Received May 11, 2018, accepted June 11, 2018, date of publication June 15, 2018, date of current version July 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2847334

Optimizing Dynamic Multi-Agent Performance in E-Learning Environment

MARIAM M. AL-TARABILY[1], REHAB F. ABDEL-KADER[1], GAMAL ABDEL AZEEM[1], AND MAHMOUD I. MARIE[2]
[1]Electrical Engineering Department, Faculty of Engineering, Port Said University, Port Fouad 42526, Egypt
[2]Computers and Systems Engineering Department, Faculty of Engineering, Al-Azhar University, Cairo 11651, Egypt

Corresponding author: Mariam M. Al-Tarabily (mariammokhtar75@hotmail.com)

ABSTRACT The main objective of e-learning systems is to improve the student learning performance and satisfaction. This can be achieved by providing a personalized learning experience that identifies and satisfies the individual learner's requirements and abilities. The performance of the e-learning systems can be significantly improved by exploiting dynamic self-learning capabilities that rapidly adapts to prior user interactions within the system and the continuous changes in the environment. In this paper, a dynamic multi-agent system using particle swarm optimization for the e-learning systems is proposed. The system incorporates five agents that take into consideration the variations in the capabilities among the different users. First, the project clustering agent is used to cluster a set of learning resources/projects into similar groups. Second, the student clustering agent (SCA) groups students according to their preferences and abilities. Third, the student-project matching agent is used to map each learner's group to a suitable project or particular learning resources according to specific design criteria. Fourth, the student-student matching agent is designed to perform the efficient mapping between different students. Finally, the dynamic SCA (DSCA) is employed to continuously track and analyze the student's behavior within the system such as changes in knowledge and skill levels. Consequently, the DSCA adapts the e-learning environments to accommodate these variations. Experimental results demonstrate the effectiveness of the proposed system in providing near-optimal solutions in considerably less computational time.

INDEX TERMS Dynamic environment, e-learning, multi-agent, particle swarm optimization.

I. INTRODUCTION

E-learning systems have become one of the most prevalent teaching methods in recent years. This broad adoption of e-learning presented new potentials as well as new challenges. One of its conventional modes is the blended learning paradigm where learners can access the teaching material asynchronously and collaborate with their colleagues while conveying physical operation in the classroom [1]–[3]. Current research focuses on improving the learning experience in this type of education by introducing innovative tools and methods.

Adapting the e-learning experience to students preferences and needs is an imperative objective of modern e-learning systems. The system should combine the ability to detect the learners' affective skills, knowledge levels, and specific needs in the context of learning to improve the overall learning process. The system should continuously capture and incorporate knowledge of prior tasks within the system as an implicit source of knowledge about the learners.

Various approaches have been proposed to support personalized learning in e-learning systems [4]–[6].

Many studies have considered the development of e-Learning systems by using data mining techniques [7], [8], artificial intelligence (AI) [9]–[11], and fuzzy theory [12], [13]. One of the foremost challenges in e-learning systems is the continuous change in the user characteristics as they interact within the system. Extensive effort has been devoted to developing intelligent e-learning systems to capture the dynamic nature of the learning process [9], [11].

Particle swarm optimization (PSO) is metaheuristic derived from the cooperative intelligence of insect colonies that live and interact in large groups. PSO has been successfully applied to many static optimization problems [14]. Applying PSO to dynamic systems requires the optimization algorithm to not only find the global optimal but also to continuously track changes and adapts the optimal solution accordingly [15]–[17]. A complete reset of the particle's memory is one possible approach to address the changes in the system environment. However, this is inefficient since the whole population has already converged to a small region of the search space and it might not be easy to jump out of likely

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

local optima to track the changes. Several PSO algorithms have been recently proposed to address problems associated with dynamic systems [18]–[20].

Other dynamic tracking algorithms used in this area employ evolutionary programming and strategies [21], [22]. Shi and Eberhart [23] utilized the dynamic tracking procedures with PSO and demonstrated successful tracking of a 10-dimensional parabolic function with a severity of up to 1.0. Carlisle and Dozier [24] used PSO to track dynamic environments with continuous changes. In [25] PSO has been extended to adaptive particle swarm optimization (APSO) which incorporates two main stages. First, the population distribution and particle fitness is evaluated. Second, an elitist learning strategy is performed when the evolutionary state is classified as convergence state.

PSO has shown to have successful applications in the e-learning field. De-Marcos *et al.* [26] employed PSO to solve the learning object (LO) sequencing problem, and then proposed a PSO agent that performs automatic (LO) sequencing. Cheng *et al.* [27] proposed a dynamic question generation system based on the PSO algorithm to cope with the problem of selecting questions from a large-scale item bank. In [6], PSO was utilized to comprise appropriate-learning materials into personalized e-courses for different learners. Ullmann *et al.* [28] developed a PSO-based algorithm to form collaborative groups based on uses level of knowledge and interest in Massive Online Open Courses (MOOCs).

Two main e-learning design issues are considered in this paper. First, the clustering of students or tasks/projects within the system based on their profiles or characteristics. Second, the mapping schema utilized between students and available tasks/projects. A good clustering or mapping schema based on prior knowledge and performance within the system can lead to a significant improvement in the learning process and user satisfaction.

To address the problem of clustering large datasets, many researchers used the well-known partitioning K-means algorithm and its variants [29]–[31]. The main drawbacks of the K-means algorithm are that the selection of the initial cluster centroids considerably affects the clustering results and that it needs a former knowledge of the number of clusters. In recent years researchers have proposed various approaches inspired by biological behaviors for the clustering problem, such as Genetic Algorithm (GA) and Ant clustering [31], [32]. Cui *et al.* [29] presented a hybrid PSO+K-means document clustering algorithm that performed document clustering. Premalatha and Natarajan [32] presented Discrete PSO with crossover and mutation operators that enhanced the performance of the clustering algorithm. Ghali *et al.* [33] investigated a new technique for data clustering using exponential particle swarm optimization (EPSO). The EPSO converged slower to lower quantization error, while the PSO converged faster to a large quantization error. In [34] a new approach to particle swarm optimization (PSO) using digital pheromones is proposed to coordinate swarms within an n-dimensional space to improve the efficiency of the search process. Izakian *et al.* [35] investigated a hybrid fuzzy clustering method based on Fuzzy C-means and fuzzy PSO (FPSO) to gain the benefits of both algorithms.

A multi-agent system (MAS) is a lightly joined network of problem-solvers that work collaboratively to solve complex problems that are beyond the capabilities of the individual solvers [36]–[41]. Several researchers proposed the use of multiple agents' implementation approach to deal with the complicated tasks. This involves dividing the task of into several subtasks and handles these subtasks by employing several software agents [40], [41].

Various attempts to develop MAS for Educational systems were presented in the literature [38], [39], [41]. Pireva and Kefalas [39] proposed that a student in a learning environment should be placed within the framework of the surrounding entities that support the student's access to the learning resources and participation in different learning activities.

In this paper, we present a dynamic multi-agent technique for e-learning systems using PSO (DMAPSO). The objective is to incorporate the intelligence of a multi-agent system in a way that enables it to effectively support the educational processes.

The first two agents are the Project Clustering Agent (*PCA*) and the Student Clustering Agent (*SCA*). The two agents are based on the subtractive-PSO clustering algorithm that is capable of fast yet efficient clustering of projects and students within the e-learning system [44], [45]. The third agent is the Student-Project Matching Agent (*SPMA*). This agent utilizes PSO to recommend appropriate e-learning projects to a particular student group. The mapping is performed based on various design criteria depending on the learner's performance within the system. The fourth agent is the Student-Student Matching Agent (*SSMA*). This agent tracks the student's knowledge, preference, learning style and time availability and maintains a dynamic learner profile. The agent recommends the best matching helpers for collaboration based on PSO. The Fifth agent is the Dynamic Student Clustering Agent (*DSCA*). This agent is used to achieve dynamic student clustering using PSO. *DSCA* substantially enhances the performance of the conventional PSO algorithm to conform to the dynamic environment.

The remainder of this paper is organized as follows: Section II presents an overview of the related work such as PSO and subtractive clustering algorithms. In Section III, the proposed dynamic multi-agent system using PSO (*DMAPSO*) is described. Experimental results are reported in Section IV. Finally, concluding remarks are summarized in Section V.

II. RELATED WORK

A. PARTICLE SWARM OPTIMIZATION

In 1959 Eberhart and Kennedy developed PSO based on the phenomenon of cooperative intelligence inspired by the social behavior of bird flocking [42], [43]. PSO is a

Algorithm 1 Basic PSO Algorithm

Generate the initial swarm;
Evaluate the fitness of each particle;
repeat
  for Each particle i **do**
    Update particle i according to (1) and (2);
    if $f(x_i) < f(x_{pbest_i})$ **then**
      $x_{pbest_i} = x_i$;
      if $f(x_i) < f(x_{gbest})$ **then**
        $x_{gbest} = x_i$;
      end if
    end if
  end for
until the stopping criterion is satisfied

population-based algorithm consisting of a swarm of processing elements identified as particles. Each particle explores the solution space to search for the optimum solution. Therefore, each particle position represents a candidate solution for the problem. When a particle moves to another location, a new problem solution is formed. Each particle compares its current fitness value to the fitness of the best previous position for that particle *pbest* and to the fitness of the global best particle among all particles in the swarm *gbest*. The particle velocity characterizes the position deviation between two consecutive iterations. The velocity and position of the i^{th} particle are updated according to the following equations:

$$v_{id}(t+1) = \omega * v_{id}(t) + c_{1*} rand_1 * (p_{bestid}(t) - x_{id}(t)) + c_2 * rand_2 * (g_{best}(t) - x_{id}(t)), \quad (1)$$

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

For $i = \{1, 2, 3, \ldots, N\}$ and N is the size of the swarm, t is the iteration number, $rand_1$, $rand_2$ are two random real number $\in [0, 1]$. Constants c_1 and c_2 are learning factors that control the weight balance of p_{ibest} and g_{best} during the iterative process. The inertia weight ω balances the local and the global search during optimization process [33], [43]. The performance of the PSO algorithm is enhanced if the inertia is initially set to a large value to stimulate global exploration at the initial stages of the search process. This value should be gradually reduced to acquire more refined solutions as we approach the end of the search process. The inertial weight (ω) is calculated as follows [43]:

$$\omega = (\omega - 0.4)\frac{(MAXITER - t)}{MAXITER} + 0.4, \quad (3)$$

Where *MAXITER* represents the maximum number of iterations, and t is the current iteration. The framework of the basic PSO algorithm is shown in Algorithm 1.

B. DATA CLUSTERING

In most clustering algorithms, the dataset is represented by a set of vectors called the feature vectors [46]. Each feature vector should include proper features to characterize the object. Objects are grouped in the same cluster according to a specific similarity measurement. Therefore, a measure of the similarity between two data sets from the same feature space is essential to most clustering algorithms. The most popular metric to compute the similarity between two data vectors m_p and m_j is the Euclidean distance, given by:

$$dist(m_p, m_j) = \sqrt{\sum_{k=1}^{d_m} \frac{(m_{pk} - m_{jk})^2}{d_m}}, \quad (4)$$

Where d_m is the dimension of the problem is space; m_{pk} and m_{jk} are weight values of the data m_p and m_j in dimension k.

The term "*dist*" is used to quantize the similarity between two data sets from the same feature space. Small "*dist*" values indicate a high similarity level between two objects in the dataset. In the E-learning domain, "*dist*" refers to the deviations between students or assignment/projects to be clustered. The Euclidean distance is a special case of the Minkowski distance [29], represented by:

$$dist_n(m_p, m_j) = \left(\sum_{i=1}^{dm} \left|m_{i.p} - m_{i,j}\right|^n\right)^{1/n}, \quad (5)$$

Cosine correlation measure is another widely used similarity measure in data clustering [31] calculated as follows:

$$Cos(m_p, m_j) = \frac{m_p.m_j}{\|m_p\|\,\|m_j\|}, \quad (6)$$

Where $m_p.m_j$ denotes the dot product of the data vectors and $\| \; \|$ indicates the length of the vector.

C. SUBTRACTIVE CLUSTERING

Subtractive clustering is a simple and effective approach to approximate estimation of cluster centers on the basis of a density measure. In subtractive clustering, each data point is a possible cluster center [44], [45]. Assume the dataset consist of n data points $\{x_1, \ldots, x_n\}$ in the d_m-dimensional search space. A density measure at data point x_i is given as follows:

$$D_i = \sum_{j=1}^{n} exp\left(-\frac{\| x_i - x_j \|^2}{(r_a/2)^2}\right), \quad (7)$$

where r_a is a positive constant which defines the radius of the neighborhood for a specific point. The data point that has the highest number of neighboring points will have the highest density ratio and will be selected as the first cluster center. Let x_{c1} be the point selected and D_{c1} is its corresponding density measure. The density measure D_i for each data point x_i in the following iteration is recalculated as follows:

$$D_i(t+1) = D_i(t) - D_{C1}(t)\, exp\left(-\frac{\| x_i - x_{c1} \|^2}{(r_b/2)^2}\right), \quad (8)$$

where t is the iteration numbers and r_b is a positive constant that defines the neighborhood that has a considerable reduction in the density measure. Consequently, data points close to x_{c1} will have low-density measure and are improbable to be chosen as the next cluster center. In general, constant r_b is usually larger than *ra* to prevent closely-spaced cluster

Algorithm 2 Subtractive Clustering Algorithm

1: Initialize all the n data points;
2: Evaluate the density measure D_i for each x_i according to (7);
3: Select the first cluster center C;
4: **repeat**
5: **for** Each data point x_i **do**
6: recalculate the density measure D_i according to (8);
7: choose the next cluster center;
8: **end for**
9: **until** sufficient number of cluster centers k are produced;

centers. A value of $r_b = 1.5\ r_a$ was suggested in [44]. The framework of the basic subtractive clustering algorithm is shown in Algorithm 2.

D. SUBTRACTIVE- PSO CLUSTERING ALGORITH

The subtractive-PSO clustering algorithm initially proposed in [45] includes two main modules: the subtractive clustering module and the PSO module. Initially, the subtractive clustering module predicts the optimal number of clusters and estimates the initial cluster centroids. Subsequently, the preliminary information is conveyed to the PSO module for refining and generating the final clustering solution. Each particle in the swarm represents a candidate solution for clustering the dataset. Each particle i maintains a position matrix $x_i = (C_1, C_2, \ldots, C_i, .., C_k)$, where C_i is the i^{th} cluster centroid vector and k is the total number of clusters. Each particle iteratively updates its position matrix based on its own experience (x_{pbest_i}) and the experience of its neighboring particles (x_{gbest}). The search process is guided by a fitness value to assess the quality of the solution represented by each particle. The average distance between the data objects and their corresponding cluster centroids is used as the PSO fitness function.

III. PROPOSED DYNAMIC MULTI-AGENT SYSTEM USING PSO (DMAPSO)

The main objective of the proposed *DMAPSO* is to enhance the performance of collaborative e-learning systems. In order to adapt the learning process according to the needs and preferences of each user, the system should maintain a databank of the learner profiles to be used in subsequent agents of the system. The learner profile integrates both explicit user demographic information and preferences with implicit information gathered thru assessment of prior system tasks/projects. The learner profile should be adaptive in the sense that it should capture the dynamic nature of the learning process. Similarly, the system maintains a databank of the available task/project profiles.

Five attributes are used to characterize each learner profile whereas four attributes are used for each task/project. For the learner profile, the five attributes are the proficiency (difficulty) level of student, the weight of association between the student and each topic, availability time, number of completed tasks/projects, and the exposure frequency of the student. Consider an e-learning system with S students. Each student s_r $(1 \le r \le S)$ has a specific difficulty level D_r and availability time (t_r). Assume that M topics are to be taught through the system. Each topic (c_j), $(1 \le j \le M)$ has its specialized learning objectives. Each student has a different knowledge level for the different topics quantified by the weight value ws_j assigned by the instructor. Additionally, the system records the exposure frequency of the student fs_r which is the number of times the user was designated as an assistant/helper by another student.

For task/projects profiles, the four attributes are the difficulty level of each project, the weight of association between the project and each topic, the average projected time for completing the project $t(p_m)$, and the exposure frequency of the project. Assume we have P projects, $1 \le m \le p$ with a specific difficulty degree d_m. Each project is relevant to each topic with different weight wp_j. Additionally, the system records the exposure frequency of each project fp_m which defines the frequency of selection of the project by the students.

Fig. 1 presents the framework of the dynamic multi-agent system using PSO (*DMAPSO*). Figs. 1(a-e) present an illustration of the *PCA*, *SCA*, *SPMA*, *SSMA* and *DSCA* agents, respectively. The five agents are explained in more detail in the following sections.

Algorithm 3 Project Clustering Agent

1: Initialize all the P projects in the dm-dimensional space;
2: Subtractive clustering;
3: Generate swarm with cluster centroid vectors CP and the number of clusters kp into the particles as an initial see;
4: **while** stopping criteria is not satisfied **do**
5: **for** each *P-particle i* **do**
6: Assign each project vector in the data set to the closest centroid vector using (4);
7: Calculate the fitness value f according to (9);
8: Local Search( );
9: **end for**
10: **end while**

A. PROJECT CLUSTERING AGENT (PCA)

The main objective of this agent is to cluster the available projects into homogenous groups based on their attributes. The *PCA* architecture is shown in Fig. 1(a).

In the *PCA* the subtractive-PSO clustering algorithm is utilized to perform fast clustering of the projects according to their level of difficulty and the degree of similarity between their topics attributes [45]. First, the subtractive clustering module estimates the optimal number of clusters and the initial cluster's centroid locations. Next, this information is sent to the PSO module for generating the final optimal clustering results as shown in Algorithm 3. Each particle is represented by a matrix $X_p = (Cp_1, Cp_2, \ldots, Cp_l, .., Cp_{kp})$, where Cp_l represents the l^{th} project cluster centroid vector

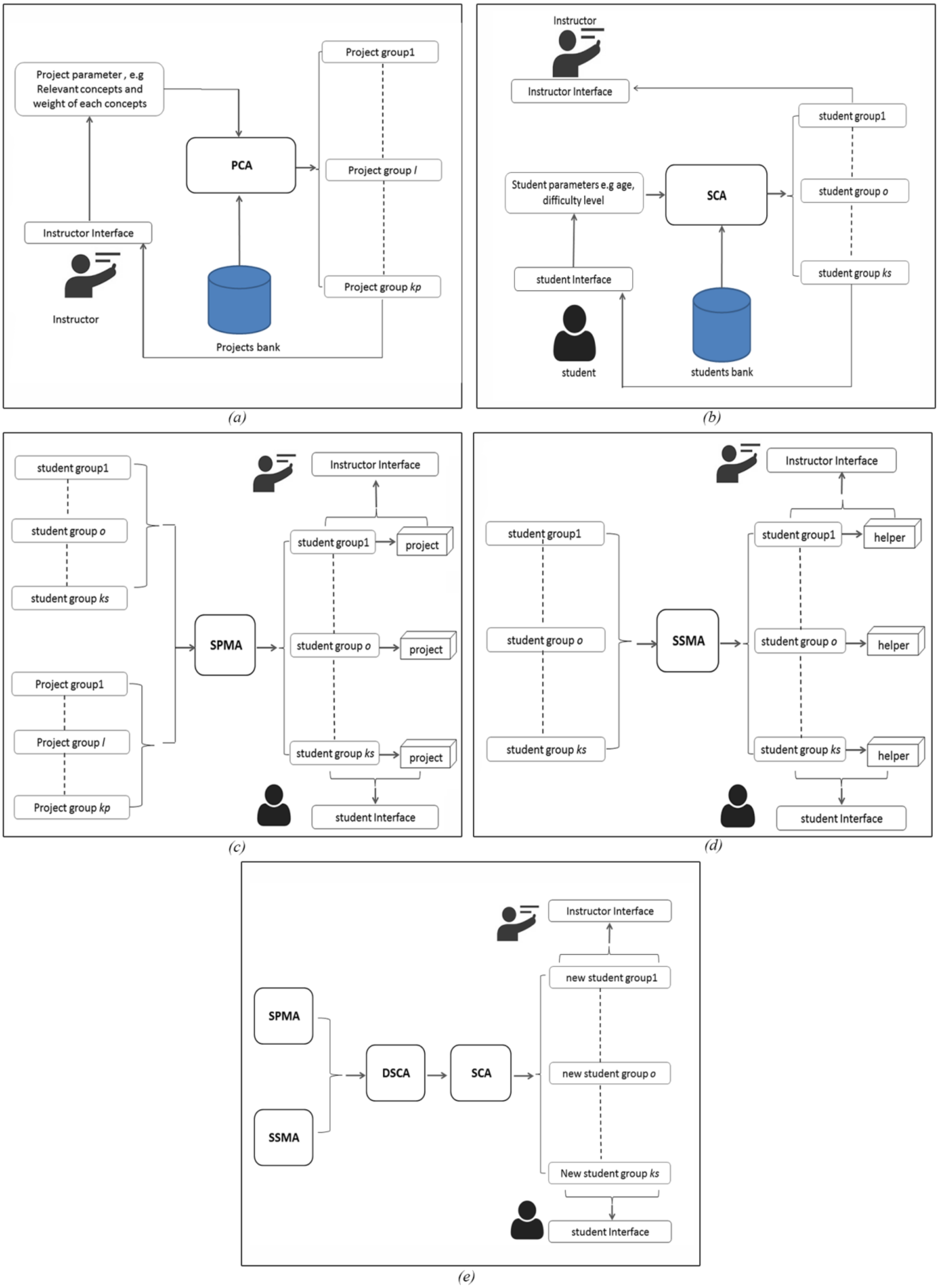

FIGURE 1. **Architecture of the proposed DMAPSO system, (a) PCA architecture, (b) SCA architecture, (c) SPMA architecture, (d) SSMA architecture, (e) DSCA architecture.**

and kp represents the number of project clusters. The fitness function is represented by the equation below:

$$f = \frac{\sum_{l=1}^{kp}\left\{\frac{\sum_{m=1}^{a_l} d(Cp_l, p_{lm})}{a_l}\right\}}{kp}, \tag{9}$$

where p_{lm} represents the m^{th} project that belongs to cluster l, Cp_l denotes the centroid vector of l^{th} cluster, $d(Cp_l, p_{lm})$ is the distance between project p_{lm} and the cluster centroid Cp_l, a_l is the number of projects that belong to cluster l.

Algorithm 4 Local Search Algorithm

for each particle i **do**
  Update particle i according to (1) and (2);
  iteration = iteration+1;
  if particle i is better than p_{besti} **then**
    Update p_{besti};
    if particle i is better than g_{best} **then**
      Update;
    end if
  end if
end for

B. STUDENT CLUSTERING AGENT (SCA)

Clustering learners according to their abilities is vital to help them attain their optimum performance and increase their motivation to learn. Nonhomogeneous student placement in groups may result in providing less assistance to weak students, obstructing the advancement of excellent students and increase the instructor's workload. The student profile is used to gather and analyze student abilities and characteristics. Each student profile is represented by static and dynamic attributes. Static attributes are demographic information such as name, age, etc. collected from the user through a questionnaire or a registration form. Dynamic attributes are parameters associated with learner's interaction with the system, such as the proficiency level, number of finished projects, etc.

The *SCA* performs two main tasks. The first task is to cluster students into homogenous groups to maximize the collaboration of the members within each cluster. This allows students to better achieve their learning goals and objectives. The second task is to call the DSCA agent when a change is detected in the e-learning environment. The architecture for the *SCA* is shown in Fig. 1(b).

Similar to *PCA*, *SCA* uses the subtractive-PSO clustering approach [45] for making quick and intelligent student clustering as shown in Algorithm 5. Each particle has a matrix $X_s = (Cs_1, Cs_2, \ldots, Cs_o, .., Cs_{ks})$, where Cs_o denotes the o^{th} cluster centroid vector and ksrepresent the number of student clusters. The fitness value is represented by the equation below:

$$f = \frac{\sum_{o=1}^{ks}\left\{\frac{\sum_{r=1}^{a_o} d(Cs_o, s_{or})}{a_o}\right\}}{ks}, \tag{10}$$

where s_{or} stands for the r^{th} student, which belongs to cluster o, Cs_o represents the centroid vector of o^{th} cluster, $d(Cs_o, s_{or})$ denotes the distance between student s_{or} and the cluster centroid Cs_o, a_o represents the number of students that belongs to cluster o.

C. STUDENT-PROJECT MATCHING AGENT (SPMA)

The *SPMA* is used to match appropriate e-learning projects/learning resources to the student groups depending on various design criteria. The different project and student clusters generated from the *PCA* and *SCA* are used as the inputs for this agent. The main function of this agent is to map projects with specific difficulty levels to suitable student groups based on the student's average ability level. The average ability of the students depends on the scores of prior contributions in the system. This includes projects that the student has successfully completed and whether the time taken to finish the projects matches its estimated finish time. The *SPMA* is described in algorithm 7 and the *SPMA* architecture is shown in Fig. 1(c).

Algorithm 5 Student Clustering Agent

Initialize all the students S in the dm -dimensional space;
Subtractive clustering;
Generate swarm with cluster centroid vectors CS and the number of clustersks into the particles as an initial seed;
while stopping criteria is not satisfied **do**
  for each *S-particle* i **do**
    Assign each student vector to the closest centroid vector using (4);
    Calculate the fitness value f according to (10);
    Local Search;
  end for
  Detect Change( );
end while

Algorithm 6 Detect Change Algorithm

Re-evaluate the global best particle over all particles;
if T **then**
  he fitness of the re-evaluated position change
  Save the *gbest* of the swarm;
  Dynamic Student Clustering Agent( );
end if

The selection probability of a particular project group to be assigned to a specific student group is based on a selection rule. The rule provides a high selection probability to the project group that has a close average difficulty to the student's average difficulty level. In particular, the selection probability of project group $pgrp_l$ is to be assigned to student group $sgrp_o$ is defined as follows:

$$prob_l = min_{o=1\sim ks}\{|\overline{d}_l - \overline{D}_0|\}, \tag{11}$$

Where $\overline{d_l}$ represents the average difficulty level of all the projects belonging to the same group $l(1 \leq l \leq kp)$. $\overline{D_0}$ represents the average difficulty level of all the students in group o, $(1 \leq o \leq ks)$.

The fitness function of *SPMA* is described as follows:

$$f(P_m) = C_1 + C_2 + C_3, \tag{12}$$

The fitness function consists of three main components C_1, C_2, and C_3 defined as follows:

$$C1 = \min_{o=1\sim ks} |d_{ml} - \overline{D_o}|, \quad 1 \leq m \leq P \tag{13}$$

Algorithm 7 Student-Project Matching Agent

```
for each sgrp_o, o = 1 to ks do
    Calculate the average knowledge weight \overline{ws} for all students in sgrp_o;
    Choose the pgrp_l which has the min. prob_l;
    Initialize all the projects p_ml in pgrp_l in the dm-dimensional space;
    while stopping criteria is not satisfied do
        for each particle i do
            Calculate the fitness value f according to (12);
            Local Searc;
        end for
        Choose p_m with the minimum fitness;
    end while
end for
```

Algorithm 8 Student-Student Matching Agent

```
for each student s̀_r in group sgr̀p_o for project p_m do
    Calculate the average experience \overline{h_rm} for all students for project p_m;
    Choose the sgrp_o which has the max Sprob_o;
    Initialize all the students s_r in sgrp_o;
    while stopping criteria is not satisfied do
        for each particle i do
            Calculate the fitness value f according to (17);
            Local Search;
        end for
        Choose s_r with the minimum fitness;
    end while
end for
```

C_1 is an indicator of the difference between the degree of difficulty of each project p_{ml} in the selected group l and the average difficulty level of the students in the same group.

$$C_2 = \min_{o=1\sim ks} |\overline{ws_o} - wp_{ml}|, \tag{14}$$

C_2 is an indicator of the difference between the degree of relevance of each project p_{ml} in the selected group l and the average knowledge level of the students in the same group.

$$C_3 = \min_{m=1\sim p} \frac{fp_{ml}}{max(fp_{1l,\ldots\ldots,}fp_{ml},\ldots\ldots\ldots fp_{pl})}, \tag{15}$$

C_3 represents the exposure frequency of project p_{ml} in cluster l.

Once a student group successfully completes the assigned project p_m within its expected completion time $t(p_m)$, the dynamic attributes for each student in the group are updated. The student performance in the most recent system interaction is reflected in student and project attributes such as difficulty level and the number of accomplished projects for the students and the exposure frequency for the selected project.

D. STUDENT-STUDENT MATCHING AGENT (SSMA)

The *SSMA* tracks the student's knowledge, preferences, learning style and time availability and maintains a dynamic learner profile. The agent recommends the best matching helpers for collaboration based on PSO. Consider that student $\grave{s}_r \in sg\grave{r}p_o$ have a question about project (p_m), the agent will suggest a helper student s_r from another group $sgrp_o$ who is available at the same time slot. The *SSMA* recommends the student with the minimum exposure frequency and with high knowledge about project p_m. The *SSMA* is described in algorithm 8 and the *SSMA* architecture is shown in Fig. 1(d).

Each student profile (s_r) maintains the number of times that the student completed project p_m successfully (h_{rm}) and the time slots in which the student is available (t_r). The selection probability of a particular students group is based on the selection rule which gives a higher selection probability to the group that has higher previous knowledge for project p_m. In particular, the selection probability of student group $sgrp_o$ is defined as:

$$Sprob_o = max_{o=1\sim ks}\,\overline{h_{rm}}, \tag{16}$$

Where $\overline{h_{rm}}$ is the average number that students in $sgrp_o$ that completed project p_m. Once the group selection process is complete, *SSMA* has to choose the best available helper s_r among the members of $sgrp_o$.
The fitness function of *SSMA* is calculated as follows:

$$f(S_r) = C_4 + C_5 + C_6, \tag{17}$$

The fitness function consists of three main components C_4, C_5, and C_6 defined as follows:

$$C_4 = min_{r=1\sim s}\,|1 - norm(h_{rm})|, \tag{18}$$

C_4 indicates the number of times that a student $s_r \in sgrp_o$ completed project p_m.

$$C_5 = \min_{r=1\sim s} |st_r - \grave{st}_r|, \tag{19}$$

C_5 represents the deviation between the available time slots of students s_r and $\grave{s}_r$.

$$C_6 = \min_{r=1\sim s} \frac{fs_{ro}}{\max\,(fs_{1o,\ldots\ldots,}fs_{ro},\ldots\ldots\ldots fs_{so})}, \tag{20}$$

C_6 represents the exposure frequency of student s_r. After *SSMA* matches a suitable helper s_r for each $\grave{s}_r$, the dynamic attributes for students will be updated.

E. DYNAMIC STUDENT CLUSTERING AGENT (DSCA)

The function of this agent is to efficiently re-cluster the students when changes in student information are perceived. The *DSCA* is described in algorithm 9 and the *DSCA* architecture is shown in Fig. 1(e).

The *DSCA* agent incorporates two new parameters that enable the automatic control of the algorithmic parameters to improve the search efficiency and convergence speed through the different stages of the search process. The first factor is the dynamic factor (α) which controls the number of particles that will reset their position vector periodically to the current positions, thus forgetting their experiences to that point, this process is different from restart the particles in that the particles, in retaining their current location, have retained the profits from their relationship to the goal at that point. The second factor is called gradual reset factor (β), which makes the gradual reset. This means that the reset value will not be the same for all particles. Particles which are farthermost from *gbest* are more likely to change their positions compared to other particles.

Algorithm 9 Dynamic Student Clustering Agent

for each *S-particle i* **do**
    Calculate the distance between each *S-particle i* and best *S-particle* in the swarm (g_{best}) and construct a distance matrix $dist(i, g_{best})$;
    Calculate the dynamic factor α according to (22);
    Calculate the number of particles that will reset its position vector *(num)* according to (23);
    Calculate gradual reset factor β_i according to (24);
    Reset the x_{pbest_i} for (*num*) *S-particles* which are the furthermost from g_{best} according to (25);
    Adjust the number of iterations *itnew* according to (26);
    Student Clustering Agent( );
end for

In *DSCA* the distance between each *S-particle i* and the global best solution (g_{best}) in the *dm*-dimensional space is calculated by the Euclidean distance initially described in (4) as follows:

$$dist(i, g_{best}) = \sqrt{\sum_{k=1}^{d_m} \frac{\left(x_{ik} - x_{gbestk}\right)^2}{d_m}}, \tag{21}$$

Consider an e-learning system with S students. Each student s_r $(1 \leq r \leq S)$ is represented by a set of vectors $S = \{x_1, x_2, \ldots, x_{dm}\}$, where each x_i is a feature vector. The dynamic factor α is calculated as follows:

$$\alpha = norm \sum_{r=1}^{S} (\sum_{i=1}^{dm} |S_{newi} - S_i|), \tag{22}$$

Where S_{newi} is the new student's feature vector after the changes have been detected, S_i is the students's feature vector before the last changes in the system.

The number of particles that will reset its position vector (*num*) is given by:

$$num = \alpha^* N, \tag{23}$$

where N is the total number of particles in the swarm.

The gradual reset factor (β_i) for each particle is calculated as follows:

$$\beta_i = dist(i, g_{\text{best}})/maxdist, \tag{24}$$

Where *maxdist* is the distance of the farthermost particle from the *gbest*.

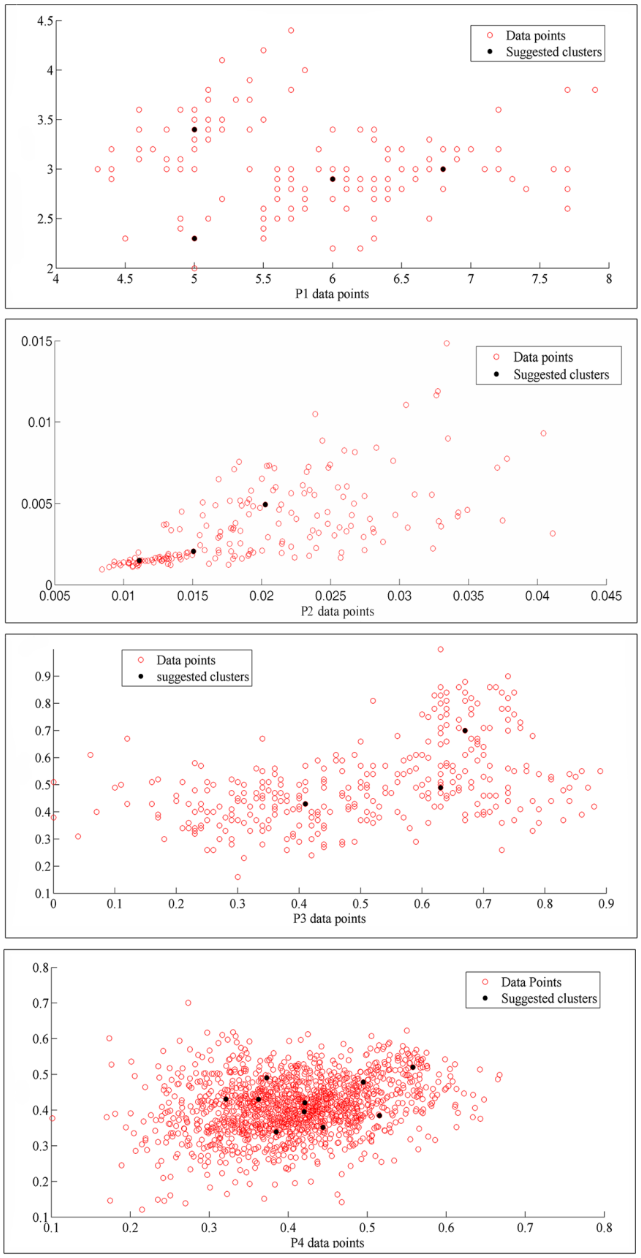

FIGURE 2. Cluster results obtained by the subtractive clustering algorithm.

The particles will reset their p_{best} according to the following formula:

$$x_{pbest_i} = x_i * \beta_i, \tag{25}$$

Where x_i is position of the i'^{th} particle in the swarm.

The new iteration number will be adjusted also according to the dynamic factor α as follow

$$itnew = round(MAXITER * \alpha), \tag{26}$$

Where *MAXITER* is the maximum number of iterations used in algorithm 5.

TABLE 1. Descriptions of the "project" and "student" banks.

Item bank	Number of instances	Number of attributes	Number of classes	Average difficulty
P_1	150	5	4	0.593
P_2	180	14	3	0.555
P_3	330	8	3	0.572
P_4	1500	8	10	0.548
S_1	350	34	7	0.557
S_2	400	35	7	0.526
S_3	1450	10	10	0.568
S_4	4200	8	3	0.552

TABLE 2. Performance of the subtractive, PSO, and subtractive-PSO clustering algorithms.

Item bank	Number of instances	Fitness value		
		Subtractive clustering	PSO clustering	Subtractive-PSO clustering
P_1	150	0.612	0.6891	0.3861
P_2	180	2.28	2.13	1.64
P_3	330	0.182	0.1781	0.1313
P_4	1500	1.30	1.289	0.192
S_1	350	0.078	0.0777	0.0725
S_2	400	0.0391	0.0363	0.0334
S_3	1450	0.02	0.019	0.0103
S_4	4200	0.0273	0.0219	0.0187

TABLE 3. Percentage error of the subtractive, PSO, and subtractive-PSO clustering algorithms.

Item bank	% Error		
	Subtractive clustering	PSO clustering	Subtractive-PSO clustering
P_1	1.2	0.4	0.2
P_2	2.6	0.8	0.3
P_3	3.6	1.3	0.7
P_4	15.3	5.2	2.4
S_1	4.8	1.2	0.2
S_2	3.7	1.5	0.6
S_3	14.8	4.1	2.7
S_4	19.3	15.8	3.4

IV. EXPERIMENTAL RESULTS

Several groups of experiments were performed to evaluate the performance of the proposed *DMAPSO* algorithm. The objective of the first group of experiments is to investigate the efficiency of the proposed *PCA* and *SCA* algorithms. In the second and third groups, the performances of the *SPMA* and *SSMA* are compared with competing approaches. Finally, in the fourth group, the performance of the *DSCA* is evaluated. The experiments examine the effect of the key design parameters and compare the performance of *DSCA* to other algorithms in the dynamic environment.

The proposed algorithm and the comparative algorithms were implemented using MATLAB and experiments were run on an Intel i7-4702MQ 2.2 GHz CPU with 16 GB of RAM using 64-bit implementations to ensure maximum utilization of the hardware.

PSO parameters were chosen experimentally in order to get an adequate solution quality in the minimal time span. Different parameter combinations from the PSO literature were tested [23], [33]. During the preliminary experiment, four swarm sizes (N) of 10, 20, 50, and 100 particles were chosen to test the algorithm. The outcome of $N = 20$ was superior and used for all further experiments. The maximal number of iterations was set to 200. The inertia weight (ω) is calculated according to (3). Learning parameters c_1 and c_2 were set to 1.49.

The *percentage error* performance metric is used to assess the overall clustering or matching results. The *percentage error* is calculated as follows:

$$\text{Percentage error} = \frac{\text{Number of incorrectly clusterd instances}}{\text{Total number of instances}} * 100 \quad (27)$$

TABLE 4. Fitness values of the *SPMA*, *RSFS* and the exhaustive search algorithms.

Student- project pairs	*SPMA*	*RSFS*	Exhaustive search
	Fitness function		
S_1-P_1	0.081	0.092	0.072
S_1-P_2	0.0922	0.098	0.092
S_1-P_3	0.099	0.17	0.098
S_1-P_4	0.187	0.198	0.187
S_2-P_1	0.094	0.18	0.095
S_2-P_2	0.097	0.099	0.097
S_2-P_3	0.18	0.21	0.017
S_2-P_4	0.191	0.32	0.19
S_3-P_1	0.185	0.191	0.185
S_3-P_2	0.186	0.192	0.186
S_3-P_3	0.1867	0.21	0.1866
S_3-P_4	0.295	0.435	0.295
S_4-P_1	0.356	0.463	0.356
S_4-P_2	0.391	0.62	0.39
S_4-P_3	0.48	0.578	0.46
S_4-P_4	0.467	0.72	0.46

TABLE 5. Error and time measurements of the *SPMA*, *RSFS* and the Exhaustive search algorithms.

Student – project pairs	*SPMA*		*RSFS*		Exhaustive search	
	% Error	Average time (sec)	*% Error*	Average time (sec)	*% Error*	Average time(sec)
S_1-P_1	0.1	17	7.4	17	0.1	19
S_1-P_2	0.3	17	8.4	17	0.2	20
S_1-P_3	0.2	19	10.7	19	0.4	22
S_1-P_4	1.2	24	10.8	25	1.2	29
S_2-P_1	1.1	18	9	16	1.2	21
S_2-P_2	1.6	18	10.7	17	1.5	21
S_2-P_3	1.5	19	10.2	19	1.3	24
S_2-P_4	2.1	24	11.9	24	2	30
S_3-P_1	1.3	21	10.2	20	1.3	27
S_3-P_2	1.8	21	11.7	21	1.6	27
S_3-P_3	1.6	23	11.9	22	1.4	28
S_3-P_4	2.8	26	15.2	26	2.8	38
S_4-P_1	2.4	29	12	28	2.2	49
S_4-P_2	2.6	29	13.8	29	2.4	50
S_4-P_3	3.2	30	16.2	30	3	52
S_4-P_4	4.1	35	17.4	35	4.2	60

TABLE 6. Fitness values of the *SSMA*, *RSFS* and the exhaustive search algorithms.

Student – Student pairs	*SSMA*	*RSFS*	*Exhaustive search*
	Fitness Function		
S_1	0.1834	0.607	0.181
S_2	0.1689	0.622	0.159
S_3	0.1649	0.548	0.169
S_4	0.1458	0.563	0.144

Any student/project object that has a distance to its corresponding cluster center greater than a predefined threshold is considered as incorrectly clustered. Due to the non-deterministic nature of the PSO algorithm and to ensure result consistency, 10 independent runs for each problem instance were performed and the average

TABLE 7. Error and time measurements of the *SSMA*, *RSFS* and the Exhaustive searchalgorithms.

Student – students pairs	*SSMA*		*RSFS*		Exhaustive search	
	% Error	Average time(sec)	% Error	Average time (sec)	% Error	Average time(sec)
S_1	0.4	15	7.1	15	0.3	18
S_2	0.3	17	6.2	17	0.5	20
S_3	0.6	20	20.3	21	0.8	23
S_4	1.2	22	25	23	0.9	26

TABLE 8. Fitness values of the student datasets.

Student's bank	Response	Min. Fitness	Mean	Std.	Range
S_1 bank	No-change	0.07018	0.07309	0.02704	0.3055
	Re-randomize 15% of particles	0.06918	0.1357	0.04787	0.264
	Re-randomize all particles	0.1574	0.208	0.06998	0.2302
	DSCA algorithm	0.06825	0.07514	0.02298	0.277
S_2 bank	No-change	0.01898	0.01999	0.002975	0.01025
	Re-randomize 15% of particles	0.1737	0.1001	0.06788	0.1547
	Re-randomize all particles	0.1467	0.1126	0.047	0.1271
	DSCA algorithm	8.227e-005	0.04138	0.05849	0.5953
S_3 bank	Do nothing	0.007529	0.00769	0.0005384	0.002043
	Re-randomize 15% of particles	0.0214	0.01566	0.004665	0.01422
	Re-randomize all particles	0.02299	0.01932	0.004873	0.01342
	DSCA algorithm	1.288e-006	0.0104	0.01045	0.04733
S_4 bank	No-change	0.01972	0.02012	0.004386	0.05347
	Re-randomize 15% of particles	0.01746	0.01801	0.003818	0.0582
	Re-randomize all particles	0.1214	0.0985	0.03076	0.09471
	DSCA algorithm	0.0136	0.1057	0.0412	0.117

fitness value was recorded to ensure meaningful results.

A. EXPERIMENT 1

The aim of this experiment is to investigate the quality of the solutions obtained from the *PCA* and *SCA* algorithms based on the attained fitness values. Four project banks with a number of projects ranging between 150 and 1500 were constructed. Similarly, four student banks with a number of students ranging from 350 to 4200 were tested. Table 1 illustrates the characteristics the student and project banks.

The fitness functions given in (9) and (10) are used to quantify the clustering quality. Table 2 compares the student and projects clustering results obtained by the subtractive clustering, PSO clustering, and the subtractive-PSO clustering algorithms. In each experiment, the PSO and the subtractive-PSO clustering algorithms are run for 200 iterations. Results reported in Table 2 demonstrate that the subtractive-PSO clustering approach generates clustering result with the lowest fitness value for all eight datasets.

Fig. 2 displays the cluster centers obtained by the subtractive clustering algorithm which is used subsequently as the seed for the PSO algorithm. Fig. 3 presents the relation between the convergence rate and the number of items in the object bank for the three algorithms. Fig. 3 demonstrates that the subtractive-PSO algorithm yields the best fitness values across the various-size objects banks.

Table 3 shows the percentage error of the different algorithms for the eight data sets. The percentage error of the subtractive clustering algorithm ranges between 1.2 and 19.3. The error of the PSO clustering algorithm ranges between 0.4 and 15.8. The error of the subtractive-PSO clustering algorithm ranges between 0.2 and 3.4.

B. EXPERIMENT 2

The objective of this experiment is to evaluate the performance of the proposed *SPMA* algorithm. A series of experiments has been conducted to compare the execution time and the solution quality of the *SPMA*, Random Selection with Feasible Solution (*RSFS*), and Exhaustive search.

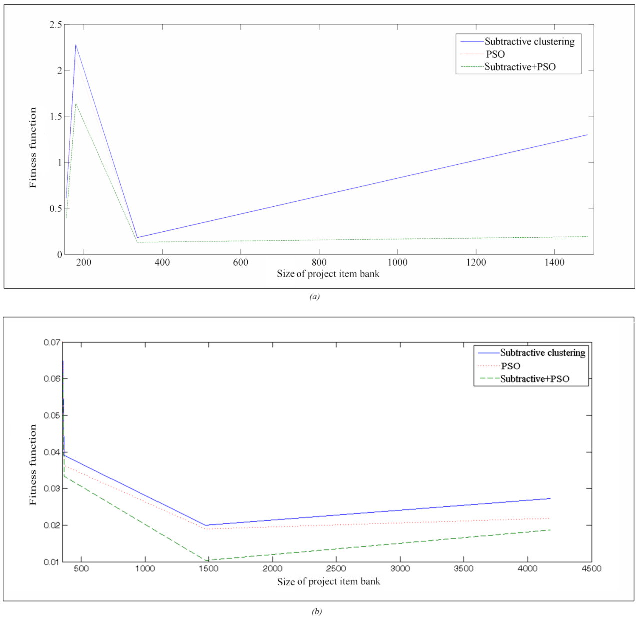

FIGURE 3. (a) Variation of the fitness function for project item banks (b) Variation of the fitness function for student item banks.

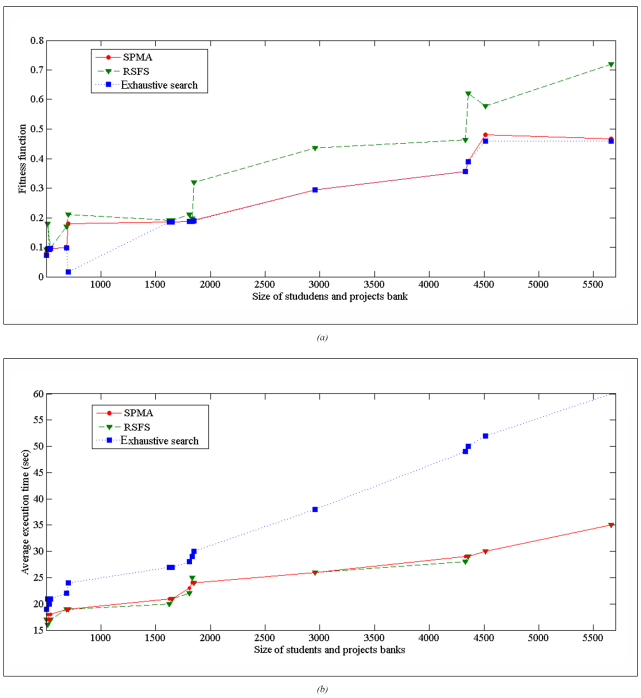

FIGURE 4. (a) Fitness values (b) Average execution time of the SPMA, RSFS and the Exhaustive search algorithms.

The *RSFS* generates a random mapping between the student groups and the projects subject to the specified design constraints. On the other hand, the Exhaustive search examines every feasible combination to find the optimal solution. Sixteen student- project combinations were examined. The fitness function $f(P_m)$ given in (12) is used to quantify the quality of the obtained solution.

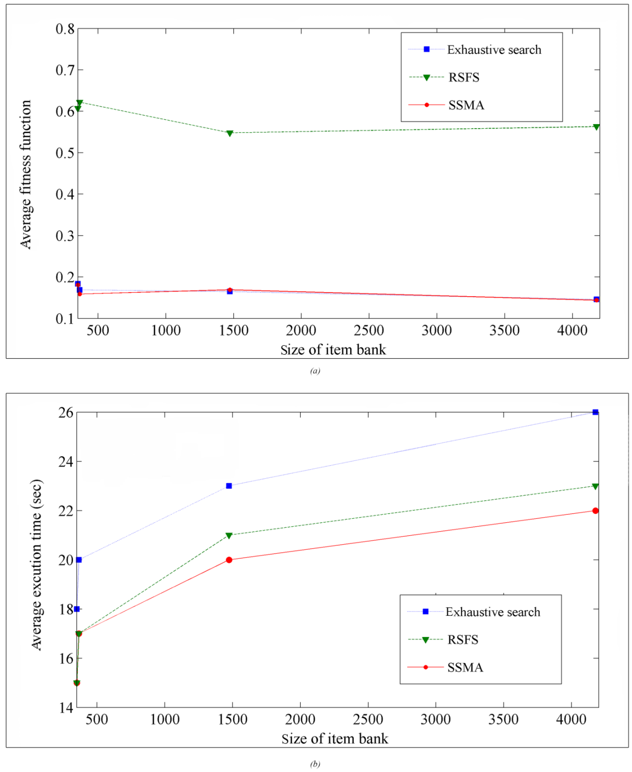

FIGURE 5. (a) Fitness values (b) Execution time of the SSMA, RSFS, and the Exhaustive search algorithms.

Table 4 presents the fitness values $f(P_m)$ obtained using the three algorithms. We observe that the *SPMA* yield optimal/near optimal solutions in all test instances. Table 5 shows the percentage error and the execution time of the three algorithms for all dataset pairs. *SPMA* and Exhaustive search obtain similar percentage error values. However, Exhaustive search requires more execution time compared to *SPMA*, especially for large-scale banks. *RSFS* algorithm shows an execution time similar to *SPMA* but with the highest values of percentage error.

Fig. 4(a) shows that the average fitness values obtained by *SPMA* were similar to the optimal solutions obtained by the Exhaustive search and significantly better than the values obtained by *RSFS*. Fig. 4(b) shows that the average execution time of the proposed system was similar to that of *RSFS* and is significantly less than the time required by the Exhaustive search, particularly for large data banks.

C. EXPERIMENT 3

The objective of this experiment is to evaluate the performance of *SSMA*. In this experiment, the execution time and fitness values of three approaches, *SSMA*, Exhaustive search, and *RSFS* were compared. Table 6 presents the fitness value of each student's bank. Table 7 shows the percentage error and time measurements of all datasets using *SSMA*, *RSFS*, and the Exhaustive search. *SSMA* and Exhaustive search yield less percentage error than *RSFS*. However, Exhaustive search

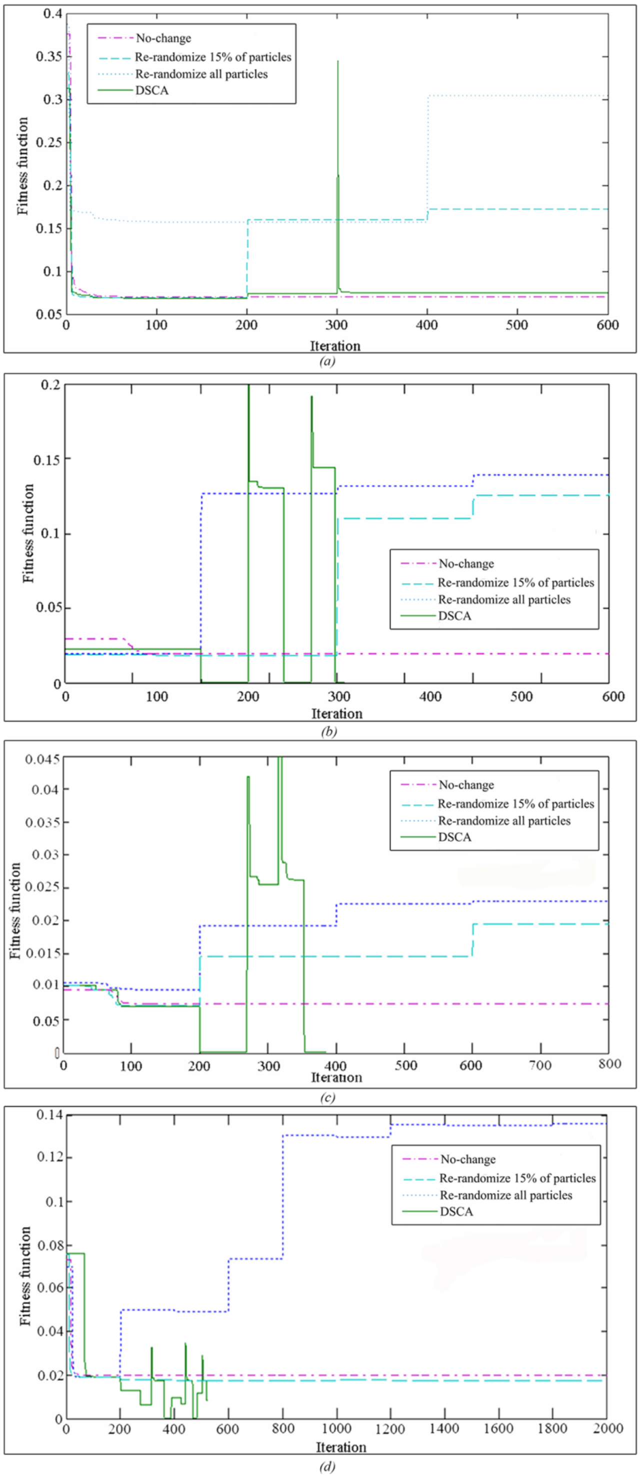

FIGURE 6. Convergence rate of student banks (a) S_1 bank, (b) S_2 bank, (c) S_3 bank, (d) S_4 bank.

requires a significantly longer execution time than *SSMA* and RSFS.

Fig. 5(a) shows the average execution time for each algorithm, the *SSMA* is much more efficient than *RSFS*, particularly when dealing with the large scale item banks. Fig. 5(b) indicates that the average best fitness values obtained by *SSMA* were very close to the optimal solutions obtained by the Exhaustive search.

D. EXPERIMENT 4

The objective of this experiment is to evaluate the performance of the *DSCA* once a change is detected in the student's attributes by *SPMA* or *SSMA*. The experiment compares four techniques to handle the variations in the dynamic environment; no change is performed, re-randomize 15% of particles, re-randomize all particles and *DSCA*. The fitness values attained by the *DSCA* and the different re-randomization methods are presented in Table 8. The *DSCA* clustering approach generates the clustering result that has the minimal fitness values across all the datasets as shown in Table 8. For example, in the S_4 data set, the mean, standard deviation, and range values indicate that the*DSCA* adapts to the changes rapidly and yield the min. fitness values. For all datasets, using the PSO algorithm without any modification obtains the lowest mean and standard deviation values and the largest range because it tapped in Local optima and it didn't adapt to the changes in the environment. Re-randomize 15% of particles gives us better results than the no-change PSO, especially for small datasets. However, it fails to adapt to the changes for large data set such as trapping in local optima in the S_4 data set. Randomization of all particles is not efficient since it starts a new search process regardless of the dynamic change. This causes an increase in the mean, standard deviation, and range without an improvement in the solution quality.

Fig. 6 presents the convergence rate of the various student banks. The changes in the e-learning environment increase with the increase in the size of the dataset. the *DSCA* was the best in tracking and adapting to the dynamic changes in the environment. *DSCA* reaches to the optimal value after 350 iterations for S_1 data set, 420 iterations for S_2 data set, 370 iterations for S_3 data set and 550 iterations for S_4 data set.

V. CONCLUSIONS

In this paper, a new dynamic multi-agent system using PSO (*DMAPSO*) to optimize the performance of e-learning systems is proposed. The system incorporates five intelligent agents that enable the system to effectively improve the educational processes.

The first two agents are the Project Clustering Agent (*PCA*) and the Student Clustering Agent (*SCA*). The two agents are based on the subtractive-PSO clustering algorithm that is capable of fast, yet efficient clustering of projects and students within the e-learning system. The third agent is the Student-Project Matching Agent (*SPMA*). This agent utilizes PSO for mapping appropriate e-learning projects/material to the student's group dynamically according to various design criteria. The fourth agent is the Student-Student Matching Agent (*SSMA*). This agent tracks the student's level of knowledge, learning style, time availability and maintains a dynamic learner profile. The acquired information is then used to recommend the best available helpers for collaboration based on PSO. The fifth agent is the Dynamic Student Clustering Agent (*DSCA*). This agent is used to achieve dynamic clustering of students using PSO. The *DSCA* incorporates two new parameters, a dynamic factor (α) and gradual reset factor (β). First, the dynamic factor regulates the number of particles that will reset their position vector periodically to the current positions omitting their private experiences up to that point. Second, the gradual reset factor is used to perform gradual particle reset. Particles that are the farthermost from *gbest* are more likely to change their positions compared to other particles.

To evaluate the performance of the proposed system, four groups of experiments were carried out. The objective of the first experiment is to investigate the efficiency of the proposed *PCA* and *SCA* algorithms. Experimental results show that subtractive-PSO algorithm presents efficient clustering results in comparison with conventional PSO and subtractive clustering algorithms. In the second and third experiments, the performance of *SPMA* and *SSMA* are compared to competing approaches. Finally, the fourth experiment evaluates the performance of *DSCA*. The performance of *DSCA* is compared to several techniques in the dynamic environment. Experimental results demonstrate that the proposed agents yield optimal or near-optimal results within reasonable execution time.

REFERENCES

[1] A. Rosen, *e-Learning 2.0: Proven Practices and Emerging Technologies to Achieve Real Results*. New York, NY, USA: American Management Association, 2009.

[2] E. Pontes, A. Silva, A. Guelfi, and S. Kofuji, *Methodologies, Tools and New Developments for E-Learning*. Rijeka, Croatia: InTech, Feb. 2012, doi: 10.5772/2468.

[3] C.-T. Yang and H.-C. Ho, "A shareable e-learning platform using data grid technology," in *Proc. IEEE Int. Conf. e-Technol., e-Commerce e-Service*, Mar./Apr. 2005, pp. 592–595.

[4] C.-M. Chen, H.-M. Lee, and Y.-H. Chen, "Personalized e-learning system using item response theory," *Comput. Educ.*, vol. 44, no. 3, pp. 237–255, Apr. 2005.

[5] M.-J. Huang, H.-S. Huang, and M.-Y. Chen, "Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach," *Expert Syst. Appl.*, vol. 33, no. 3, pp. 551–564, Oct. 2007.

[6] C.-P. Chu, Y.-C. Chang, and C.-C. Tsai, "PC^{2}PSO: Personalized e-course composition based on particle swarm optimization," *Appl. Intell.*, vol. 34, no. 1, pp. 141–154, Feb. 2011, doi: 10.1007/s10489-009-0186-7.2011.

[7] C.-M. Chen, Y.-L. Hsieh, and S.-H. Hsu, "Mining learner profile utilizing association rule for Web-based learning diagnosis," *Expert Syst. Appl.*, vol. 33, no. 1, pp. 6–22, Jul. 2007.

[8] C. Romero and S. Ventura, "Educational data mining: A survey from 1995 to 2005," *Expert Syst. Appl.*, vol. 33, pp. 135–146, Jul. 2007.

[9] J. M. M. Vazquez, J. A. O. Ramirez, L. Gonzalez-Abril, and F. V. Morente, "Designing adaptive learning itineraries using features modelling and swarm intelligence," *Neural Comput. Appl.*, vol. 20, no. 5, pp. 623–639, Jul. 2011.

[10] L. de Marcos, J.-J. Martínez, and J.-A. Gutierrez, "Swarm intelligence in e-learning: Swarm intelligence in e-learning: A learning object sequencing agent based on competencies," in *Proc. 10th Annu. Conf. Genetic Evol. Comput.*, Atlanta, GA, USA, Jul. 2008, pp. 17–24.

[11] P. Brusilovsky and C. Peylo, "Adaptive and intelligent technologies for Web-based education," *Int. J. Artif. Intell. Educ.*, vol. 13, no. 2, pp. 159–172, Apr. 2003.

[12] R. Stathacopoulou, M. Grigoriadou, M. Samarakou, and D. Mitropoulos, "Monitoring students' actions and using teachers' expertise in implementing and evaluating the neural network-based fuzzy diagnostic model," *Expert Syst. Appl.*, vol. 32, no. 4, pp. 955–975, May 2007.

[13] Y.-M. Huang, J.-N. Chen, T.-C. Huang, Y.-L. Jeng, and Y.-H. Kuo, "Standardized course generation process using dynamic fuzzy Petri nets," *Expert Syst. Appl.*, vol. 34, no. 1, pp. 72–86, Jan. 2008.

[14] A. Carlisle and G. Dozier, "An off-the-shelf PSO," in *Proc. Part. Swarm Optim. Workshop*, Apr. 2001, pp. 1–6.

[15] Y.-M. Chen, T.-Y. Chen, H.-C. Chu, and K.-C. Su, "Ontology-based adaptive dynamic e-learning map planning method for conceptual knowledge learning," *Int. J. Web-Based Learn. Teach. Technol.*, vol. 11, no. 1, pp. 1–20, Jan. 2016.

[16] K. Almohammadi, H. Hagras, D. Alghazzawi, and G. Aldabbagh, "A survey of artificial intelligence techniques employed for adaptive. Educational systems within e-learning platforms," *J. Artif. Intell. Soft Comput. Res.*, vol. 7, no. 1, pp. 47–64, Dec. 2016.

[17] T. GopalaKrishnan and P. Sengottuvelan, "A hybrid PSO with Naïve Bayes classifier for disengagement detection in online learning," *Program*, vol. 50, no. 2, pp. 215–224, Apr. 2016.

[18] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1272–1282, Dec. 2005.

[19] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 459–472, Aug. 2006.

[20] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 440–458, Aug. 2006.

[21] H. Wang, D. Wang, and S. Yang, "Triggered memory-based swarm optimization in dynamic environments," in *Applications of Evolutionary Computing* (Lecture Notes in Computer Science), vol. 4448. Berlin, Germany: Springer-Verlag, Jun. 2007, pp. 637–646.

[22] X. Li and K. H. Dam, "Comparing particle swarms for tracking extrema in dynamic environments," in *Proc. Congr. Evol. Comput. (CEC)*, Canberra, ACT, Australia, Dec. 2003, pp. 1772–1779.

[23] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proc. Congr. Evol. Comput.*, Seoul, South Korea, May 2001, pp. 94–97.

[24] A. Carlise and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in *Proc. Int. Conf. Artif. Intell.*, Las Vegas, NV, USA, Jun. 2000, pp. 429–434.

[25] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1362–1381, Dec. 2009.

[26] L. De-Marcos, C. Pages, J.-J. Martinez, and J.-A. Gutierrez, "Competency-based learning object sequencing using particle swarms," in *Proc. 19th IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, vol. 2, Oct. 2007, pp. 111–116.

[27] S.-C. Cheng, Y.-T. Lin, and Y.-M. Huang, "Dynamic question generation system for Web-based testing using particle swarm optimization," *Expert Syst. Appl.*, vol. 36, no. 1, pp. 616–624, 2009.

[28] M. R. D. Ullmann, D. J. Ferreira, C. G. Camilo, S. S. Caetano, and L. de Assis, "Formation of learning groups in cMoocs using particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 3296–3304.

[29] X. Cui, T. E. Potok, and P. Palathingal, "Document clustering using particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Pasadena, CA, USA, Jun. 2005, pp. 185–191.

[30] S. Z. Selim and M. A. Ismail, "K-means-type algorithms: A generalized convergence theorem and characterization of local optimality," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 1, pp. 81–87, Jan. 1984.

[31] J. Kogan, C. Nicholas, and M. Teboulle, Eds., *Grouping Multidimensional Data*. Berlin, Germany: Springer, 2005.

[32] K. Premalatha and A. M. Natarajan, "Discrete PSO with GA operators for document clustering," *Int. J. Recent Trends Eng.*, vol. 1, no. 1, pp. 20–24, 2009.

[33] N. I. Ghali, N. El-dessouki, A. N. Mervat, and L. Bakraw, "Exponential particle swarm optimization approach for improving data clustering," *Int. J. Elect. Electron. Eng.*, vol. 3, no. 4, pp. 208–212, 2009.

[34] V. Kalivarapu, J.-L. Foo, and E. Winer, "Improving solution characteristics of particle swarm optimization using digital pheromones," *Struct. Multidisciplinary Optim.*, vol. 37, no. 4, pp. 415–427, Jan. 2009.

[35] H. Izakian, A. Abraham, and V. Snášel, "Fuzzy clustering using hybrid fuzzy c-means and fuzzy particle swarm optimization," in *Proc. IEEE NaBIC*, Dec. 2009, pp. 1690–1694.

[36] K. Yamada, K. Nakakoji, and K. Ueda, ''A mutli-agent systems approach to analyze online community activities,'' in *Proc. ICAM*, Hokkaido, Japan, 2004, pp. 595–600.
[37] M. Wooldrigge, *Introduction to Multi-Agent Systems*, 1st ed. Chichester, U.K.: Wiley, 2002.
[38] C.-Y. Chou, T.-W. Chan, and C.-J. Lin, ''Redefining the learning companion: The past, present, and future of educational agents,'' *Comput. Educ.*, vol. 40, no. 3, pp. 255–269, 2003.
[39] K. Pireva and P. Kefalas, ''The use of multi agent systems in cloud e-learning,'' in *Proc. SEERC Doctoral Conf.*, Thessaloniki, Greece, Sep. 2015, pp. 324–336.
[40] V. Sugumaran, *Distributed Artificial Intelligence, Agent Technology, and Collaborative Applications*. Hershey, PA, USA: IGI Global, 2009.
[41] V. Tabares, N. Duque, and D. A. Ovalle, ''Multi-agent system for expert evaluation of learning objects from repository,'' in *Proc. Int. Conf. Practical Appl. Agents Multi-Agent Syst.*, 2015, pp. 320–330.
[42] T. M. Blackwell, ''Particle swarms and population diversity,'' *Soft Comput.*, vol. 9, no. 11, pp. 793–802, 2005.
[43] R. C. Eberhart and Y. Shi, ''Comparing inertia weights and constriction factors in particle swarm optimization,'' in *Proc. Congr. Evol. Comput.*, vol. 1, 2000, pp. 84–88.
[44] J. Chen, Z. Qin, and J. Jia, ''A weighted mean subtractive clustering algorithm,'' *Inf. Technol. J.*, vol. 7, no. 2, pp. 356–360, 2008.
[45] M. El-Tarabily, R. Abdel-Kader, M. Marie, and G. Abdel-Azeem, ''A PSO-based subtractive data clustering algorithm,'' *Int. J. Res. Comput. Sci.*, vol. 3, no. 2, pp. 1–9, Mar. 2013.
[46] S. Liangtu and Z. Xiaoming, ''Web text feature extraction with particle swarm optimization,'' *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 7, no. 6, pp. 132–136, Jun. 2007.

MARIAM M. AL-TARABILY received the B.S. degree in computer engineering from the Faculty of Engineering, Port Said University, in 1998, and received the M.Sc. degree in computer engineering from Port Said University in 2005. She is currently an Assistant Lecturer with the Education Technology Department, Specified Education Faculty, Port Said University. Her research interests include artificial intelligence and information theory.

REHAB F. ABDEL-KADER received the B.S. degree in computer engineering from Suez Canal University in 1996, the M.Sc. degree in electrical engineering from Tuskegee University in 1999, and the Ph.D. degree from Auburn University in 2003. She is currently an Associate Professor with the Electrical Engineering Department, Faculty of Engineering, Port Said University, Port Fouad, Egypt. Her current research interests include artificial intelligence and computer vision.

GAMAL ABDEL AZEEM received the B.Sc. and M.Sc. degrees in electrical engineering from Suez Canal University in 1976 and 1983, respectively, and the Ph.D. degree from Newcastle University, U.K., in 1986. He is currently an Associate Professor with the Electrical Engineering Department, Faculty of Engineering, Port Said University, Port Fouad, Egypt.

MAHMOUD I. MARIE received the B.Sc., M.Sc., and Ph.D. degrees in electronic and communication engineering from Cairo University in 1972, 1981, and 1985, respectively. He is currently a Professor of communications with the Computer and System Engineering Department, Al-Azhar University, Cairo, Egypt. His fields of interest include digital communication, computer networks, and protocol development.