Using Fuzzy Clustering Chaotic-based Differential Evolution to solve multiple resources leveling in the multiple projects scheduling problem

Duc-Hoc Tran\textsuperscript{a,*}, Min-Yuan Cheng\textsuperscript{b,1}, Anh-Duc Pham\textsuperscript{a,2}

\textsuperscript{a} Faculty of Project Management, University of Science and Technology-The University of Da Nang, \#54, Nguyen Luong Bang Rd., Danang, Viet Nam
\textsuperscript{b} Department of Civil and Construction Engineering, National Taiwan University of Science and Technology, \#43, Sec. 4, Keelung Rd., Taipei 106, Taiwan

Received 15 January 2015; revised 2 March 2016; accepted 28 March 2016
Available online 11 April 2016

Abstract Project scheduling is an important part of construction project planning. Resource leveling is the process used within project scheduling to reduce fluctuations in resource usage over the period of project implementation. These fluctuations frequently create the untenable requirement of regularly hiring and firing temporary staff resources to meet short-term project needs. Construction project decision makers currently rely on experience-based methods to manage fluctuations. However, these methods lack consistency and may result in unnecessary wastage of resources or costly schedule overruns. This research introduces a novel optimization model called the Fuzzy Clustering Chaotic-based Differential Evolution for solving multiple resources leveling in the multiple projects scheduling problem (FCDE-MRLMP). The novel Fuzzy Clustering Chaotic-based Differential Evolution (FCDE) algorithm integrates fuzzy c-means clustering and chaotic techniques into the original Differential Evolution (DE) algorithm to handle complex optimization problems. The chaotic technique prevents the optimization algorithm from converging prematurely. The fuzzy c-means clustering technique acts as several multi-parent crossover operators in order to utilize population information efficiently and enhance convergence efficiency. Experiments run indicate that the proposed model obtains optimal results more reliably and efficiently than the benchmark algorithms considered. The proposed optimization model is a promising alternative approach to assist project managers to handle resource-leveling project scheduling problems effectively.

1. Introduction

Managing corporate resources efficiently is critical to the long-term success and sustainability development of a construction company \cite{1}. The efficient management helps on keeping
operational expenses within planned budget and schedules on time. Time overruns often cause financial loss to the owner due to delays in facility availability [2] and may spark contract disputes that raise operational costs, degrade the company’s reputation, and occasionally lead to project failure [3,4].

Construction resources primarily consist of manpower, equipment, materials, funds, and expertise. Construction schedules generated by using network scheduling techniques that often cause resource fluctuations are impractical, inefficient, and costly to implement [5,6]. Therefore, construction managers must adjust construction schedules manually to eliminate these fluctuations.

Resource fluctuations are a troublesome issue for contractors [7] because hiring and firing workers to harmonize with fluctuating resource profiles are impractical. However, resources must be managed efficiently in order to maximize resource expenditures and meet contracted schedules. Contractors are thus inevitably burdened by a certain percentage of idle resources during periods of low demand, which detracts from project profits.

The process of smoothing out resources, known as resource leveling, has been studied extensively [8–11]. Resource leveling attempts to minimize both the demand peak and the fluctuations in the pattern of resource usage [12,13] by optimizing noncritical activities within their available floats while keeping the project duration unchanged. The application of Evolutionary Algorithms (EAs) to resource leveling has attracted increasing attention in recent years [14–16]. Based on the principles of natural evolution, EAs are stochastic optimization techniques that have successfully resolved optimization problems in diverse fields [17]. However, EAs suffer from certain weaknesses. Geng et al. [14] identified premature convergence and poor exploitation as the main obstacles preventing EAs from coping effectively with complex optimization problems.

Research on resource leveling has focused mainly on three aspects: (1) single-resource leveling in single-project scheduling, (2) multiple-resource leveling in single-project scheduling, and (3) single-resource leveling in multiple-project scheduling. However, multiple-resource leveling in multiple-project scheduling is the most typical scenario in the construction and manufacturing industries, a situation that is relatively more complex and difficult due to the lack of a standard handling procedure [16]. Thus, developing a methodology for multiple-resource leveling in multiple-project problems and a more efficient algorithm to attain better resource-leveling-problem solutions are essential to improve the management of construction project resources.

The Differential Evolution (DE) [18] algorithm is an evolutionary computation technique. DE has drawn increasing interest from researchers, who have explored the capabilities of this algorithm in a wide range of problems. DE is a population-based stochastic search engine that is efficient and effective for global optimization in the continuous domain. DE uses mutation, crossover, and selection operators at each generation to move a population toward the global optimum. The superior performance of DE over competing algorithms has been verified in multiple published research works [19–21].

Despite these advantages, the original DE and its numerous variants have some drawbacks. DE does not guarantee convergence to the global optimum and is easily trapped in local optima, resulting in low optimization precision or even optimization failure [22]. Further, populations may not be distributed over the search space, potentially trapping individuals in a local solution. DE may also require more generations than competing algorithms to converge on the optimal or near-optimal solution [23]. DE is particularly weak in situations in which the global optimum must be located using a limited number of fitness function evaluations. Finally, although DE is good at exploring the search space and locating the region of global minimum, it is slow to exploit the solution [24].

The inherent characteristics of chaotic systems provide an efficient approach maintaining population diversity in search algorithms. Chaos is the irregular motion of a deterministic nonlinear system under deterministic conditions. Because chaotic systems are sensitive to small differences in initial conditions, they may widely generate variant outcomes. This extreme sensitivity to initial conditions reflects the so-called butterfly effect or Liapunov’s sense [25]. Some studies have hybridized DE with the chaotic algorithm. Jia et al. [22] combined chaotic local search (CLS) with a “shrinking” strategy. The CLS helps improve the optimizing performance of the canonical DE by exploring a huge search space in the early run phase to avoid premature convergence and exploiting a
small region in the later run phase to refine the final solutions. Bedri Ozer [23] embedded seven chaotic maps to create the initial DE population. It was found that coupling emergent results in different areas, like those of DE and complex dynamics, improving the quality of results in some optimization problems.

Fuzzy c-means clustering is the process of dividing a set of objects into groups or clusters of similarities in order to increase the speed of optimization search in DE. Successful clustering identifies true natural groupings in the dataset. Fuzzy c-means clustering, a soft clustering approach, was introduced to DE to help track the evolution of the search algorithm. In fuzzy clustering, data elements belong to more than one cluster and are associated with each element via a set of membership levels that indicate the strength of association between a particular data element and a particular cluster. Kwedlo [26] proposed a new version of DE, which uses k-means clustering to find the speed of optimization search in DE. Successful clustering results in different areas, like those of DE and complex dynamics, improving the quality of results in some optimization problems.

Fuzzy c-means clustering is the process of dividing a set of data elements into groups or clusters of similarities in order to increase the speed of optimization search in DE. Successful clustering identifies true natural groupings in the dataset. Fuzzy c-means clustering, a soft clustering approach, was introduced to DE to help track the evolution of the search algorithm. In fuzzy clustering, data elements belong to more than one cluster and are associated with each element via a set of membership levels that indicate the strength of association between a particular data element and a particular cluster. Kwedlo [26] proposed a new version of DE, which uses k-means clustering to fine-tune each candidate solution obtained by the mutation and crossover operators of DE. Wang et al. [27] utilized a clustering technique to improve solution accuracy with less computational effort. Experiments demonstrated that the new method efficiently identifies near-optimal solutions.

Hybridization with other different algorithms is an interesting direction for the further improvement of DE [28]. Although there have been many proposals to improve DE, few have studied the hybridization of the clustering and chaotic techniques with the DE method [28]. To the best of our knowledge, fuzzy c-means clustering and chaotic have not previously been used to enhance the performance of DE.

Therefore, the objective of this research was to use fuzzy c-means clustering and chaotic techniques to overcome the aforementioned shortcomings of the original DE. Chaotic sequences rather than random sequences are adopted. Further, very interesting and good results are exploited to prevent the new approach from premature convergence. Meanwhile, fuzzy c-means clustering acts as several multi-parent crossover operators that use population information efficiently to facilitate faster algorithm convergence. The remainder of this paper is organized as follows: Section 2 briefly reviews literature relating to the establishment of the new optimization model. Section 3 provides a detailed description of the proposed optimization model for the resource-leveling problem. Section 4 uses two numerical experiments to demonstrate model performance. The final section presents conclusions and suggests directions for future work.

2. Literature review

2.1. Multiple resources leveling in the multiple projects problem

A total of \(n\) projects must be started simultaneously in an enterprise. Each project includes multiple activities and each activity uses \(p\) resources. Symbols used in related formulas include the following: the set of activities in the project \(k\) is \(\{(i_k, j_k)\} = \{A_k, \ldots, Z_k\}\); \(R_{m}(t)\) is the demand for resource \(m\) by all \(n\) projects on day \(t\); \(R_{m}(k, j)\) is the demand for resource \(m\) by activity \((i_k, j_k)\) on day \(t\); \(R_{m}(i_k, j_k)\) is the demand for resource \(m\) by activity \((i_k, j_k)\) on day \(t\). \(T_l(i_k, j_k), T_r(i_k, j_k), T_s(i_k, j_k), T_f(i_k, j_k), T_e(i_k, j_k)\) represent early start time, late start time, actual start time, actual finish time, duration, and slack time of \((i_k, j_k)\), respectively. The precedence set of activity \((i_k, j_k)\) is \(\{pset_{i_k, j_k}\}\).

Multiple-resource leveling in multiple-project scheduling differs from conventional resource leveling techniques primarily as follows [16]:

- Firstly, due to differing levels of resource demand, assimilation must transform absolute demand into relative demand in order to enable all the \(p\) resources to be comparable in terms of quantity. The relative demand of resource \(m\) in all \(n\) projects on day \(t\) may be expressed as follows:
  \[
  SR_{m}(t) = \frac{R_{m}(t)}{R_{\max m}}
  \]
  where \(R_{\max m} = \max \{R_{m}(t)\}\) denotes the maximum demand for resource \(m\) in total \(n\) projects on one day and \(\lambda\) is an amplifying coefficient within [1100] used to increase simulation accuracy.

- The above formula limits the relative demand for each resource in a total of \(n\) projects on every single day to between 0 and \(\lambda\).

- Secondly, the weight score \(w_{m}\) measures degree of importance for each resource. This paper uses the analytical hierarchy process (AHP) to set the weights of different resources. Larger weight scores correlate to greater priority.

The mathematical formulation objective function for multiple-resource leveling in multiple-projects scheduling is as follows:

\[
\text{Min}RI = \sum_{i=1}^{n} \sum_{m=1}^{p} w_{m}(SR_{m}(t) - SR_{\text{max}})^2
\]

subject to:

\[
T_e(i_k, j_k) \leq T_s(i_k, j_k) \leq T_f(i_k, j_k)
\]

\[
\max\{T_s(pset_{i_k, j_k}) + T_z(pset_{i_k, j_k})\} \leq T_s(i_k, j_k) \leq T_f(i_k, j_k)
\]

\[
R_{m}(t) = \sum_{k=1}^{n} \sum_{j=1}^{n} R_{m}(i_k, j_k) ; SR_{m} = \frac{1}{T} \sum_{t=1}^{T} SR_{m}(t)
\]

\[
R_{m}(i_k, j_k) = \begin{cases} R_{m}(i_k, j_k) & \text{if: } T_s(i_k, j_k) < t \leq T_f(i_k, j_k) \\ 0 & \text{if: } T_s(i_k, j_k) \text{ ort } T_f(i_k, j_k) \end{cases}
\]

\[
S(i_k, j_k) = T_f(i_k, j_k) - T_e(i_k, j_k)
\]

where \(T\) is equal to the difference between the maximum of the latest finish time and the minimum of the earliest start time for all \(n\) projects.

2.2. Related works

In construction projects, many researchers have done numerous studies of modeling resource leveling problem in the literature. A variety of models, ranging from mathematical methods, heuristics to evolutionary, meta-heuristic approaches (e.g. Genetic Algorithm, Particle Swarm Optimization, Differential Evolution, Ant Colony Optimization) were proposed to solve the resource leveling problems [29]. Initially, the mathematical approaches were used to solve the resource leveling problems because they can provide optimal solutions to the problem at hand. However, these methods become impractical when the size of project network reaches a considerably large number. Hence, the increasing number of decision variables causes the problem solving to become infeasible [8]. As a
result, mathematical approaches are not computationally manageable for real-world construction projects [12].

Other researchers attempted to utilize heuristic methods in solving the resource leveling problem [9,30]. However, using heuristic methods frequently is not sufficiently enough to satisfy project managers despite their simplicity and wide implementation on commercial project management software (e.g. Microsoft Project). The reason is because the methods operate on the basis of pre-defined rules. Consequently, their performance relies on specific types of problem and on which rules are implemented. For this reason, only a decent feasible solution can be produced without any guarantee on finding an optimum solution [31].

Due to the limitations of mathematical and heuristic methods, the application of Evolutionary Algorithms (EAs) for resource leveling has attracted more attention in recent years [14,15,32–34]. Metaheuristic is recognized as a stochastic optimization method that is inspired by the phenomena seen in nature. They have been successfully used to solve optimization problems in diverse fields [17]. Some widely known metaheuristic algorithms, such as Genetic Algorithm, Particle Swarm Optimization, Ant Colony Optimization, and Differential Evolution, remain an active research area in the scientific community. However, these algorithms are not free from certain limitations. Geng et al. [14] point out that premature convergence and poor exploitation are the major drawbacks for the metaheuristic when facing more complex problems. Thus, more advanced algorithms are continuously needed to achieve satisfactory solutions for resource leveling problem in modern construction projects.

2.3. Differential Evolution optimization algorithm

Differential Evolution (DE) is a simple population-based, direct-search approach to solving global optimization problems [18,35]. The original DE algorithm is described briefly as follows:

Let $S \subset \mathbb{R}^n$ be the search space of the problem under consideration. DE utilizes $NP$, $D$-dimensional parameter vectors: $X_i = \{x_{i1}, x_{i2}, \ldots, x_{iD}\}, i = 1, 2, \ldots, NP$ as a population for each algorithm generation. The initial population is generated randomly and should cover the entire parameter space. At each generation, DE applies two operators, mutation and crossover (recombination), to yield one trial vector $U_{iG+1}$ for each target vector $X_i$. Next, a selection phase determines whether or not the trial vector enters the population in the next generation. For each target vector $X_i$, the following equation is used to determine the mutant vector:

$$V_{iG+1} = X_{iG} + F(X_{iG} - X_{G})$$

where $r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$ are randomly selected such that $r_1 \neq r_2 \neq r_3 \neq i$ and $F$ is a scaling factor such that $F \in [0, 1]$.

Following the mutation phase, the crossover operator is applied to increase diversity. For each mutant vector $V_{iG+1}$, the following formula is used to generate a trial vector $U_{iG+1}$:

$$U_{iG+1} = \left\{ u_{iG+1}^{1}, u_{iG+1}^{2}, \ldots, u_{iG+1}^{D} \right\}$$

$$u_{jG+1}^{i} = v_{jG+1}^{i} \begin{cases} \text{if}(\text{rand}[0,1]) \leq CR \text{rand} = j_{\text{rand}} & \text{if} j = 1, 2, \ldots, D \\ X_{iG}^{j} & \text{otherwise} \end{cases}$$

where $CR \in [0, 1]$ is a user-defined crossover constant and $j_{\text{rand}}$ is a randomly chosen index from $\{1, 2, \ldots, D\}$ that ensures trial vector $U_{iG+1}$ differs from its target $X_{iG}$ by at least one parameter.

Trial vector $U_{iG+1}$ is next compared to the corresponding target vector $X_{iG}$ using the greedy criterion to determine whether the former should remain a member of the population in the next generation. The selection operator is expressed as follows:

$$X_{iG+1} = \begin{cases} U_{iG+1} & \text{if}(U_{iG+1}) < (X_{iG}) \\ X_{iG} & \text{otherwise} \end{cases}$$

With memberships in the next generation now determined, the DE evolutionary cycle iterates until attaining the stopping condition.

2.4. Chaos sequences

Chaos theory is a scientific theory that describes erratic behavior in certain nonlinear dynamic systems. Chaotic mappings may be visualized as particles traveling within a limited range in a deterministic, nonlinear and dynamic system with no definite regularity associated with the path of travel of these particles. Although particle movement is randomized, movement is extremely sensitive to the initial conditions [36]. Because chaotic sequences may be generated and stored quickly and easily, there is no need for storage over long sequences [23]. Moreover, these sequences are deterministic and reproducible. Thus, many researchers have adopted chaotic sequences instead of random sequences [37,38].

The one-dimensional logistic map is one of the simplest systems with a density of periodic orbits:

$$X_{n+1} = \mu X_n(1 - X_n)$$

In this equation, $X_n$ is the $n^\text{th}$ chaotic number, where $n$ denotes the iteration number. Obviously, $X_n \in (0, 1)$ under conditions that initial $X_0 \in (0, 1)$ and that $X_0 \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$. The variance of control parameter $\mu$ in Eq. (11) directly and significantly impacts the behavior of $X$. The domain area of control parameter $\mu$ has often been defined as $[0, 4]$. $\mu = 4$ has been used in several previously published research experiments [39,40].

2.5. Fuzzy c-means clustering

Clustering decomposes a set of objects into subgroups or clusters based on similarity so that objects within a cluster are as similar as possible to one another and as dissimilar as possible from objects in other clusters. Principal clustering algorithm categories are as follows: (1) crisp (or hard) clustering procedures in which each data point is assigned to exactly one cluster and (2) fuzzy clustering techniques in which each data point belongs to every cluster according to a specific algorithm degree of membership [41]. Many clustering algorithms have been introduced in the literature. Fuzzy clustering presents the advantage of dealing efficiently with overlapping clusters, delivering better, more stable results than other clustering techniques [42,43]. This study employs fuzzy c-means (FCM) clustering [44].
3. Fuzzy c-means Clustering Chaotic-based Differential Evolution for Multiple Resources Leveling in Multiple Projects (FCDE-MRLMP)

This section describes the newly proposed FCDE-MRLMP model in detail. The FCDE, the core optimizer in the FCDE-MRLMP model, integrates original DE [18,35] with fuzzy c-means clustering and chaotic techniques. The chaos approach effectively exploits the whole search space and provides the diversity necessary in the DE population. This operation incurs additional time and iterations to search for the global optimum. The fuzzy c-means clustering technique introduces cluster centers that provide direction to the global optimum search, which improves overall search algorithm efficiency and enhances algorithm convergence speed. Fig. 1 illustrates the proposed model. The objective of the FCDE-MRLMP was to minimize daily fluctuations in resource utilization without changing total project duration.

3.1. Initialization

Inputs required by the FCDE-MRLMP optimization model include activity relationship, activity duration, and resource demand. In addition, the user must provide search engine parameter settings such as maximum number of search generations \(G_{\text{max}}\) and population size (NP). The scheduling procedure uses these inputs in the calculation process to obtain the project duration and resource amount required for each activity. The model operates automatically.

Prior to beginning of the search process, a uniform random number generator creates an initial population of feasible solutions. A solution for the resource-leveling problem is represented as a vector with \(D\) elements as follows:

\[
X = [X_1, X_2, \ldots, X_D]
\]

where \(D\) is the number of decision variables in the problem at hand. \(D\) is also the number of non-critical activities in the project network. The index \(i\) denotes the \(i\)th individual in the population. The vector \(X\) represents the start time of \(D\) non-critical activities in the network. Because the original DE operates with real-value variables, a function is employed to convert the start times of those activities from real values to integer values that are constrained within the feasible domain.

\[
X_{ij} = \text{Round}(LB(j) + \text{rand}[0, 1] \times (UB(j) - LB(j)))
\]

where \(X_{ij}\) is the start time of activity \(j\) at the individual \(i\th\). \(\text{rand}[0, 1]\) denotes a uniformly distributed random number between 0 and 1. \(LB(j)\) and \(UB(j)\) are the early start time and late start time for activity \(j\). In multiple resources leveling in the multiple projects scheduling problem, two constraint conditions limit the actual start time of all activities: (1) actual start time must be between the early and late start times and (2) actual start time is limited by the actual start time of its predecessor activities. The first constraint is simple to handle because limits are fixed prior to calculation. However, the minimum limit of the second constraint is unknown prior to calculation and thus more difficult to elicit. For the decision variables of FCDE on each dimension is determined in turn, when calculating the actual start time of one activity, actual start time of all activities in its predecessor set \(T_s(pset_k, i_k)\) has been computed, and the max\{\(T_s(pset_k, i_k) + T(pset_k, i_k)\}\) has been confirmed simultaneously.

3.2. Mutation

Once initialized, DE mutates the population to produce a set of mutant vectors. A mutated vector \(V_{i,G+1}\) is generated that corresponds to the target vector \(X_{i,G}\), as in Eq. (8).

3.3. Crossover

The crossover operation exchanges components of the target vector and the mutant vector to diversify the current population. In this stage, a new vector, named the trial vector, is created using Eq. (9).

3.4. Selection

In this stage, the trial vector is compared to the target vector (or the parent) using the greedy criterion. The trial vector replaces the position of the target vector if the trial vector yields a lower objective function value than its parent. Otherwise, the target vector retains its place in the population for another generation. The selection operator is expressed as Eq. (10).

3.5. Chaotic operation

The Chaos Differential Evolution (CDE) algorithm generates chaotic sequences in DE that ensure individuals in the population are distributed as evenly as possible throughout the search space. Integrating CDE into DE enhances global convergence by escaping suboptimal solutions. Fig. 2 shows the main steps used to generate a chaotic population. In this figure, \(g\) and \(G_{\text{max}}\) are the current generation and the maximum generation, respectively. If the probability condition is satisfied, a percentage of the population \((CF)\) is selected for chaos. \(CF\) is then mapped to the chaotic feasible region in range \((0, 1)\) according to chaotic conditions and performs
the logistic map according to Eq. (11) in order to yield chaotic values $cm_g^{j,k}$. Afterward, Eq. (14) is used to map the chaotic values to the feasible region:

$$X_g^{j,k} = X_{\text{min}}^{j} + cm_g^{j,k}(X_{\text{max}}^{j} - X_{\text{min}}^{j})$$

where $X_g^{j,k}$ is the $j$th decision variable of the $k$th individual in the $CF$-population at generation $g$; $X_{\text{max}}^{j}$ and $X_{\text{min}}^{j}$ are, respectively, the upper and lower bounds of the $j$th decision variable.

### 3.6. Fuzzy clustering operation

The FCM clustering technique adopted in DE, named FDE, easily conducts an efficient convergence of DE. The FCM was introduced in this study to track the main stream of population movement during DE evolution. Each cluster center may thus be treated approximately as one of the items in the main stream of evolution and replaced in population as candidate individuals. Fig. 3 illustrates the FDE algorithm.

- **Selection plan:** Choose $k$ individuals from current population randomly (set A).
- **Generation plan:** Create $k$ offspring (set B) using fuzzy c-means clustering.
- **Replacement plan:** Choose $k$ best solutions (set C) from the combined set (set A + set B) for replacement.
- **Update plan:** Update the population as $P = P - \text{set A} + \text{set C}$.

In the update plan, the $k$ best solutions are chosen from the combined set, which ensures that elites are preserved.

### 3.7. Stopping condition

The optimization process terminates when a user-set stopping criterion is met. This stopping criterion is often set as the maximum generation $G_{\text{max}}$ or the maximum number of function evaluations (NFE). Search process termination identifies the optimal solution. The project schedule and its corresponding resource histogram may then be constructed based on the optimal start time for activities.

### 4. Case study

Two construction case studies were used to demonstrate the capability of the newly developed FCDE-MRLMP model. The first case study was adapted from Yan et al. [16]. In the first case study, an enterprise must start two projects with same total project duration. Fig. 4 shows the precedence relationships of the network projects. Each activity in both projects uses three resources ($R_1$ human, $R_2$ fund, and $R_3$ equipment) and has a certain duration $D$ that is indicated above the arrow line.
Based on the importance of each resource, the AHP method makes pairwise comparison of each resource. The comparison matrix is obtained as follows:

\[
R_1 = \begin{bmatrix} 1 & 3 & 5 \\ 3 & 1 & 3 \\ 5 & 3 & 1 \end{bmatrix}
\]

Consistency inspection demonstrates that this comparison matrix is acceptable. Weights for each resource are set as follows:

\[
w_1 = 0.637, \quad w_2 = 0.258, \quad w_3 = 0.105.
\]

Consequently, the objective function for the case study 1 is calculated as follows:

\[
\min RI = \frac{1}{18} \sum_{i=1}^{18} \left[ 0.637 \left( SR_i(t) - \overline{SR}_i(t) \right)^2 + 0.258 \left( SR_i(t) - \overline{SR}_i(t) \right)^2 \right] + 0.105 \left( SR_i(t) - \overline{SR}_i(t) \right)^2
\]

\[
\begin{align*}
0 & \leq T_i(A_i) \leq 7 \\
0 & \leq T_i(B_i) \leq 3 \\
T_i(B_i) + 5 & \leq T_i(C_i) \leq 8 \\
0 & \leq T_i(F_i) \leq 6 \\
T_i(B_i) + 4 & \leq T_i(G_i) \leq 10 \\
0 & \leq T_i(H_i) \leq 3 \\
0 & \leq T_i(I_i) \leq 15 \\
0 & \leq T_i(A_2) \leq 9 \\
0 & \leq T_i(D_2) \leq 9 \\
5 & \leq T_i(G_2) \leq 7 \\
5 & \leq T_i(H_2) \leq 13
\end{align*}
\]

The second case study is a real construction project named Sky Park Residence Project in Viet Nam. All the detailed data of the project were obtained from the Licogi 16 Joint Stock Co. The project is located at the northwest gate of the capital of Hanoi. The total land area of the project is 9262 m², of which construction area is 3342 m², accounting for 36%. Scale of the project includes two blocks, office blocks which are 25 stories tall, tall apartment blocks have 35 floors, 3 basements for car, five floors are used as commercial centers, All the activities considered are extracted from the foundation and basement construction phase. The data for the activities are obtained based on previous data and experts’ experience. Test network was slightly modified to adapt to model requirements. Fig. 5 shows the precedence relationships of the network projects of the second case study. Each activity in both projects uses two resources (R₁ human and R₂ equipment) and has a certain duration D that is indicated above the activity’s node.

In the second case, each recourse has the same importance weights \(w_1 = w_2 = 0.5\).

4.1. Optimization result for the FCDE-MRLMP

Application of the FCDE-MRLMP model reduces significant fluctuations in resource usage. This study set parameters for the FCDE optimizer are based on proposed values from the literature and several experiments as shown in Table 1.

Fig. 6 shows the network resource profile for the projects at initialization and after FCDE-MRLMP optimization process of the first case. It can be seen in Fig. 6 that the proposed algorithm significantly reduces undesirable resource fluctuations. Example on the resource R₂, at initial schedule the maximum daily resource usage and maximum deviation in daily resource usage are 110 and 38, respectively. However, after being optimized by proposed algorithm, the maximum daily resource usage and maximum deviation in daily resource usage reduce considerably to 62 and 8, respectively.
4.2. Result comparisons

Three different algorithms were used to verify the comparative performance of the proposed model (FCDE-MRLMP). These algorithms were as follows: original DE (DE) [35], Genetic Algorithm (GA) [47], and Particle Swarm Optimization (PSO) [48]. For comparison purposes, all four algorithms used an equal number of function evaluations. In GA, the constant mutant and crossover probability factors were set at 0.5 and 0.9, respectively. In PSO, the two learning factors, $c_1$, $c_2$, were both chosen at 2.05, and the inertia factor $w$ is set in the range of 0.3–0.7. DE and FCDE control parameters remained the same, as stated previously in Table 1. Twenty-five independent runs were carried out for all experiments.

Table 2 lists the optimal results, optimal non-critical-activity start times obtained from the new proposed model,
and other benchmark algorithms for the first case. $RI_m$ in Table 2 is the resource intensity for single resource $m$:

$$\min RI = \frac{1}{18} \sum_{t=1}^{T} \left( R_m(t) - \bar{R}_m(t) \right)^2, \quad \bar{R}_m = \frac{1}{18} \sum_{t=1}^{T} R_m(t)$$

As shown in Table 2, the optimal resource intensity ($RI$) obtained by FCDE was, respectively, 94.9%, 2.1%, 6.9%, and 8.6% less than the initial schedule, DE, PSO, and GA. Fig. 7 presents the resource profile after being optimized by each algorithm. It can be seen from these figures that the FCDE leveled projects shows a more promising distribution of resources compared to those of benchmark algorithms.

To evaluate the stability and accuracy of each algorithm, optimization performance was expressed in terms of best result found (best), average result (avg), standard deviation (std), and worst result (worst) after 25 runs (Table 3). The best and worst results demonstrate the capacity of each algorithm to find the optimal solution for all of the performance measurement metrics. Average and standard deviation are two additional characteristics that describe solution quality. Standard deviation occurs in cases when algorithms are not able to generate optimal solutions in all executions.

As shown in Table 3, the performance of the FCDE-MRLMP is competitive in terms of accuracy and stability. The highlighted row in Table 3 shows the proposed algorithm with best performance in both case studies. It is clearly shown that the FCDE-MRLMP is only able to find optimal solutions

<table>
<thead>
<tr>
<th>Item</th>
<th>$RI_1$</th>
<th>$RI_2$</th>
<th>$RI_3$</th>
<th>$RI_4$</th>
<th>Actual start time of non-critical activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>89.46</td>
<td>76.95</td>
<td>1169</td>
<td>123.8</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>5.327</td>
<td>1.06</td>
<td>13.76</td>
<td>9.17</td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td>4.897</td>
<td>0.84</td>
<td>26.65</td>
<td>7.95</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>4.652</td>
<td>0.84</td>
<td>21.97</td>
<td>5.73</td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>4.558</td>
<td>0.62</td>
<td>21.31</td>
<td>9.51</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case 1</th>
<th>Fitness value</th>
<th>GA</th>
<th>PSO</th>
<th>DE</th>
<th>FCDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>5.327</td>
<td>4.897</td>
<td>4.652</td>
<td>4.558</td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>6.907</td>
<td>6.327</td>
<td>5.400</td>
<td>4.760</td>
<td></td>
</tr>
<tr>
<td>Std.</td>
<td>2.112</td>
<td>0.742</td>
<td>0.424</td>
<td>0.300</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case 2</th>
<th>Fitness value</th>
<th>GA</th>
<th>PSO</th>
<th>DE</th>
<th>FCDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>25.329</td>
<td>24.479</td>
<td>24.469</td>
<td>24.469</td>
<td></td>
</tr>
<tr>
<td>Std.</td>
<td>0.616</td>
<td>1.051</td>
<td>0.123</td>
<td>0.025</td>
<td></td>
</tr>
</tbody>
</table>
in fitness function in the first case. Further, in terms of average results, FCDE-MRLMP performed the best of the considered algorithms because it generated the lowest average fitness solution, with a value of 4.760 and deviation value of 0.300 for the first case, and in the second case, the value of 24.472 for average fitness and deviation value of 0.025. Fig. 8 illustrates the best fitness value results of different approaches by number of iterations. As can be seen in Fig. 8 the proposed algorithm outperformed the other approaches in terms of convergence since the proposed model found the best solution in fewer iterations than the original DE and other benchmark algorithms.

4.3. Hypothesis test

A hypothesis test was performed to further demonstrate the superiority of the FCDE performance over that of benchmark algorithms. Because all indicator comparisons demonstrated that the DE performed better on average than either PSO or GA, the hypothesis tests only considered FCDE and DE. A one-tailed $t$-test with equal sample sizes and unequal and unknown variances analyzed the following hypothesis tests:

Hypothesis: FCDE versus standard DE in terms of resource intensity (RI) (Table 3).

H$_0$: There is no difference in RI of the FCDE algorithm and that of the standard DE algorithm.

H$_1$: The FCDE algorithm is significantly better than the standard DE algorithm.

For the first case: FCDE $s_1 = 0.300$; DE: $s_2 = 0.424$; $n_1 = n_2 = n = 25$;

$$v = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{(s_1^2/n_1^2 + s_2^2/n_2^2)^2} = \frac{(0.300^2/25 + 0.424^2/25)^2}{(0.300^2/25 + 0.424^2/25)^2}$$

$$= 43.2$$

Critical value: with significant level of $t$-test $\alpha = 0.05$; $v = 43$; we have $t_{v/2} = t_{0.05;43} = 1.681$.

Statistical test: $t = \frac{(x_1 - x_2)}{\sqrt{s_1^2/n_1 + s_2^2/n_2}} = \frac{4.760 - 5.400}{0.300^2/25 + 0.424^2/25}$

$$= -6.161 < -1.681 = -t_{0.05;43}$$

where $n$ is the sample size (number of experimental runs), $v$ is the degrees of freedom used in the test, and $s_1^2$ and $s_2^2$ are the unbiased estimators of the variances of the two samples (FCDE and DE). The denominator of $t$ is the standard error of the difference between two means $x_1$, $x_2$ (average).

For the second case: FCDE $s_1 = 0.025$; DE: $s_2 = 0.123$; $n_1 = n_2 = n = 25$;

$$v = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{(s_1^2/n_1^2 + s_2^2/n_2^2)^2} = \frac{(0.025^2/25 + 0.123^2/25)^2}{(0.025^2/25 + 0.123^2/25)^2}$$

$$= 25.98$$

Critical value: with significant level of $t$-test $\alpha = 0.05$; $v = 26$; we have $t_{v/2} = t_{0.05;26} = 1.706$

$$t = \frac{(x_1 - x_2)}{\sqrt{s_1^2/n_1 + s_2^2/n_2}} = \frac{(24.472 - 24.580)}{\sqrt{0.025^2/25 + 0.123^2/25}}$$

$$= -1.912 < -1.706 = -t_{0.05;26}$$

The statistical test value above is smaller than critical value in both case studies. Therefore, $H_0$ is rejected. The proposed FCDE is thus demonstrated to be statistically superior to the standard DE in terms of resource intensity.

5. Conclusions

This paper presents FCDE to solve the problem of multiple-resource leveling in the context of multiple-projects scheduling. Integrating fuzzy clustering and chaos algorithms into the DE effectively eliminates the drawbacks of the original DE. The inherent randomness in a chaos algorithm enhances population diversity and avoids entrapment in the local optima, while fuzzy c-means clustering uses moving cluster centers to improve the convergence speed of the search algorithm. Two application case studies were analyzed to illustrate the effectiveness of the proposed model and to demonstrate the capabilities of the model in generating an optimal schedule by eliminating undesirable resource fluctuations and resource idle times. Experimental results and a comparison of results indicate that the FCDE-MRLMP effectively improves the performance of the original DE beyond the levels of performance attained by other benchmark algorithms. The proposed model obtained the optimal values with fewer iteration, the lowest
The FCDE has a potential application in broad cases because the model is easily modified to solve many other classes of single-objective optimization problems in the construction management field such as resource-allocation and resource-constrained problems. Moreover, resource-leveling problems in the realm of total-project-cost minimization are frequently encountered in construction management. Trade-offs between time and cost are necessary to improve overall construction project benefits. Further work is necessary to address these issues in order to apply FCDE to the solution of complicated resource-leveling problems that consider multi-objective optimizations. Extending the current FCDE from a single-objective to a multi-objective format using multiple objective Differential Evolution theory is an interesting direction for further research.

Acknowledgments

The authors wish to acknowledge the University of Science and Technology-The University of Da Nang (DUT) for financially supporting this work. Appreciation is also expressed to Faculty of Project Management’s research group at DUT for assistance provided during this investigation. Finally, the authors would like to thank anonymous referees for their valuable comments.

References