 2.

*Table 2: Analysis Of The Depth And Breadth of Determining Learning Objects*

| KNOWLEDGE DIMENSIONS | | BREADTH | | | | | |
|---|---|---|---|---|---|---|---|
| | | Remember (C1) | Understand (C2) | Apply (C3) | Analyze (C4) | Evaluate (C5) | Create (C6) |
| DEPTH | Factual | FC1 | FC2 | FC3 | FC4 | FC5 | F6 |
| | Conceptual | KC1 | LO1 | KC3 | KC4 | KC5 | KC6 |
| | Procedural | PC1 | LO2 | LO3 | LO4 | PC5 | LO7 |
| | Metacognitive | MC1 | M2 | LO5 | LO6 | M5 | LO8 |

Basic competency is the ability and least learning material that must achieved by students for a subject in each education unit that refers to core competencies. Table 3 presents the relationship between basic competencies with the learning objects in determining competency targets.

*Table 3: Metadata Learning Object*

| Basic Competency | Learning Object | Competency Target | Position |
|---|---|---|---|
| 3.1 | Object Oriented Methodology | KC2 | (2,2) |
| 3.2 | The Basic and Rules in Object Oriented Programming | PC2 | (2,3) |
| 3.3 | Class and Object | PC3 | (3,3) |
| 3.4 | Data Encapsulation and Information | PC4 | (3,4) |
| 3.5 | Inheritance | MC3 | (4,3) |
| 3.6 | Polymorphism | MC4 | (4,4) |
| 3.7 | Interface | PC6 | (3,6) |
| 3.8 | Package | MC6 | (4,6) |

## 3.4 Ontology Learning Object

The ontology of learning objects developed in this study refers to the first semester XI object-oriented programming subjects in software engineering expertise programs at Vocational High Schools (SMK).

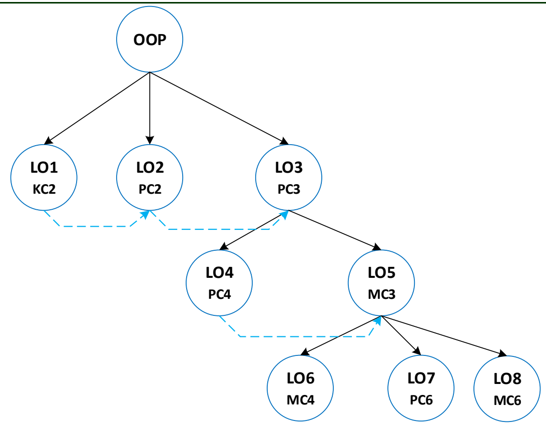


*Figure 5: Ontology Learning Object with RBT*

The distance values in the ontology are: the value of LO parent connected to its LO below it has a value of 1. Subjects distanced more than three levels is declared to have no connection. For those LO that are not connected to each other directly is valued 0.5. Table 4 presents the distance calculation data between LO in the ontology.

*Table 4: The Distance Value Of Each Lo In Ontology*

| LO | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0.5 | 1 | 2 | 2 | 3 | 3 | 3 |
| 2 | 0.5 | 0 | 0.5 | 1.5 | 2 | 2.5 | 2.5 | 2.5 |
| 3 | 0.5 | 0.5 | 0 | 1 | 1 | 2 | 2 | 2 |
| 4 | 2 | 1.5 | 1 | 0 | 1.5 | 1.5 | 1.5 | 1.5 |
| 5 | 2 | 2 | 1 | 0.5 | 0 | 1 | 1 | 1 |
| 6 | 3 | 2.5 | 2 | 1.5 | 1 | 0 | 0.5 | 1.5 |
| 7 | 3 | 2.5 | 2 | 1.5 | 1 | 0.5 | 0 | 2 |
| 8 | 3 | 2.5 | 2 | 1.5 | 1 | 1.5 | 2 | 0 |

## 3.5 The Proposed HDPSO

The application of the HDPSO algorithm in this study, starting with the LO particle representation, updating the velocity and position of the particles by transposition, calculating the fitness function based on the relationship between RBT and ontology, and finally writing the HDPSO algorithm to solve this problem.

### 3.5.1 Particle Representation

The particle representation in this combinatorial problem is to change the arrangement of the positions of each permutation value into an integer form from the solution representation. The solution of the combinatorial problem optimization case is to change the position arrangement of each permutation value into an integer form from the representation of the solution. The Hybrid Discrete Particle Swarm Optimization (HDPSO) is used to overcome combinatorial problems, namely ontology learning objects with Discrete PSO controlled by cognitive classes using Binary PSO. Figure 6 shows the learning object sequence randomly from three groups of particles.

| | | Discrete PSO | | | | | | | | Binary PSO | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Learning Object | | | | | | | | Cognitive Low | | | | | | | | Cognitive Medium | | | | | | | | Cognitive High | | | | | | | |
| $x_i^t$ | 1 | 7 | 2 | 8 | 4 | 5 | 3 | 1 | 6 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 2 | 5 | 3 | 4 | 8 | 1 | 7 | 6 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 3 | 8 | 6 | 2 | 1 | 4 | 7 | 3 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| $Pbest_i^t$ | 1 | 7 | 2 | 8 | 4 | 5 | 3 | 1 | 6 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 2 | 5 | 3 | 4 | 8 | 1 | 7 | 6 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| | 3 | 8 | 6 | 2 | 1 | 4 | 7 | 3 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| $Gbest_i^t$ | 2 | 5 | 3 | 4 | 8 | 1 | 7 | 6 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

*Figure 6: Particle representation at iteration t = 0*

At the 0th iteration (t = 0), the value of all particle is $v_i(t) = \emptyset$ and the starting position of all particle is randomly generated in the form of integer numbers. These numbers represent LO number and uniquely combined. For example, LO $x_{i=1}$ [7 2 8 4 5 3 1 6] means that LO sequence is started from LO7 toward LO 2, 8, 4, 5, 3, 1, 6 and return to LO7. Meanwhile, cognitive class consists of CL [ 1 1 1 1 1 1 0 0] , CM [1 1 1 0 1 1 1 0] and CM [0 0 1 0 1 1 1 1] is used as LO controller.

*Connection Weight* (cw) and the amount of unused particle (*UnLO*) from each CL, CM, and CH classes are counted to determine Fitness Function. $P_{best}$ value at 0th iteration (t = 0) is the same value with particle starting position, i.e. $P_{best_i}(t) = x_i(t)$.

*Pbest* with the heightest fitness value determines $G_{best}$ value $(k = argMax_i\{fitness\ P_{best_i}(t)\} = 2)$, so that

$G_{best_{g=1}}(t=0) = P_{best_{i=2}}(t=0)$, i.e. $G_{best_1}(0)$ [5 3 4 8 1 7 6 2].

### 3.5.2 Update Position

Figure 7 shows learning object update positions. Transposition pattern allows learning object with particle $x_i$[7 2 8 4 5 3 1 6] and $G_{best_i}$ [5 3 4 8 1 7 6 2] target shifted several times. The shift was started from position (1,5)-(2,6)-(3,4)-(5,7)-(6,7)-(7,8). Equation (5) and (6) will produce particle position of $x_{i+1}$ [5 3 4 8 7 2 1 6].

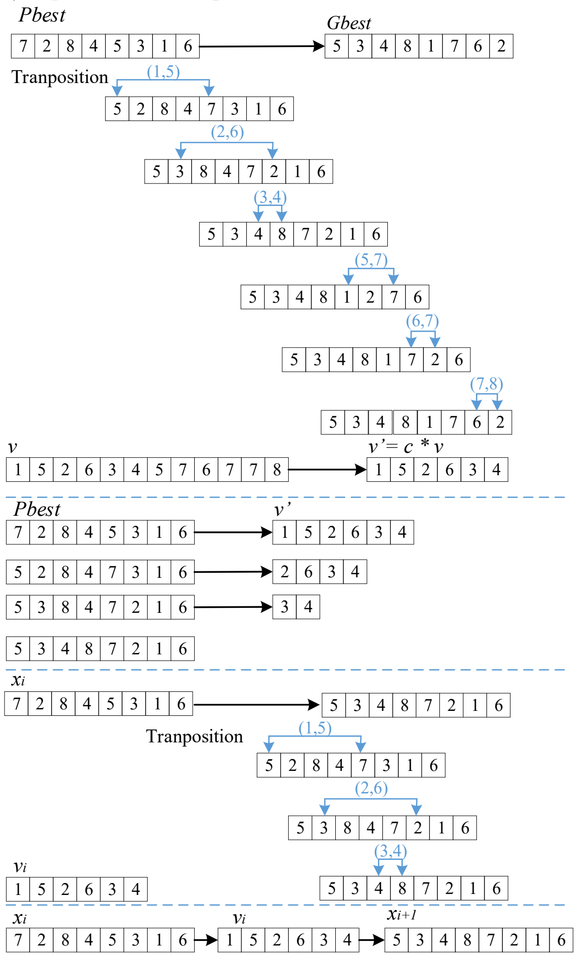


Figure 7: Learning Object Position Update

Velocity and position updates for the cognitive class are determined as follows, particles 7 and 8 in the CM class, particles 1 and 8 in the CM class, and particles 1, 2, and 4 in class CH. Other particles are not updated and marked by giving a value of 1 to the particle, as shown in Figure 8.

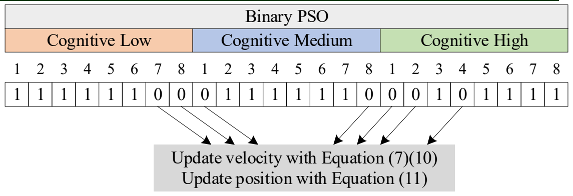

Binary PSO
Cognitive Low
Cognitive Medium
Cognitive High
Update velocity with Equation (7)(10)
Update position with Equation (11)

Figure 8: Class Cognitive Position Update

### 3.5.3 Fitness Function

The fitness function is used as a measuring tool to select the best object from a set of objects. In the evolution algorithm, the fitness function is responsible for determining the best solution from a set of existing solutions [25].

Connection Weight ($CW$) was used to assess the relationship of LO in RBT ontology as cognitive level evaluators [11] in equation 19. Cognitive level evaluators assess only the cognitive levels relationship of (C1, C2, C3, C4, C5, C6) where the value between levels is 1.

$$cw = \frac{k}{t_1 \cdot |DBO| + t_2 \cdot |DBB|} \tag{19}$$

This study uses Distance by Bloom (DBB) to measure the cognitive distance depth and breadth ($d_{k,l}$)between LO using equation 20,

$$d_{k,l} = \sqrt{(k_2 - k_1)^2 + (l_2 - l_1)^2} \tag{20}$$

Equation 20 is used to calculate the cognitive distance of LO1 with LO3, LO1 with cognitive target KC2 in cognitive position (2.2), while LO3 with cognitive target PC2 in cognitive position (3.2) obtained cognitive distance 1.414.

Distance by Ontology (DBO) is the distance found as the number of levels in an ontology. For example DBO distance calculation between "LO1" and "LO3". LO1 and LO2 in the ontology are not directly connected. The DBO calculation starts from the distance of LO1 to LO2 is 0.5 and LO2 to LO3 is 0.5, so DBO is equivalent to 1 level. The coefficients $t_1$and $t_2$depend on the type of LO which can be both theoretical and practical. For practical LO types, taxonomic distance (DBB) is more important. For theoretical LO types, ontology distance (DBO) is more important.

In the following is how to calculate the $CW$ value between "Object Oriented Medotology: KC2" and "Class and Object: PC2". By default, the value of k is 100, the value of $t_1$ is 1 because LO1 KC2 is theoretical, and the value of $t_2$ is equivalent to 5 because the LO3 PC2 is practical.

$$CW = \frac{k}{t_1 \cdot |DBO| + t_2 \cdot |DBB|} = \frac{100}{1 \cdot |1| + 5 \cdot |1{,}414|} = 12{,}392$$

The fitness function proposed in this study is to make an individual learning path or route based on RBT and the learning object ontology shown in, Equation (21) $FC_l$ is fitness function to determine LO relationships in CL class,

$$FC_L = \alpha_L * cw_L + \frac{1}{\beta_L * \sum UnLO_l} \tag{21}$$

with:
$\alpha_L, \beta_L$ is 0 -1.
$cw_l$ is *connection weight* for CL class
$UnLO_L$ is *unsed Learning Object* for CL class.
Equation (22) $FC_M$ is fitness function to determine LO relationships in CM class,

$$FC_M = \alpha_M * cw_M + \frac{1}{\beta_M * \sum UnLO_M} \tag{22}$$

with:
$\alpha_M, \beta_M$ is 0 -1.
$cw_M$ is *connection weight* for CM class
$UnLO_M$ is *unsed Learning Object* for CM class.
Equation (23) $FC_H$ is fitness function to determine LO relationships in CH class,

$$FC_H = \alpha_H * cw_H + \frac{1}{\beta_H * \sum UnLO_H} \tag{23}$$

with:
$\alpha_H, \beta_H$ is 0 -1.
$cw_H$ is *connection weight* for CH class
$UnLO_H$ is *unsed Learning Object* for CH class.

### 3.5.4 Application of the HDPSO Algorithm

The methodology, steps and strategies of the Hybrid Discrete Particle Swarm Optimization algorithm in detail are as follows:

**Step 1**: Initialization.
Initialize population, the number of iterations ($Itermax$), and speed of each particle. Particle position $x_i$ is LO arranged in a random generated array [1…n]. Cognitive class particles (CL, CM, CH) are arranged in arrays [n] [1 ... n] 1 ... n]. Calculate connection weight (CW) through DBO and DBB calculations with equation (19)(20) between LO.

**Step 2**: Fitness Function Calculation.
Calculate the fitness function of each cognitive class based on CW of each particle with equations (21), (22), and (23).

**Step 3**: Initialization of $P_{best}$ and $G_{best}$
The initialati value of Pbest Value is $x_i(t)$, select the $P_{best}$ with highest fitness value to determine

$$G_{best}(k = argMax_i\{fitnessP_{best_i}(t)\})$$

**Step 4**: Start the iteration, $iter = 1$

**Step 5**: Velocity Update
Update velocity for each LO with the transposition pattern using equation (12) and update the particle speed of the cognitive class (CL, CM, CH) using equation (7),(10).

**Step 6**: Position Update
Update particle position for each LO equation (8) and particle position of the cognitive class (CL, CM, CH) using equation (11), then calculate the fitness function of each cognitive class based on CW for each particle.

**Step 7**: $P_{best}$Update
Change the current particle $P_{best}$ with the current position of the particle if and only if the current fitness value is better than the previous $P_{best}$.

**Step 8**: $G_{best}$Update
Determine $G_{best}$ by choosing one $P_{best}$ with the highest fitness value.

**Step 9**: Iteration Termination Criteria
If the current iteration of the $iter < Itermax$, then proceed to Step 4, if not continue to Step 10.

**Step 10**: The outcome of the best $G_{best}$ position.

## 4. EXPERIMENTAL RESULTS

LVQ network testing to determine the cognitive class into three classes, namely CL, CM, and CH. After that the CW test is used to determine the quality of RBT and ontology relationships, then test and discuss the fitness function based on cognitive class with the number of particles used, and the final step is to test the HDPSO algorithm in order to display the learning path through the $G_{best}$ of each cognitive class (CL, CM, and CH).

### 4.1 LVQ Network Training on Cognitive Classification

Teachers are selected as respondents to get ideal cognitive characteristics based on the assumption that the teacher is the best evaluator of cognitive skills. Correspondingly, which is also a consideration is, that the teacher has the qualifications as a pedagogical assessor indicated by their diploma, certificate, and teaching experience. Hence, teachers are reliable to determine cognitive indicator parameters [26].

Teacher proffers variable reference weight that affects the cognitive class value (CC). Teacher's references variation includes (C1) remember, (C2) understand, (C3) apply, (C4) analyze, (C5) evaluate, and (C6) create. Teacher's variable reference value is a cognitive level characteristic used as cognitive reference data. Cognitive class references are the ideal cognitive values. Teachers' cognitive characteristics data is applicable to be applied during LVQ cognitive pattern learning.

The learning rate (α) is a network parameter in controlling the weight adjustment process. The optimal value of learning depends on the case at guidance. A meager learning rate causes slower network convergence while an enormous learning rate causes instability in the network. The coefficient α serves to converge the value of weight changes. The learning rate coefficient value is in the ranges of 0 <α coefficient <1. The reduction rate is determined by value β with equation (24) [27].

$$\beta = 0.7 \times (p)^{1/n} \tag{24}$$

where $p$ = number of hidden units and $n$ = number of input units. Learning rate upated (α) by equation (25)

$$\alpha_i = \alpha_i \times e^{(-\beta \times i)} \tag{25}$$

The amount of $p$ is 3 hidden units. The amount of $n$ is 20 input units. The calculation results β value is equal to 0.74. The initial learning rate value is 0.3. The α value is 0.182 at the first iteration. The α value is 0.0 at the maximum iteration (100).

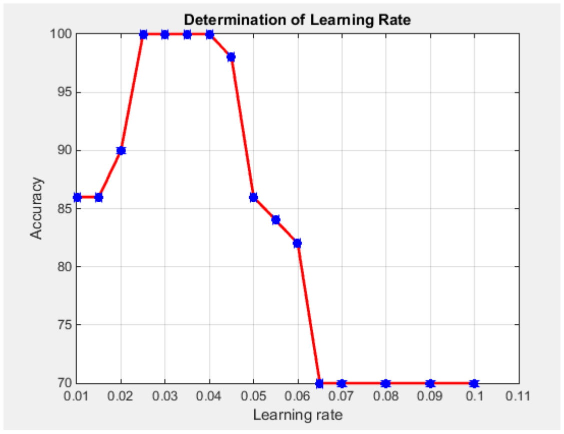


*Figure 8: Graph of Learning Rate Testing on Cognitive Classification*

The average learning rate accuracy is 85.125% shown in Figure 8. The learning rate value above 0.065 gives the lowest accuracy, which is 70%. Learning rate values from 0.025 up to 0.04 gives 100% accuracy. The learning rate value below 0.02 decreases accuracy steadily by 86%. Meager learning rate value accelerates convergence while an enormous learning rate value diverges the accuracy. This research applies the learning rate of 0.03.

## 4.2 Cognitive Classification Result

Based on equation (14) to equation (18), it can be stated that, this study is an LVQ network implementation to find out three cognitive classifications of 32 students, in the form of three groups of cognitive data types, namely; Cognitive Low (CL), Cognitive Medium (CM), and Cognitive High (CH). Table 4 shows the results of students' cognitive classifications. Thirty-one percent of students have low cognitive. Forty-four percent of students have moderate cognitive, while twenty-five percent of students have high cognitive.

*Table 4: Results of RBT Cognitive Classification Experiments with LVQ*

| ID St | Answers | | | | | | | | | | | | | | | | | | | | Class Cognitive | % |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | | | C2 | | | C3 | | | | C4 | | | | C5 | | | C6 | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | | |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | *CL* | |
| 22 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | *CL* | |
| 23 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | *CL* | |
| 32 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | *CL* | |
| 31 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | *CL* | |
| 29 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | *CL* | 31% |
| 30 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | *CL* | |
| 28 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | *CL* | |
| 27 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | *CL* | |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | *CL* | |
| 25 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | *CM* | |
| 24 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 21 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 9 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 15 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | *CM* | |
| 13 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 12 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | *CM* | |

| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 11 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | *CM* | 44% |
| 14 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | *CM* | |
| 10 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | *CM* | |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 2 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 3 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 16 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | *CM* | |
| 17 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | *CH* | 25% |
| 20 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | *CH* | |
| 19 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | *CH* | |
| 18 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | *CH* | |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | *CH* | |
| 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | *CH* | |
| 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | *CH* | |
| 7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | *CH* | |

## 4.3 Testing For Connection Weight RBT

The mechanism for testing connection weight according to in accordance with the procedure shown in Figure 9.

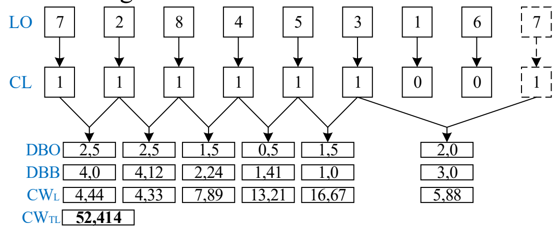


*Figure 9: Testing for CW*

The cognitive class CL controls LO. A connection weight calculation process occurs when one LO is paired together with CL of value 1. The calculation starts with finding the DBO, DBB, and CW values of each LO. In the state of one LO paired together with CL is 0 then the LO is not used in the connection weight calculation process. The CM and CH class calculations were using the same CW testing model. The complete CW testing is presented in Table 5.

*Table 5: Connection Weight Testing Data*

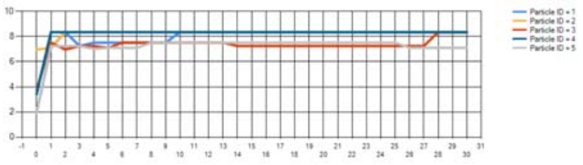
| No | Learning Object | | | | | | | | Class Cognitive | | | | | | | | | CW |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 7 | 2 | 8 | 4 | 5 | 3 | 1 | 6 | CL | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 52.41396 |
| | | | | | | | | | CM | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 49.1299 |
| | | | | | | | | | CH | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 50.65407 |
| 2 | 5 | 3 | 4 | 8 | 1 | 7 | 6 | 2 | CL | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 70.74803 |
| | | | | | | | | | CM | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 66.98564 |
| | | | | | | | | | CH | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 62.54119 |
| 3 | 8 | 6 | 2 | 1 | 4 | 7 | 3 | 5 | CL | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 72.99074 |
| | | | | | | | | | CM | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 64.23374 |
| | | | | | | | | | CH | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 50.51642 |

## 4.4 Fitness Fuction Testing

The fitness function testing is done to ensure the proposed learning path model can work well by three cognitive classes, namely CL, CM, CM. This experiment includes testing $P_{best}$ and $G_{best}$ of each cognitive class.

### 4.4.1 Fitness Fuction Testing at Cognitive Low

Figure 10 (a) presents a testing of fitness functions for class CL with LO consisting of 5 groups of particles, whereas Figure 10 (b) tests the fitness function with 10 groups of particles.



(a)

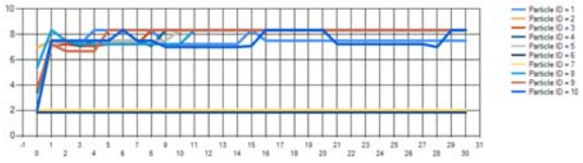

(b)

*Figure 10: Testing the $P_{best}$ Class CL with 5 Particles (a) and The $P_{best}$ Class CL with 10 Particles (b)*

### 4.4.2 Fitness Fuction Testing at Cognitive Medium

Figure 11 (a) presents a testing of fitness functions for class CM with LO consisting of 5 groups of particles, whereas Figure 11 (b) tests the fitness function with 10 groups of particles.

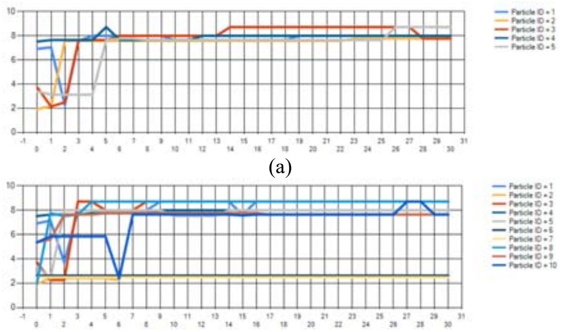

(a)
(b)

*Figure 11: Testing the $P_{best}$ Class CM with 5 Particles (a) and The $P_{best}$ Class CM with 10 Particles (b)*

### 4.4.3 Fitness Fuction Testing at Cognitive High

Figure 12 (a) presents a testing of fitness functions for class CH with LO consisting of 5 groups of particles, whereas Figure 12 (b) tests the fitness function with 10 groups of particles.

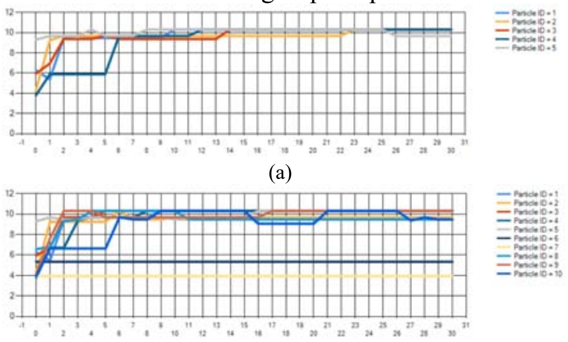

(a)
(b)

*Figure 12: Testing the $P_{best}$ Class CH with 5 Particles (a) and The $P_{best}$ Class CH with 10 Particles (b)*

The testing of the fitness function above shows that the higher the iteration that is used, the more optimal the solution produced by the system with the result of increasing fitness. This is due to the increasing number of iterations that are used to make particles move to find more optimal solutions, allowing particles to find the optimal solution.

### 4.5 Learning Path Recomendation

The graph in Figure 13 shows the $G_{best}$ value of each cognitive class.

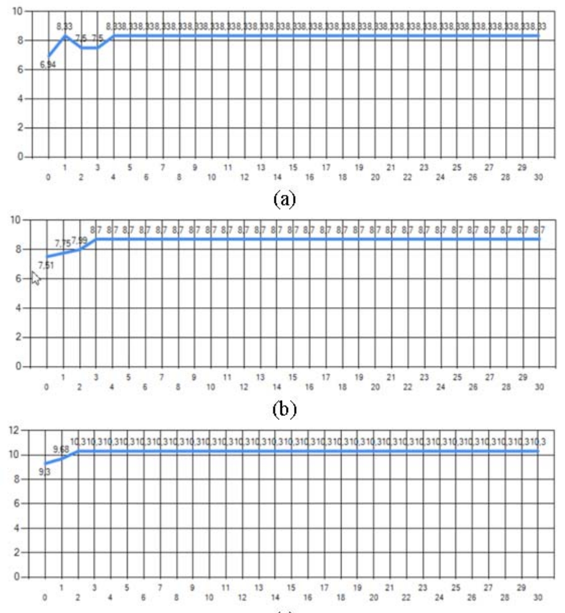

(a)
(b)
(c)

*Figure 13: Testing $G_{best}$ Class CL (a), $G_{best}$ Class CM (b), $G_{best}$ Class CH (c)*

$G_{best}$ is the best value that takes into account all particles in$G_{best}$ the population in each CL, CM, and CH class. Each iteration, each participant is given information about the latest Guest value so that there is a mechanism for sharing one-way information to carry out the process of finding the best solution with a fast convergence movement. The increasing value of fitness resulting representation can be caused by particles or particles such as evaluation and improvement strategies randomization strategy that is used to explore the entire swarm the existing space, or it is possible swarm existing space on this issue has a considerable scope.

Learning path recommendations based on the $G_{best}$ values shown in Table 6 indicate that an increase in the number of particles affects the value of the resulting learning path sequence, but is still within the schema of each cognitive class.

*Table 6: Connection Weight Testing Data*

| No | Class Cognitive | Number Of Particles | Learning Path | |
|---|---|---|---|---|
| | | | HDPSO | Manual Set |
| 1 | CL | 5 | 1-3-2-5-4-6-7-8 | 1-2-3-4-5-6-7-8 |
| 2 | | 10 | 1-2-3-5-4-6-7-8 | |
| 3 | CM | 5 | 3-5-6-7-4-2-8-1 | 1-2-3-4-5-6-7-8 |
| 4 | | 10 | 3-5-7-6-2-4-8-1 | |
| 5 | CH | 5 | 8-5-7-6-4-3-1-2 | 1-2-3-4-5-6-7-8 |
| 6 | | 10 | 8-5-7-6-4-3-2-1 | |

The HPSO algorithm can create learning pathways by CL, CM, and CH cognitive classes. The cognitive class CL tendency of its learning path (1-3-2-5-4-6-7-8) shows the order of the basic LO to a higher LO. Whereas the cognitive class CH of the learning path tendency (8-5-7-6-4-3-1-2) shows a sequence of high LO to the basic LO. In CM cognitive class the tendency of learning path is (3-5-7-6-4-2-8-1). The similarity of the learning path sequence based on the number of CL cognitive particles is 87.5%, CM is 75%, and CH is 87.5%, so the average similarity of the learning path sequence is 83.3%.

## 5. CONCLUSION

The first phase of this research succeeded in developing a RBT cognitive model based on Learning Vector Quantization. Teachers' cognitive abilities in the form of competency matrices based on RBT are used as training data. RBT cognitive cognitive model succeeded in classifying cognitive with very high accuracy through the determination of an appropriate learning rate (0.3). The test results of 32 students showed that the developed model can classify cognitive in three cognitive groups namely; 1) Cognitive Low with 31% results, 2) Cognitive Medium with 44% results, and 3) Cognitive High with 25% results.

Hybrid Discrete Particle Swarm Optimization (HDPSO) algorithm was applied to overcome combinatorial problems more practically and regularly in determining the learning path. Determination of the order of learning objects through an LO ontology is based on controlling cognitive classes (cognitive low, cognitive medium, and cognitive high) by using RBT to assess the quality of the connection. Experiments show that the models and techniques presented are suitable for finding learning paths that are suitable for students' cognitive classes.

In the future research is the development of HDPSO-based learning path model applications that can give teachers the flexibility in determining the learning object ontology based on the target Basic Competencies (KD) adjusted to RBT taxonomy tables which are cognitive interrelations (breadth) and knowledge (depth).

## REPRENCES:

[1] Krathwohl, D. R., Anderson, L. W., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., … Wittrock, M. C. “A Taxonomy For Learning, Teaching, And Assessing: A Revision Of Bloom’s Taxonomy Of Educational Objectives”, *New York Longman 41(4)*, 2002, 302.

[2] Saido, *et al.*, ”Teaching strategies scale for promoting higher order thinking skills among students in science,” *Proceedings of ISER 5th , International Conference,* 2015,91–94.

[3] D. Sukla and A.P. Dungsungneon, “Students Perceived Level and Teachers Teaching Strategies of Higher Order Thinking Skills A Study on Higher Educational Institutions in Thailand,” *Journal of Education and Practkice*, 7(12), , 2016, 211–219.

[4] Chinedu, C. C., Olabiyi, O. S., & Kamin, Y. Bin. “Strategies for improving higher order thinking skills in teaching and learning of design and technology education”. *Journal of Technical Education and Training*, *7*(2), 2015,35–43.

[5] Thitima, G., & Sumalee, C. “Scientific Thinking of the Learners Learning with the Knowledge Construction Model Enhancing Scientific Thinking”. *Procedia - Social and Behavioral Sciences*, 46(1999), 2012, 3771–3775.

[6] Santos, Á., Gomes, A., & Mendes, A.”A taxonomy of exercises to support individual learning paths in initial programming learning”. *Proceedings - Frontiers in Education Conference, FIE*, 2013, 87–93.

[7] De-Marcos, L., Pages, C., Martínez, J. J., & Gutiérrez, J. A. “Competency-based learning object sequencing using particle swarms”. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, 2, 2007,111–116.

[8] Al-Muhaideb, S., & Menai, M. E. B. “Evolutionary computation approaches to the Curriculum Sequencing problem”. *Natural Computing*, 10(2), 2011, 891–920.

[9] Huang, M. J., Huang, H. S., & Chen, M. Y. “Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach”. *Expert Systems with Applications*, 33(3), 2007, 551–564.

[10] Christudas, B. C. L., Kirubakaran, E., & Thangaiah, P. R. J. “An evolutionary approach for personalization of content delivery in e-learning systems based on learner behavior forcing compatibility of learning materials”. *Telematics and Informatics Jurnal*, 2017.

[11] Shmelev, V., Karpova, M., & Dukhanov, A. “An Approach of Learning Path Sequencing Based on Revised Bloom’s Taxonomy and Domain Ontologies with the Use of Genetic Algorithms”. *Procedia Computer Science* (Vol. 66), 2015.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[12] Kennedy, J., & Eberhart, R. “Particle swarm optimization”. *IEEE International Conference on Particle Swarm Optimization*, 1995.

[13] Kohonen, T. K., Hynninen, J., & Kangas, J. “LVQ PAK : The Learning Vector Quantization Program Package LVQ PAK : The Learning Vector Quantization Program Package”.1996.

[14] Fausett, L., “Fundamentals of Neural Networks Architectures, Algorithms, and Applications”. *Prentice-Hall*, Inc New Jersey:1994.

[15] Chen C-R., Tsai L-T. and Yang C-C., “A Neural Network Approach for Random Samples to Stratified Psychometrical Population”, *Proceedings of the WSEAS International Conference on SOCIOLOGY, PSYCHOLOGY, PHILOSOPHY*, Penang ,2010, pp. 51-54.

[16] J. Kennedy and R. C. Eberhart, “A discrete binary version of the particle swarm algorithm”, *IEEE, International Conference on Systems*, Man, and Cybernetics, Orlando, FL, vol.5, 1997, pp.4104-4108.

[17] Engelbrecht, A. P., “Computational Intelligence An Introduction (Second Edi)”, *John Wiley & Sons Ltd*, England, 2007.

[18] Li, X., & Deb, K. “PSO Niching algorithms Using Different Position Update Rules”, 2010.

[19] Engelbrecht, A. P., “Computational Intelligence An Introduction*”*. England: *John Wiley & Sons Ltd*, 2007.

[20] Das, S., Abraham, A., & Konar, A., “Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives”, *Studies in Computational Intelligence*, *116*, 2008, 1–38.

[21] Kennedy, J. and Eberhart, R.C., “A discrete binary version of the particle swarm algorithm”,1997.

[22] Clerc, M. “Discrete Particle Swarm Optimization Illustrated by the Traveling Salesman Problem”, 2000.

[23] Goldbarg, E.F.G; Souza, G.R. & Goldbarg, M.C. “Particle swarm for the traveling salesman problem”. *Proceedings of the EvoCOP*, Vol. 3906, Budapest, Hungary, April 2006a, pp. 99-110, ISBN: 3540331786.

[24] Goldbarg, E.F.G; Souza, G.R. & Goldbarg, M.C.”Particle swarm optimization for the biobjective degree-constrained minimum spanning tree”. *Proceedings of Congress on Evolutionary Computation*, Vol. 1, pp. 420-427, ISBN: 0780394879, Vancouver, BC, Canada, July 2006b, IEEE.

[25] Tang, P. and Lee, G.K. “An Adaptive Fitness Function for Evolutionary Algorithms Using Heuristics and Prediction”. *World Automation Congress*, 2006.

[26] M.A. Syufagi, *et al.*, “Petri Net Model for Serious Games based on Motivation Behavior Classification,” *International Journal of Computer Games Technology (IJCGT)*, 2013.

[27] Marcin Blachnik and Włodzisław Duch, “LVQ algorithm with instance weighting forgeneration of prototype-based rules,” *Elsevier*, 2011.