Optimizing Multiple-Resources Leveling in Multiple Projects Using Discrete Symbiotic Organisms Search

Min-Yuan Cheng, A.M.ASCE[1]; Doddy Prayogo, A.M.ASCE[2]; and Duc-Hoc Tran, A.M.ASCE[3]

Abstract: Resource leveling is used in project scheduling to reduce fluctuation in resource usage over the period of project implementation. Fluctuating resource usage frequently creates the untenable requirement of regularly hiring and firing temporary staff 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 waste of resources or costly schedule overruns. This research introduces a novel discrete symbiotic organisms search for optimizing multiple resources leveling in the multiple projects scheduling problem (DSOS-MRLMP). The optimization model proposed is based on a recently developed metaheuristic algorithm called symbiotic organisms search (SOS). SOS mimics the symbiotic relationship strategies that organisms use to survive in the ecosystem. Experimental results and statistical tests indicate that the proposed model obtains optimal results more reliably and efficiently than do the other optimization algorithms considered. The proposed optimization model is a promising alternative approach to assisting project managers in handling MRLMP effectively.
DOI: 10.1061/(ASCE)CP.1943-5487.0000512.

Author keywords: Multiple-resources leveling; Scheduling; Symbiotic organisms search; Optimization; Construction management.

Introduction

Resource management is one of the key success factors that allow construction contractors to remain competitive in today's construction business environment. Proper resource management helps to keep the operational expenses of the project within budget and the schedule on time (Cheng and Tran 2014; Damci et al. 2013; Hossein Hashemi Doulabi et al. 2011). Construction resources consist primarily of manpower, equipment, materials, funds, and expertise. Obviously, proper management of these resources plays a significant role in the successful accomplishment of any project.

One of the most common problems faced by construction project managers is a scarcity of resources. Timing the need for resources should be determined during project scheduling. However, project schedules generated using network scheduling techniques, such as PERT and CPM, often cause resource fluctuations that are impractical, inefficient, and costly to implement (Martinez and Ioannou 1993). In fact, resource fluctuations become a troublesome issue for contractors because hiring and firing the workers necessary to harmonize with fluctuating resource profiles is impractical (Christodoulou et al. 2010). Thus, contractors are inevitably burdened by a certain percentage of idle resources during periods of low demand, which reduces project profits. Therefore, resources must be managed efficiently to minimize resource expenditures and meet contracted schedules (Hariga and El-Sayegh 2011; Tang et al. 2014).

The process of smoothing out resource demand, known as resource leveling, has been studied extensively (Doulabi et al. 2011; Savin et al. 1996; Son and Skibniewski 1999). Resource leveling attempts to minimize both the demand peak and the fluctuations in patterns of resource use (El-Rayes and Jun 2009; Yan et al. 2005) by optimizing noncritical activities in their available floats while keeping the project duration unchanged. Research on resource leveling has focused mainly on three aspects: (1) single-resource leveling in single-project scheduling (Damci and Polat 2014); (2) multiple-resources leveling in single-project scheduling (Ponz-Tienda et al. 2013); and (3) single-resource leveling in multiple-project scheduling. However, multiple-resources leveling in multiple-projects scheduling (MRLMP) is the most typical scenario in the construction and manufacturing industries. It is relatively more complex and difficult to solve and lacks a standard handling procedure (Guo et al. 2009). Thus, developing a more efficient optimization algorithm for MRLMP problems and achieving better resource-leveling solutions are essential to improving the management of construction project resources.

As optimization problems vary extensively, a great number of studies have been devoted to the development of new metaheuristic algorithms (Alsayegh and Hariga 2012; Cheng et al. 2014; Koulinas and Anagnostopoulos 2013). Metaheuristic algorithms perform better than most traditional mathematical techniques in solving modern optimization problems because they do not require substantial gradient information (Koulinas and Anagnostopoulos 2013; Wu et al. 2014). A very promising recent development in the field of metaheuristic algorithms is the symbiotic organisms search (SOS) algorithm (Cheng and Prayogo 2014). The SOS algorithm is based on interactive behavior among organisms in nature. Preliminary studies indicate that it is superior to the widely used genetic algorithm (GA), particle swarm optimization (PSO), differential evolution (DE), and bees algorithm (BA) in solving a various continuous benchmark function and engineering

[1]Professor, Dept. of Civil and Construction Engineering, National Taiwan Univ. of Science and Technology, 43 Keelung Rd., Daan District, Taipei 106, Taiwan.

[2]Ph.D. Candidate, Dept. of Civil and Construction Engineering, National Taiwan Univ. of Science and Technology, 43 Keelung Rd., Daan District, Taipei 106, Taiwan; and Lecturer, Dept. of Civil Engineering, Petra Christian Univ., 121-131 Siwalankerto, Surabaya 60236, Indonesia.

[3]Ph.D. Candidate, Dept. of Civil and Construction Engineering, National Taiwan Univ. of Science and Technology, 43 Keelung Rd., Daan District, Taipei 106, Taiwan (corresponding author). E-mail: duchoc87@gmail.com

Note. This manuscript was submitted on December 26, 2014; approved on May 1, 2015; published online on June 16, 2015. Discussion period open until November 16, 2015; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801/04015036(9)/$25.00.

problems (Cheng and Prayogo 2014). Because the SOS algorithm is relatively new, its ability to find a global solution is very interesting and should be further explored and investigated.

The aim of this paper is to propose a new discrete optimization model for solving MRLMP problems based on the SOS metaheuristic algorithm (DSOS-MRLMP). A methodology for transforming the continuous SOS into a discrete search is provided because SOS was originally designed for continuous optimization problems and MRLMP is considered a discrete problem.

Research Contributions

In presenting the discrete SOS (DSOS) as a novel optimization algorithm, this study makes two important contributions. First, DSOS is a new, discrete version of the basic SOS algorithm. It transforms continuous solutions into discrete solutions to fit the MRLMP. Second, DSOS is more effective and efficient than widely used evolutionary algorithms, as demonstrated in a numerical construction case study. DSOS outperformed GA, PSO, and DE in terms of accuracy, solution stability, and satisfaction.

The remaining sections of this paper briefly review the literature on the new optimization model. Then a detailed description of the proposed model for the resource-leveling problem is presented. Subsequently, the model's performance is demonstrated using numerical experiments and result comparisons. The final section presents conclusions and suggestions for future work.

Literature Review

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; $R_{mt}(i_k, j_k)$ 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 one day. $T_E(i_k, j_k)$, $T_L(i_k, j_k)$, $T_s(i_k, j_k)$, $T_f(i_k, j_k)$, $T(i_k, j_k)$, and $S(i_k, j_k)$ 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 $[(\text{pset}_k, i_k)]$.

Multiple-resources leveling in multiple-projects scheduling differs from conventional resource leveling primarily as described in the following paragraphs (Guo et al. 2009). First, because of differing levels of resource demand, assimilation must transform absolute demand into relative demand to enable all 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

$$SR_m(t) = \lambda R_m(t)/R_{\max_m} \tag{1}$$

where $R_{\max_m} = \max\{R_m(t)\}$ = maximum demand for resource m in a total of n projects on one day and λ = amplifying coefficient within [1,100] used to increase simulation accuracy.

Eq. (1) limits the relative demand for each resource in a total of n projects on every single day to between 0 and λ.

Second, the weight score w_m measures the 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 with greater priority.

The mathematical formulation of the objective function for multiple-resources leveling in multiple-projects scheduling is

$$\min \text{RI} = \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{p}\{w_m[\text{SR}_m(t) - \overline{\text{SR}_m}]^2\} \tag{2}$$

subject to

$$T_E(i_k, j_k) \le T_s(i_k, j_k) \le T_L(i_k, j_k) \tag{3}$$

$$\max[T_s(\text{pset}_k, i_k) + T_s(\text{pset}_k, i_k)] \le T_s(i_k, j_k) \le T_L(i_k, j_k) \tag{4}$$

$$R_m(t) = \sum_{k=1}^{n}\sum_{i_k, j_k} R_{mt}(i_k, j_k); \qquad \overline{\text{SR}_m} = \frac{1}{T}\sum_{t=1}^{T}\text{SR}_m(t) \tag{5}$$

$$R_m(i_k, j_k) = \begin{cases} R_m(i_k, j_k) \text{ if:} T_s(i_k, j_k) < t \le T_f(i_k, j_k) \\ 0 \text{ if:} t < T_s(i_k, j_k) \text{ or } t > T_f(i_k, j_k) \end{cases} \tag{6}$$

$$S(i_k, j_k) = T_L(i_k, j_k) - T_E(i_k, j_k) \tag{7}$$

where T = difference between the maximum of the latest finish time and the minimum of the earliest start time for all n projects.

Related Works on the Resource-Leveling Problem

The literature includes many studies of modeling the resource-leveling problem in construction projects. A variety of methods, ranging from mathematical and heuristic to evolutionary and metaheuristic (e.g., GA, PSO, DE, ant colony ptimization) have been proposed to solve the resource-leveling problem. At the beginning, researchers used various mathematical approaches because they could provide near-optimal solutions. However, as many project networks became increasingly complex, these methods became impractical. Moreover, because resource leveling is a combinatorial problem, the increasing number of decision variables makes problem solving infeasible (Savin et al. 1996). As a result, mathematical approaches are not computationally manageable for real-world construction projects (Yan et al. 2005).

To overcome these shortcomings, many researchers began to propose heuristic methods as alternatives for handling the resource-leveling problem (Harris 1990; Son and Skibniewski 1999). However, heuristic methods frequently are not sufficient to satisfy project managers despite their simplicity and wide implementation in commercial project management software (e.g., Microsoft Project). The reason is that these methods operate on the basis of predefined rules. Consequently, their performance relies on specific types of problems and on the rules implemented. For this reason, a decent feasible solution has no guarantee of finding an optimum solution (Hegazy 1999).

The shortcomings of the aforementioned mathematical and heuristic methods have encouraged studies of metaheuristic algorithms for solving the resource-leveling problem in recent years (El-Rayes and Jun 2009; Geng et al. 2011; Hossein Hashemi Doulabi et al. 2011; Leu et al. 2000). Metaheuristics is recognized as a stochastic optimization method inspired by phenomena seen in nature. Such methods have been successfully used to solve optimization problems in diverse fields (Das and Suganthan 2011). Some widely known metaheuristic algorithms, such as the GA, PSO, ant colony optimization, and DE, remain an active research area in the scientific community. However, these algorithms are not free from certain limitations. Geng et al. (2011) pointed out premature convergence and poor exploitation as the major drawbacks of metaheuristics when facing more complex problems. Thus, more advanced algorithms are needed to achieve satisfactory solutions for resource-leveling problem in modern construction projects.

Symbiotic Organisms Search Algorithm

The SOS algorithm is a new metaheuristic algorithm developed by Cheng and Prayogo (2014). It was inspired by the dependency-based interactions seen among organisms in nature, which are known as symbiosis. Like most population-based metaheuristic algorithms, SOS has the following features: (1) it uses a population of organisms that contains candidate solutions used to seek the global solution over the search space; (2) it has special operators that use the candidate solutions to guide the search process; (3) it uses a selection mechanism to preserve the better solutions; and (4) it requires the proper setting of common control parameters such as population size and maximum number of evaluations.

However, unlike most metaheuristic algorithms that have additional control parameters (e.g., GA has crossover and mutation rates; PSO has inertia weight, cognitive factors, and social factors), SOS requires no algorithm-specific parameters. This is considered an advantage over competing algorithms because SOS does not need to perform parameter tuning . Improper tuning related to algorithm-specific parameters might increase computational time and produce local optima solutions.

In the early stage, a random ecosystem (population) matrix is created, each row representing a candidate solution to the corresponding problem. The number of organisms in the ecosystem, the so-called ecosystem size, is predetermined by the user. The rows in the matrix are called organisms, as in other metaheuristic algorithms. Each virtual organism represents a candidate solution to the corresponding problem or objective. The search begins after the initial ecosystem has been generated. During the search process, each organism benefits from continuous interaction with others in three different phases:

- Mutualism: one organism develops a relationship that benefits itself and the other—a classic example is the interaction between bees and flowers;
- Commensalism: one organism develops a relationship that benefits itself but does not impact the other—an example is the relationship between remora fish and sharks; and
- Parasitism: one organism develops a relationship that benefits itself but harms the other—an example is the plasmodium parasite, which uses its relationship with the anopheles mosquito to transfer between human hosts.

The three phases are adopted from the most common symbioses used by organisms to increase their fitness and survival advantage over the long term. During the interaction, the one that receives a benefit evolves to a fitter organism whereas the one who is harmed perishes. The mechanisms for updating the best organism are conducted after one generation of organisms has completed its three phases. The phases are repeated until the stopping criterion is achieved. The pseudocode shown in Fig. 1 summarizes the basic step of the SOS optimization procedure:

```
1: Initialization (initial ecosystem, set ecosystem size and maximum iteration)
2: For counter=1 to maximum iteration
3:   For each organism in the ecosystem
4:         Mutualism Phase
5:         Commensalism Phase
6:         Parasitism Phase
7:         Update the best organism
8:   End For
9: End For
```

Fig. 1. SOS algorithm pseudocode

Discrete Symbiotic Organisms Search for Multiple-Resources Leveling in Multiple Projects—the DSOS-MRLMP Model

This section describes the newly proposed DSOS-MRLMP model in detail. The SOS algorithm plays an important role as the DSOS-MRLMP's core optimizer. Fig. 2 shows its overall operational architecture. The objective of the DSOS-MRLMP is to minimize daily variation in resource utilization without changing the total project duration.

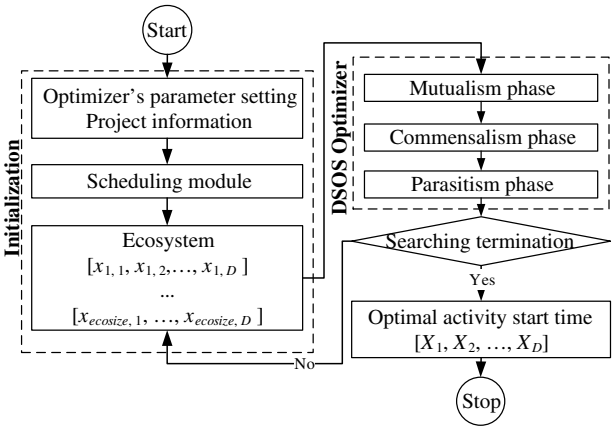

Fig. 2. Flowchart for the DSOS-MRLMP

Initialization

Inputs required by DSOS-MRLMP include activity precedence relationship, activity duration, and resource demand. In addition, the user must provide search engine parameter settings such as the maximum number of search iterations ($G_{\max}$) and the ecosystem size (*ecosize*). The scheduling procedure uses these inputs in the calculation process to obtain the project duration and resource amount required for each activity. With all the necessary information provided, the model is capable of operating automatically without human intervention.

Prior to the search process, a random generator generates an initial ecosystem comprising organisms (feasible solutions):

$$\text{Ecosystem} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_{\text{ecosize}} \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & & \vdots \\ x_{i,1} & x_{i,2} & x_{i,j} & x_{i,D} \\ \vdots & \vdots & & \vdots \\ x_{\text{ecosize},1} & x_{\text{ecosize},2} & \cdots & x_{\text{ecosize},D} \end{bmatrix} \tag{8}$$

where $x_{i,j}$ = uniformly distributed random number between 0 and 1 at the initial step, and is optimized by the SOS algorithm during the search process.

Decision variables for the resource-leveling problem are represented as a vector

$$X = [x_{i,1}, x_{i,2}, \ldots, x_{i,j}, \ldots, x_{i,D}] \tag{9}$$

where D = number of elements in a vector of decision variables in the problem at hand and also = number of noncritical activities in the project networks; index i = ith member in the ecosystem.

Because the original SOS operates with real-value variables, a function is employed to convert those variables from real values to integer values that are constrained in the feasible domain

$$X_{i,j} = \text{round}\{LB(j) + x_{i,j} \times [UB(j) - LB(j)]\} \tag{10}$$

where $X_{i,j}$ = start time of noncritical activity j in the networks at the ith individual of the population; LB(j) and UB(j) = respective early and late start times for activity j.

In multiple-resources leveling in multiple-projects scheduling, two constraint conditions limit the actual start time of all activities: (1) it must be between the early and late start times; and (2) it 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 is more difficult to find. Each dimension of the decision variables is determined in turn. When calculating the actual start time of one activity, the actual start times of all activities in its predecessor set, $T_s(\text{pset}_k, i_k)$, have been computed and the max $[T_s(\text{pset}_k, i_k) + T_s(\text{pset}_k, i_k)]$ has been confirmed simultaneously.

Mutualism Phase

Let X_i be the organism matched to the ith row of the ecosystem matrix. The organism X_i randomly selects organism X_j as its partner from the ecosystem. Organism X_i is associated with the jth row of the ecosystem where j is not equal to i. The mutualistic symbiosis between organisms X_i and X_j is modeled in Eqs. (11) and (12)

$$X_{i\text{new}} = X_i + \text{rand}(0,1) \times (X_{\text{best}} - \text{Mutual\_Vector} \times \text{BF}_1) \tag{11}$$

$$X_{j\text{new}} = X_j + \text{rand}(0,1) \times (X_{\text{best}} - \text{Mutual\_Vector} \times \text{BF}_2) \tag{12}$$

$$\text{Mutual\_Vector} = \frac{X_i + X_j}{2} \tag{13}$$

$$\text{BF}_1 = 1 + \text{round}[\text{rand}(0,1)] \tag{14}$$

$$\text{BF}_2 = 1 + \text{round}[\text{rand}(0,1)] \tag{15}$$

The following observations on the mutualism mathematical model can be made:

- rand(0,1) in Eqs. (11) and (12) is a vector of random numbers between 0 and 1;
- *Mutual_Vector* in Eq. (13) represents the mutual connection between organisms X_i and X_j;
- X_{best} represents the organism with the current highest state of adaptation to the ecosystem;
- Organism X_i might benefit significantly when interacting with organism X_j; at the same time, organism X_j might benefit only slightly when interacting with organism X_i; here, benefit factors (BF_1) and (BF_2) are determined stochastically as either 1 or 2 [Eqs. (14) and (15)], indicating whether an organism partially or fully benefits from the interaction;
- Organisms evolve to a fitter version only if their new fitness is better than their preinteraction fitness; if so, the old X_i and X_j are replaced by $X_{i\text{new}}$ and $X_{j\text{new}}$; this mechanism is similar to greedy selection; and
- For each organism X_i, this interaction counts for two function evaluations.

Commensalism Phase

After the mutualism phase is finished, the organism X_i selects a new partner randomly from the ecosystem, organism X_j. In this case, organism X_i attempts to benefit from the interaction but organism X_j neither benefits nor suffers from it. The commensal symbiosis between organisms X_i and X_j is modeled in Eq. (16):

$$X_{i\text{new}} = X_i + \text{rand}(-1,1) \times (X_{\text{best}} - X_j) \tag{16}$$

Some observations on the commensalism mathematical model can be made:

- rand$(-1,1)$ in Eq. (16) is a vector of random numbers between -1 and 1;
- X_{best} reflects the current highest state of adaptation to the ecosystem, similar to that used in the mutualism phase;
- Organism X_i is updated to $X_{i\text{new}}$ only if its new fitness is better than its preinteraction fitness; and
- For each organism X_i, this interaction counts for one function evaluation.

Parasitism Phase

After the commensalism phase is completed, the organism X_i again randomly selects a new organism from the ecosystem, organism X_j. In parasitism, organism X_i is given a role similar to that of the anopheles mosquito through the creation of an artificial parasite called *Parasite_Vector*. Organism X_j serves as host to *Parasite_Vector*. During the interaction, *Parasite_Vector* tries to kill host X_j and replace it in the ecosystem. Organism X_i may gain a benefit because, by cloning *Parasite_Vector*, its influence in the ecosystem may increase whereas X_j may suffer and die.

The creation of *Parasite_Vector* is described as follows:

1. An initial *Parasite_Vector* is created in the search space by duplicating organism X_i; some decision variables from the initial *Parasite_Vector* are modified randomly to differentiate *Parasite_Vector* from organism X_i;
2. A random number is created within a range from one to the number of decision variables, representing the total number of modified variables;
3. The location of the modified variables is determined stochastically using a uniform random number,which is generated for each dimension; if the random number is less than 0.5, the variable is modified; otherwise, it stays the same; and
4. The variables are modified using a uniform distribution within the search space and *Parasite_Vector* is ready for the parasitism phase.

Both *Parasite_Vector* and organism X_j are then evaluated to measure their fitness. If *Parasite_Vector* has a better fitness value, it kills organism X_j and assumes its position in the ecosystem. If the fitness value of X_j is better, X_j has immunity from the parasite and *Parasite_Vector* can no longer live in that ecosystem. For each organism X_i, this interaction counts for one function evaluation.

Stopping Condition

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

Case Study

A case study adapted from Guo et al. (2009) was used to demonstrate the capability of the newly developed DSOS-MRLMP model. In this case study, an enterprise had to start two projects with the same total project duration. Fig. 3 shows the precedence relationships of the network projects. Each activity in both projects used three resources (R_1: human; R_2: fund; R_3: equipment) and had a certain duration D, indicated above the arrow line in the figure.

Based on the importance of each resource, the analytical hierarchy method made a pairwise comparison of each resource. The comparison matrix was obtained as follows:

$$\begin{matrix} R_1 \\ R_2 \\ R_3 \end{matrix} \begin{bmatrix} 1 & 3 & 5 \\ 3^{-1} & 1 & 3 \\ 5^{-1} & 3^{-1} & 1 \end{bmatrix}$$

Consistency inspection demonstrated that this comparison matrix was acceptable. The computational process for the weights of each resource followed these steps: (1) sum up each column of the comparison matrix; (2) divide each corresponding element by its sum-up value; and (3) take the average of each row to obtain the weights. The weights for each resource were thus set approximately as $w_1 = 0.637, w_2 = 0.258$, and $w_3 = 0.105$. Consequently, the objective function for the case study was calculated as follows:

$$\min \text{ RI} = \frac{1}{18}\sum_{t=1}^{18}\{0.637[\text{SR}_1(t) - \overline{\text{SR}_1(t)}]^2 + 0.258[\text{SR}_2(t) - \overline{\text{SR}_2(t)}]^2 + 0.105[\text{SR}_3(t) - \overline{\text{SR}_3(t)}]^2\}$$

$$\text{s.t.} \begin{cases} 0 \le T_s(A_1) \le 7 & 0 \le T_s(I_1) \le 15 \\ 0 \le T_s(B_1) \le 3 & 0 \le T_s(A_2) \le 9 \\ T_s(B_1) + 5 \le T_s(C_1) \le 8 & 0 \le T_s(C_2) \le 15 \\ 0 \le T_s(F_1) \le 6 & 5 \le T_s(D_2) \le 9 \\ T_s(B_1) + 4 \le T_s(G_1) \le 10 & 5 \le T_s(G_2) \le 7 \\ 0 \le T_s(H_1) \le 3 & 5 \le T_s(H_2) \le 13 \end{cases}$$

Optimization Result for the DSOS-MRLMP

The DSOS-MRLMP model significantly reduces fluctuation in resource use. This study used parameters for the DSOS optimizer based on proposed values from the literature and several experiments, as shown in Table 1 (Cheng and Prayogo 2014). Fig. 4 shows the network resource profile for the projects at initialization and after leveling using DSOS-MRLMP optimization.

Result Comparisons

Three different algorithms were used to verify the comparative performance of the DSOS-MRLMP model. These were DE (Storn and Price 1997), PSO (Clerc 2006), and GA (Haupt and Haupt 2004). For comparison purposes, all four algorithms used an equal number of function evaluations, had population sizes of 100, and used a maximum of 200 generations. 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, and c_2, were both chosen as 2.05, and the inertia factor w was set in the range 0.3–0.7. DE control parameters were set as 0.5 and 0.8 for mutant factor F and crossover probability Cr, respectively. Fifty independent runs were carried out for all experiments.

Table 2 lists the optimal results—that is, the optimal noncritical activity start times obtained from the proposed model and other benchmark algorithms. In Table 2, RI_m is the resource intensity for a single resource m

$$\min \text{ RI} = \frac{1}{18}\sum_{t=1}^{18}[R_m(t) - \overline{R_m(t)}]^2, \qquad \overline{R_m} = \frac{1}{18}\sum_{t=1}^{T} R_m(t)$$

As shown in the table, the optimal resource intensities (RIs) obtained by DSOS-MRLMP were, respectively, 94.9%, 2.0%, 6.9%, and 14.4% less than the initial schedule, DE, PSO, and GA. Fig. 5 shows the resource profile after optimization by each algorithm.

To evaluate the stability and accuracy of each algorithm, optimization performance was expressed in terms of best result found, average result, standard deviation, and worst result after 50 runs (Table 3). The best and worst results demonstrate the capacity of each algorithm to find the optimal solution for all performance measurement metrics. Average and standard deviation are two additional characteristics that describe solution quality. Standard deviation occurs when algorithms are not able to generate optimal solutions in all executions.

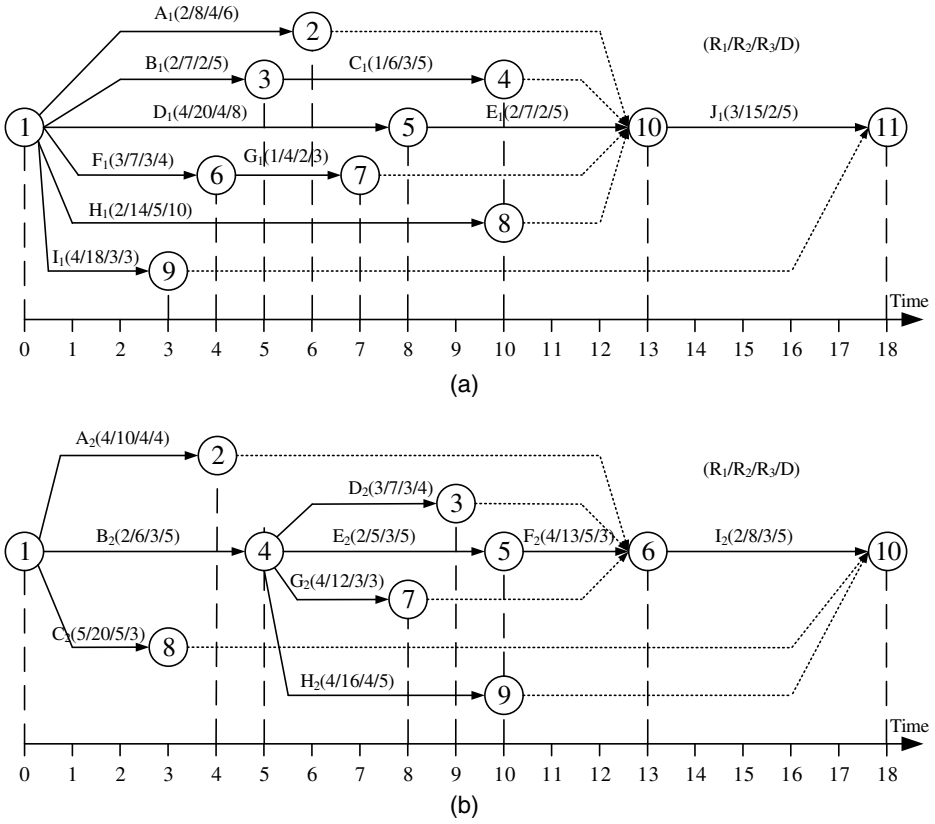

Fig. 3. Networks of two projects: (a) project 1; (b) project 2

Table 1. Settings for DSOS-MRLMP Parameters

Input parameter	Notation	Setting
Number of decision variables	D	12
Population size	NP	100
Amplification coefficient	λ	30
Maximum generation	G_{max}	200

As shown in Table 3, the performance of DSOS-MRLMP is competitive in terms of accuracy and stability. It is clearly shown that the model can find optimal solutions in fitness function. Furthermore, in terms of average results DSOS-MRLMP performs the best of the considered algorithms because it generates the lowest average fitness solution, with a value of 4.739 and a deviation value of 0.291. Fig. 6 shows the best fitness values from the different approaches by number of iterations.

Statistical Test

Hypothesis tests were performed to further demonstrate the superiority of the DSOS-MRLMP over the benchmark algorithms. Because all indicator comparisons demonstrate that DE performs better on average than either PSO or GA, the hypothesis tests only considered DSOS-MRLMP and DE. A one-tailed t-test with equal sample sizes and unequal and unknown variances analyzed DSOS-MRLMP versus standard DE in terms of resource intensity (RI) (Table 3) to determine which of the following hypotheses holds:

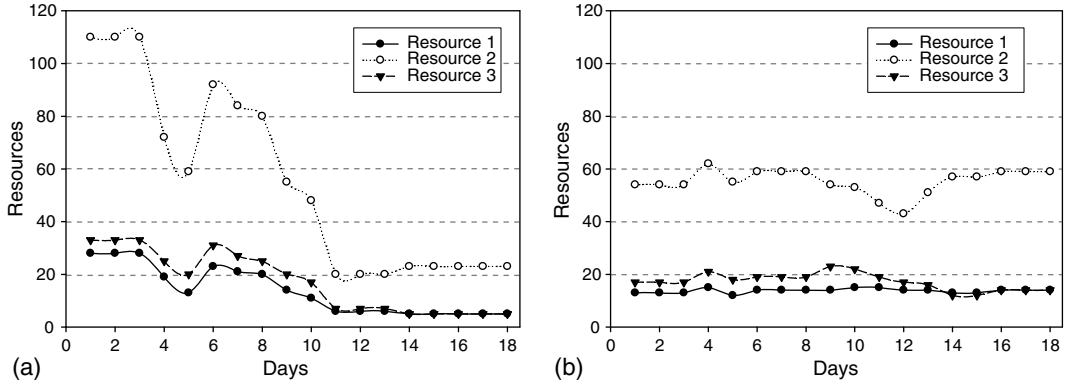

Fig. 4. Resource profile of projects: (a) resource demand of initial network; (b) resource demand after DSOS optimization

Table 2. Comparison of Optimal Performance for Algorithms

Algorithms	RI	RI_1	RI_2	RI_3	Actual start time of noncritical activities											
					A_1	B_1	C_1	F_1	G_1	H_1	I_1	A_2	C_2	D_2	G_2	H_2
Initial	89.46	76.95	1,169	123.8	0	0	5	0	4	0	0	0	0	5	5	5
GA	5.327	1.06	13.76	9.17	0	3	8	0	9	0	15	8	12	9	6	13
PSO	4.897	0.84	26.65	7.95	0	0	8	6	10	3	12	0	15	8	5	13
DE	4.652	0.84	21.97	5.73	0	3	9	0	10	0	12	8	15	9	6	13
DSOS-MRLMP	4.558	0.62	21.31	9.51	3	0	8	0	8	0	12	8	15	9	5	13

Note: RI = resource intensity.

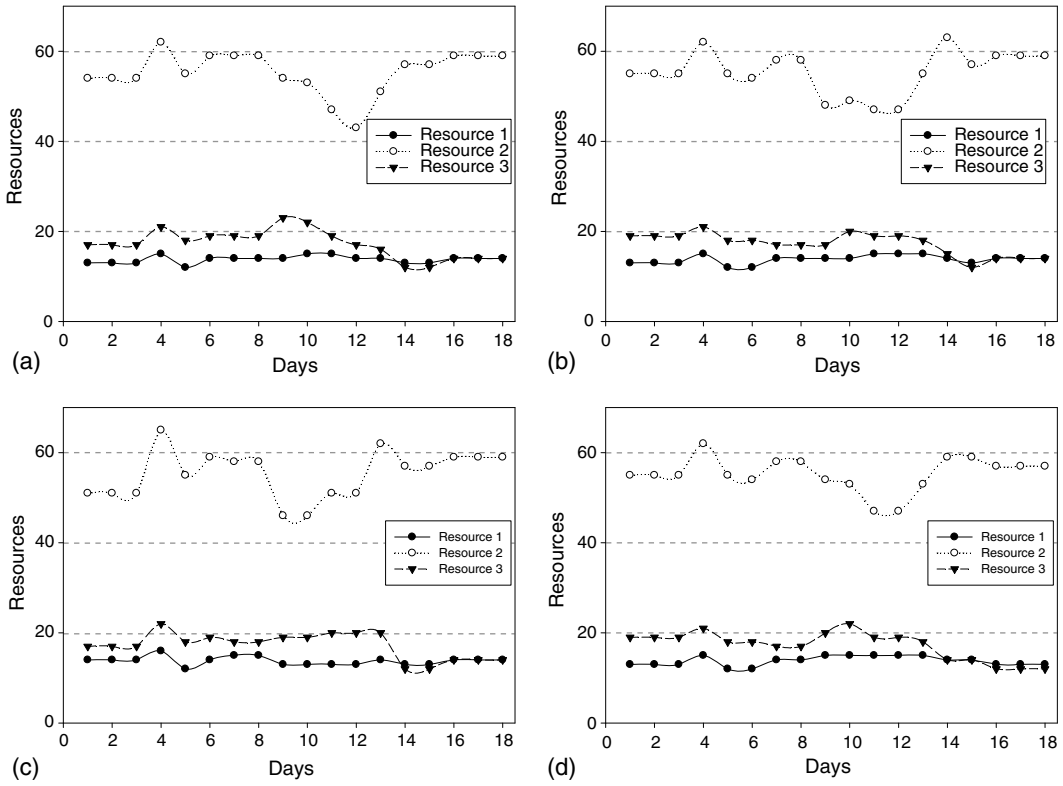

Fig. 5. Resource profile of projects by different algorithms: (a) resource demand optimized by DSOS optimizer; (b) resource demand optimized by DE optimizer; (c) resource demand optimized by PSO optimizer; (d) resource demand optimized by GA optimizer

Table 3. Comparison of Results for the DSOS-MRLMP and Benchmarked Algorithms

Evaluation indicator	Performance measurement	GA	PSO	DE	DSOS-MRLMP
Fitness value	Best	5.327	4.897	4.652	4.558
	Average	6.832	6.154	5.346	4.739
	Standard	1.979	0.864	0.501	0.291
	Worst	13.385	10.038	6.506	5.518

- H_0: There is no difference in the RI of the DSOS algorithm and that of the standard DE algorithm.
- H_1: The DSOS algorithm is significantly better than the standard DE algorithm.

The following hypothesis tests were used:

- DSOS-MRLMP: $s_1 = 0.291$; DE: $s_2 = 0.501$; $n_1 = n_2 = n = 50$:

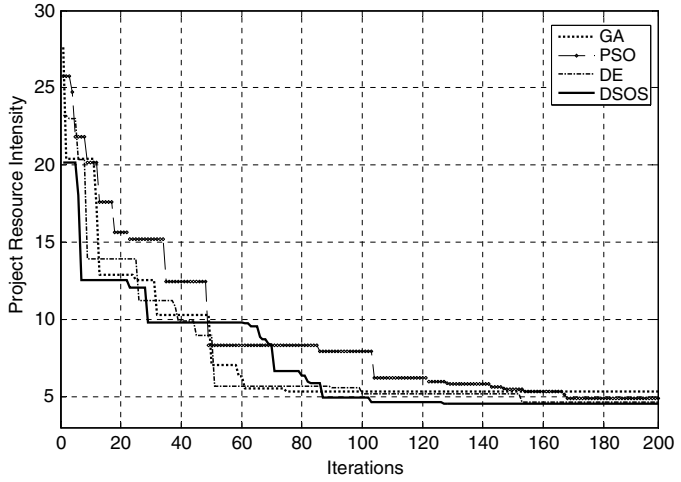

Fig. 6. Best project RI curves for the algorithms

$$v = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}} = \frac{(0.291^2/50 + 0.501^2/50)^2}{\frac{(0.291^2/50)^2}{50 - 1} + \frac{(0.501^2/50)^2}{50 - 1}}$$
$$= 78.7 \text{ (closest to 79)}$$

- Critical value: with significant level of t-test $\alpha = 0.05$; $\nu = 79$; the following is obtained:

$$t_{\alpha;\nu} = t_{0.05;79} = 1.664$$

- Statistical test:

$$t = \frac{(\bar{x}_1 - \bar{x}_2)}{\sqrt{s_1^2/n_1 + s_2^2/n_2}} = \frac{(4.739 - 5.346)}{\sqrt{0.291^2/50 + 0.501^2/50}} = -7.406$$
$$< -1.664 = -t_{0.05;79}$$

where n = sample size (number of experimental runs); ν = degrees of freedom used in the test; and s_1^2 and s_2^2 = unbiased estimators of the variances of the two samples (DSOS-MRLMP and DE).

The denominator of t is the standard error of the difference between two means $\bar{x}_1$, and $\bar{x}_2$ (average).

The statistical test value is smaller than the critical value. Therefore, H_0 is rejected. The proposed DSOS-MRLMP is thus demonstrated to be statistically superior to the standard DE in terms of resource intensity.

Conclusions

This paper used DSOS to solve the problem of multiple-resources leveling in multiple-projects scheduling (MRLMP). An application example was analyzed to illustrate the effectiveness of the proposed model and to demonstrate its capabilities in generating an optimal schedule that eliminates undesirable resource fluctuations and resource idle times.

Obtained results indicate the excellent performance of the DSOS-MRLMP algorithm in solving the MRLMP problem. It is the only algorithm that yields the optimal actual start time of noncritical activities with the best RI, 4.558. Furthermore, it significantly outperforms GA, DE, and PSO. The closet competing algorithm to the proposed algorithm is DE. When these two algorithms are compared, DSOS-MRLMP has an approximately 2.0%, 11.4%, 41.9%, and15.2% better solution than DE in terms of best solution, average, standard deviation, and worst solution, respectively. Moreover, DSOS-MRLMP produces a significantly better result than DE in terms of RI, verified by the one-tailed t-test.

The search strategy using organism interactions was important to the extraordinary performance of the proposed algorithm. This strategy differs significantly from previous metaheuristic algorithms such as GA, PSO, and DE because it simulates natural patterns using the mutualism, commensalism, and parasitism phases of the SOS algorithm to gradually improve candidate solutions. These three phases are simple and require only a few additional lines of code on *MATLAB* platforms. Another major advantage of SOS over competing algorithms is the small number of parameters that must be tuned.

The DSOS algorithm has broad application potential because it is easily modifiable for solving many other classes of single-objective optimization problems in the construction management field, such as resource allocation and resource constraint. Moreover, resource-leveling problems in the realm of minimization of total project cost are frequently encountered in construction management. Trade-offs between time and cost are necessary to improve overall project benefits. Further work is necessary to address these issues so that DSOS can be applied to the resolution of complicated resource-leveling problems that consider multiobjective optimizations. Extending the current DSOS from a single-objective to a multiobjective format is an interesting direction for further research.

References

Alsayegh, H., and Hariga, M. (2012). "Hybrid meta-heuristic methods for the multi-resource leveling problem with activity splitting." *Autom. Constr.*, 27, 89–98.

Cheng, M.-Y., and Prayogo, D. (2014). "Symbiotic organisms search: A new metaheuristic optimization algorithm." *Comput. Struct.*, 139, 98–112.

Cheng, M.-Y., and Tran, D.-H. (2014). "Two-phase differential evolution for the multiobjective optimization of time-cost tradeoffs in resource-constrained construction projects." *IEEE Trans. Eng. Manage.*, 61(3), 450–461.

Cheng, M.-Y., Tran, D.-H., and Wu, Y.-W. (2014). "Using a fuzzy clustering chaotic-based differential evolution with serial method to solve resource-constrained project scheduling problems." *Autom. Constr.*, 37, 88–97.

Christodoulou, S., Ellinas, G., and Michaelidou-Kamenou, A. (2010). "Minimum moment method for resource leveling using entropy maximization." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000149, 518–527.

Clerc, M. (2006). *Particle swarm optimization*, ISTE, London.

Damci, A., Arditi, D., and Polat, G. (2013). "Multiresource leveling in line-of-balance scheduling." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000716, 1108–1116.

Damci, A., and Polat, G. (2014). "Impacts of different objective functions on resource leveling in construction projects: A case study." *J. Civ. Eng. Manage.*, 20(4), 537–547.

Das, S., and Suganthan, P. N. (2011). "Differential evolution: A survey of the state-of-the-art." *IEEE Trans. Evol. Comput.*, 15(1), 4–31.

Doulabi, S. H. H., Seifi, A., and Shariat, S. Y. (2011). "Efficient hybrid genetic algorithm for resource leveling via activity splitting." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000261, 137–146.

El-Rayes, K., and Jun, D. H. (2009). "Optimizing resource leveling in construction projects." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000097, 1172–1180.

Geng, J.-Q., Weng, L.-P., and Liu, S.-H. (2011). "An improved ant colony optimization algorithm for nonlinear resource-leveling problems." *Comput. Math. Appl.*, 61(8), 2300–2305.

Guo, Y., Li, N., and Ye, T. (2009). "Multiple resources leveling in multiple projects scheduling problem using particle swarm optimization." *Proc., 5th Int. Conf. on Natural Computation, 2009. ICNC '09*, IEEE, 260–264.

Hariga, M., and El-Sayegh, S. (2011). "Cost optimization model for the multiresource leveling problem with allowed activity splitting." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000251, 56–64.

Harris, R. B. (1990). "Packing method for resource leveling (pack)." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)0733-9364(1990)116:2(331), 331–350.

Haupt, R. L., and Haupt, S. E. (2004). *Practical genetic algorithms*, Wiley, Hoboken, NJ.

Hegazy, T. (1999). "Optimization of resource allocation and leveling using genetic algorithms." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)0733-9364(1999)125:3(167), 167–175.

Hossein Hashemi Doulabi, S., Seifi, A., and Shariat, S. (2011). "Efficient hybrid genetic algorithm for resource leveling via activity splitting." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000261, 137–146.

Koulinas, G. K., and Anagnostopoulos, K. P. (2013). "A new tabu search-based hyper-heuristic algorithm for solving construction leveling problems with limited resource availabilities." *Autom. Constr.*, 31, 169–175.

Leu, S.-S., Yang, C.-H., and Huang, J.-C. (2000). "Resource leveling in construction by genetic algorithm-based optimization and its decision support system application." *Autom. Constr.*, 10(1), 27–41.

Martinez, J., and Ioannou, P. (1993). "Resource leveling based on the modified minimum moment heuristic." Proc., 5th Int. Conf., Computing in Civil and Building Engineering, ASCE, Reston, VA, 287–294.

Ponz-Tienda, J. L., Yepes, V., Pellicer, E., and Moreno-Flores, J. (2013). "The resource leveling problem with multiple resources using an adaptive genetic algorithm." *Autom. Constr.*, 29, 161–172.

Savin, D., Alkass, S., and Fazio, P. (1996). "Construction resource leveling based using neural networks." *Can. J. Civ. Eng.*, 23(4), 917–925.

Son, J., and Skibniewski, M. J. (1999). "Multiheuristic approach for resource leveling problem in construction engineering: Hybrid approach." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)0733-9364(1999)125:1(23), 23–31.

Storn, R., and Price, K. (1997). "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces." *J. Global Optim.*, 11(4), 341–359.

Tang, Y., Liu, R., and Sun, Q. (2014). "Two-stage scheduling model for resource leveling of linear projects." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000862, 04014022.

Wu, C., Wang, X., and Lin, J. (2014). "Optimizations in project scheduling: A state-of-art survey." *Optimization and control methods in industrial engineering and construction*, H. Xu and X. Wang, eds., Vol. 72, Springer, Dordrecht, Netherlands, 161–177.

Yan, L., Sheng-Li, Z., Xi-Kai, D., and Shu-Quan, L. (2005). "Optimization of resource allocation in construction using genetic algorithms." *Proc., 2005 Int. Conf. on Machine Learning and Cybernetics*.