Discrete time-cost tradeoff model for optimizing multi-mode construction project resource allocation

The resource-constrained project scheduling problem has received broad attentions and was evolved into various sub-problems such as resource-constrained discrete time-cost tradeoff problem. The resource leveling problem was proposed to reduce the resource fluctuation and was always studied independently with RCPSP. This research proposed a new model which integrates the resource leveling problem and resource-constrained time-cost tradeoff problem. The evolutionary multi-objective optimization technique, strength Pareto evolutionary approach II (SPEA II) was applied to calculate the Pareto front of time and cost. The resource leveling measured by the metric, resource release/re-hiring, was converted to resource cost. The analysis of the time complexity of the model showed that the runtime of the algorithm was polynomial times of the number of activities. The results of case testing showed that the model was reasonably accurate in comparison with a proposed baseline model.

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

Discrete time-cost tradeoff model for optimizing multi-mode construction project resource allocation

Shuangshuang Nie, Jihong Gao
Shaoxing University, Shaoxing, China

Abstract:
The resource-constrained project scheduling problem has received broad attentions and was evolved into various sub-problems such as resource-constrained discrete time-cost tradeoff problem. The resource leveling problem was proposed to reduce the resource fluctuation and was always studied independently with RCPSP. This research proposed a new model which integrates the resource leveling problem and resource-constrained time-cost tradeoff problem. The evolutionary multi-objective optimization technique, strength Pareto evolutionary approach II (SPEA II) was applied to calculate the Pareto front of time and cost. The resource leveling measured by the metric, resource release/re-hiring, was converted to resource cost. The analysis of the time complexity of the model showed that the runtime of the algorithm was polynomial times of the number of activities. The results of case testing showed that the model was reasonably accurate in comparison with a proposed baseline model.

Keywords: Scheduling; Resource-constrain project scheduling; Resource leveling; Genetic algorithm; Strength Pareto evolutionary approach II

1 Introduction

The widely used critical path method (CPM) does not consider resource availability. Resource-constrained project scheduling problem (RCPSP) was proposed to optimize scheduling under resource constraints [1]. RCPSP was defined from three aspects of characteristics: resource environment; activity characteristics; and objective function [2].

The RCPSP can be further divided into the single-mode and the multi-mode RCPSPs in terms of the number of modes in which resources are used or consumed [2]. Simulation has been used to solve building design and construction problem [3, 4].When the activity durations become variable in different modes, the combination of different modes generates varied costs. In this sense, the RCPSP may be evolved into a discrete time-cost tradeoff problem, which can be subdivided into the deadline problem and the budget problem [5]. The deadline problem is to minimize resource cost within the limit of the maximum project duration while the budget problem is to minimize the project duration within a maximum cost [6].

The methods for solving the discrete time-cost tradeoff problem are mainly divided into two categories: exact algorithms and heuristic algorithms. Since the RCPSPs are combinatorial problems in nature, the exact algorithms, such as branch and bound algorithm, use mathematical programming to find the optimum solutions [7]. However, the exact algorithms are inefficient in the case of relative large projects with many activities or variables because the time-cost tradeoff is a NP problem that is hard to solve [8]. Heuristic methods and machine learning techniques have been successfully used to solve building and construction problems [9, 10].

Metaheuristic algorithms, especially genetic algorithms (GAs), were widely applied to solve the building design and project scheduling problems [4, 11]. The objective function of the GA were usually set to minimizing the project duration [12-14]. The chromosome representation of GAs could be permutation based, priority rule based and priority value. Serial and parallel scheduling techniques were adopted to convert the chromosome into the actual schedule [11, 15, 16].

The time-cost tradeoff problem can be divided into two types: continuous and discrete. The second type is also a sub-problem of the RCPSP. Early researches did not consider resource constraints when developing GA based approaches to solve time-cost tradeoff problems [17-19]. In recent years, some researchers tried to investigate multi-mode discrete time-cost tradeoff using GAs [20-22].

Resource fluctuation is mainly studied in the scope of the resource leveling problem, which describes the process of reducing the fluctuations in resource usage over the project duration. Undesired resource fluctuation may cause

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

inefficient and costly implementation of construction, for example, frequently rehiring and releasing workers, lowering productive level and interruption of learning curve effects [23], incurring indirect costs for training cost of new workers [24]. These implicit costs incurred by undesired resource fluctuations may account for a large amount of resources costs. Exact methods and heuristic methods were used to solve this problem [24]. However, the exact methods are still inefficient for addressing projects with a larger number of activities. Heuristic algorithms, such as particle swarm optimization (PSO) and GA were adopted to solve leveling problems under different circumstances [23-27]. The algorithms search possible locations of noncritical activities to smooth the resource usage. The objective is usually to achieve uniform resource consumption. However, the resource leveling metrics proposed in these models measured and penalized the difference between fluctuating resource consumption and a predefined desirable shape [23]. Alternative shapes of resource profiles which may have more efficient resource utilization may be penalized. Thus, in El-Rayes and Jun [23], the authors developed two innovative metrics without predetermining the shape of the resources, release and re-hire (RRH) and resource idle days (RID). The RRH metric can be useful when the release and rehire of resource are allowed in projects while the RID metric may be used where additional idle resources are required to kept on site during low demand periods.

Resource leveling and resource-constrained project scheduling problems are inherently interrelated. A certain schedule having a higher resource cost may have a lower resource fluctuation. However, the two problems were usually studied independently [26]. Only a few integrated models were developed to solve two problems simultaneously [28]. In Chan and Chua [25], authors presented a commercial GA package Genesis which was formulated to consider both the problems. The objective was to minimize the deviations of required resources from the available resource profiles. Resource leveling was addressed by calculating the resource underutilization and overutilization. The paper Hu and Flood [2] developed an integrated scheduling method to minimize the project duration and resource fluctuation by using the strength Pareto evolutionary approach II (SPEA II) which outperformed several other multi-objective optimization techniques in solving the resource-constrained project scheduling problem. In Hu and Flood [2]Hegazy [27], authors combined resource allocation and leveling by using a GA technique. The objective was to minimize the total project duration under resource constraints and the appropriate moments of selected resources important to resource leveling. In Leu and Yang [29], authors integrated time-cost tradeoff, resource constraint and resource leveling based on GA. The sum of the absolute differences between actual resource usage and average resource usage was used as a metric of measuring the resource leveling. The ideal resource profile was predefined as a rectangular shape. In Lucko [30] the authors used GA for optimizing linear scheduling project. Only project duration was address.

The literature review shows that PCPSP has not been fully explored in the integrated models. The researches mainly used a single objective function to minimize project duration, cost, or deviation between resource usage and the defined value. Resource leveling tended to be address in these models by adding a constraint. In addition, only single-mode RCPSP was integrated with the resource leveling problem. Therefore, this research aimed to develop a model that could solve the resource leveling problem and multi-mode RCPSP (more specifically, resource-constrained discrete time-cost tradeoff problems) simultaneously by using a multi-objective optimization technique in order to achieve the optimum solutions.

2 Problem description

The mathematical description of the problem was defined in this section. The problem was defined from three characteristics: resource environment, activity characteristics and objective functions. Activities are subject to precedence constraint. The start time (ST) of an activity should be larger than the finish time (FT) of the preceding activities. d_{il} denotes the duration of an activity i in a mode l. The total resource set K includes all the resource types of a project. The capacity of a resource type $k_m \in$ K per time period is set to R_m. The quantity of resource m used by an activity i in a time period is r_{iml} if the activity is performed in a mode l.

The problem is solved based on a multi-objective optimization technique whose objective functions are to minimize the resource cost and the total project duration. The cost is split up to

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

two parts: resource usage cost (RUC) and resource fluctuation cost (RFC).

2.1 Resource usage cost (*RUC*)

The resource usage cost (*RUC*) is defined as the cost of resources used by all the activities in a given activity modes. For a multi-mode RCPSP, an activity may have different modes each of which would require varied duration and resource amount. *RUC* denotes the total resource cost in the project duration *T* (see Eqs. (1) and (2)). C_m denotes the unit cost of the resource *m*. $RU_{m,t}$ is used to calculated amount of resource *m* used at a particular time *t*. Z_t is equal to 1 if an activity *i* is performed at a time period *t* and o otherwise.

$$RUC = \sum_{t=1}^{T} RU_{m,t} \times C_m$$

$$RU_{m,t} = \sum_{i=1}^{n} (Z_t \times r_{iml})$$

2.2 Resource fluctuation cost (*RFC*)

When resources are newly added or dismissed, additional costs will incur. Such costs may include training, transportation and bidding costs. El-Rayes and Jun [23] proposed two metrics: release and re-hire (RRH) and resource idle days (RID), to measure the level of the resource fluctuation. The RRH and RID are calculated using Eqs. (3) and (4) [23].

$$RRH = H - MRD = \frac{1}{2} \times \left[r_1 + \sum_{t=1}^{T-1} |r_1 - r_{t+1}| + r_T \right] - Max(r_1, r_2, ..., r_T)$$

Eq. (3)

$$RID = \sum_{t=1}^{T} \left[Min\{Max(r_1, r_2, ..., r_t), Max(r_t, r_{t+1}, ..., r_T)\} - r_t \right]$$

Eq. (4)

Where:
H=total increases in the daily resource demand;
T=total project duration;
r_t = resource demand on day t;
MRD=maximum resource demand during the entire project duration.

The two metrics are agreeable though they are used in different circumstances. In this research, the metric of measuring the resource leveling RFC was calculated based on the metric RRH. RFC is calculated by Eq. (5). UHC_m denotes the unit cost of hiring and releasing a resource type *m* and RRH_m denotes the amount of a hired or increased resource *m*. The cost of MRD is alse added to the RFC to minimize the resource demand.

$$RFC = UHC_m \bullet RRH_m + C_m \bullet MRD$$

Eq. (5)

3 Model development

The model consists of four modules: the evolutionary multi-objective optimization (EMO), the project scheduling, the resource leveling and the local search modules (see Fig. 1). The model begins and ends with the EMO module. The fitness values of the EMO are calculated based on time and cost fed into by the project scheduling and resource leveling modules. The local search module is to further optimize the results calculated by the EMO module.

3.1 The EMO module

This module searches the solution space to find the Pareto optimal. If the objective is to minimize $F_i(x)$, a point (i.e., a solution), x' $\in$ X, is Pareto optimal if there does not exist another point, x $\in$ X, such that F(x) $\leq$ F(x'), and Fi (x) <Fi (x') for at least one function. This definition is illustrated in Fig. 1. The objective functions are to minimize the F_1 and F_2, and the solid dark points denote Pareto optimal (i.e., nondmoninated). The white points are non Pareto optimal (i.e., dominated) points because at least one dark points have larger values in terms of F_1 and F_2.

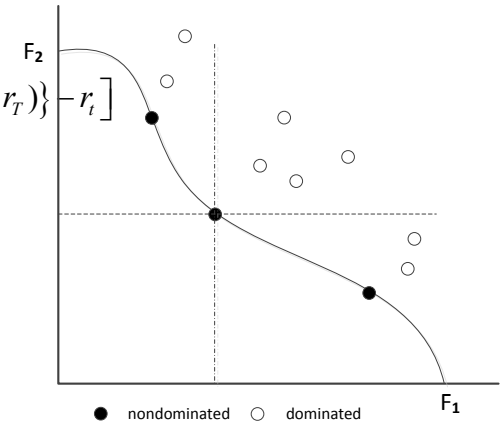


Fig. 1 Illustration of Pareto optimal

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

SPEA II [31] was developed based on the natural evolutionary principle. SPEA II conducts crossover and mutation as shown in Fig. 1. Unlike the single-objective GA algorithm, SPEA II uses the environmental selection scheme to preserve the Pareto optimal. Pareto-based fitness assignment is used to identify nondominated vectors from the current population. The three high level goals are achieved in SPEA II [32].

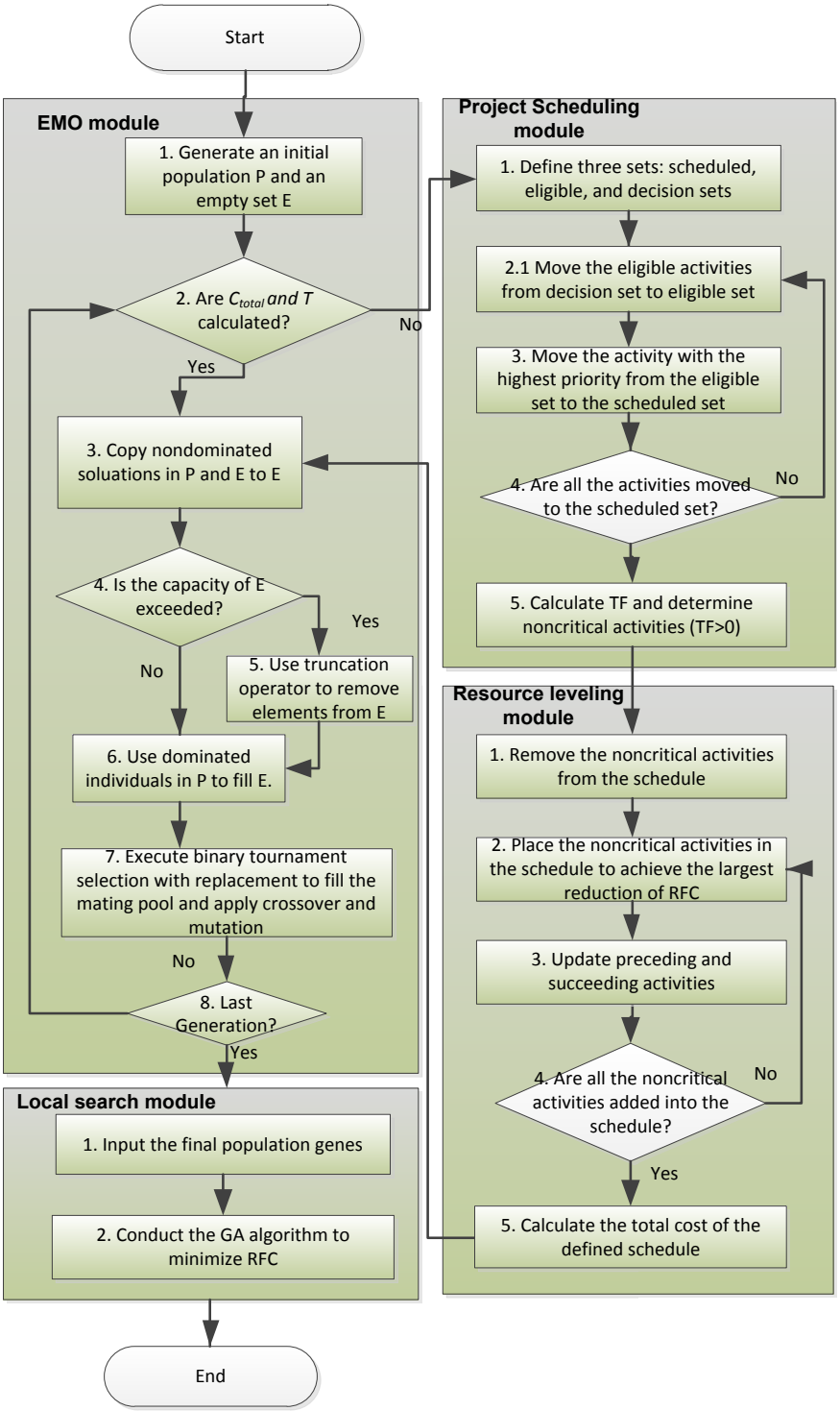


Fig. 2. Flow of the model

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

(1) Preserving nondominated points at each generation.
(2) Generating the known Pareto front and determining the nondominated sets and converging to the true Pareto front.
(3) Generating a uniform distribution across the Pareto.

The line in Fig. 2 can be imagined as a Pareto front line. The EMO algorithm needs to ensure known Pareto optimal is evenly distributed along the Pareto front line and to avoid cluster. SPEAII uses an external archive containing nondominated solutions previously found (Goal 1). Nondominated individuals are copied to the external nondominated set (see Steps 4-6 in the EMO module of Fig. 1). SPEA II has an enhanced archive truncation method that ensures to preserve the boundary solutions. The fitness assignment strategy (see Step 2 in Fig. 1) and GA operations (Step 7 in Fig. 1) considers both closeness to the true Pareto front (Goal (2)) and even distribution of solutions (Goal (3)). The nondominated points are also preserved based on the fitness values.

Application of the SPEA II algorithm into this research requires specifying some parameters and variables. The following sections introduced methods of generating the initial population (see Step 1 in Fig. 1), calculating the fitness values (see Step 2 in Fig. 1), and defining crossover and mutation strategies (see Step 7 in Fig. 1).

3.1.1 Initial population

The scheme of chromosome representation should be first defined before generating the initial population. There are three encoding methods: permutation based, priority value based and priority rule based methods [14]. Hartmann [33] applied permutation based encoding method into the multi-mode RCPSP. In the permutation based method, each individual is given by a sequence of activity. In the beginning, the dummy start activity is added into the schedule, and then each activity is added into the schedule with earliest start time according to the activity sequence specified by the individual. For priority value based methods, each activity is given by a unique priority value and each time a precedence feasible activity with the highest priority value is selected to add into the schedule. Both the permutation based and priority value based methods can the serial scheduling scheme to convert the chromosome to genotype (i.e., schedule).

Because this problem defined in this research has a broader scope than the classic RCPSP and the resource leveling module also needs to employ the gene representation, the priority based encoding approach was adopted.

The specific encoding approach is illustrated in Fig. 3. The first section of chromosome denotes the priority values which are unique integers in the range from 1 to n (i.e., the total number of activities). The second section denotes the mode of activities. When the two sections are specified, chromosome gene values would be fed into the resource leveling module to generate genotype (i.e., schedule).

A_1	...	A_n
M_1	...	M_n

Priority values

Mode of Activities

n:The number of activity

Fig. 3 Chromosome structure of GA module

3.1.2 Fitness assignment

SPEA II calculated fitness using Eq (9) [31]. R(i) is the function of "strength" $S(j)$ calculated by considering for each individuals that the number of individuals which it dominates to and that the number of individuals which it dominates. The values of S(j) can be derived from cost and project duration. SPEA II uses a nearest neighbor density estimation technique to calculate D(i) in the fitness values and to ensure the points are evenly distributes along the know Pareto front and to avoid cluster.

The fitness values of the nondominated individuals are less than 1.

$$F(i) = R(i) + D(i)$$

Eq. (6)

$$R(i) = \sum_{j \in P_t + \bar{P}_t} S(j)$$

Eq. (7)

$$D(i) = \frac{1}{\sigma_i^k + 2}$$

Eq. (8)

Where:

$F(i)$ = Fitness value for solution i

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

$R(i)$ = Raw fitness value calculated by summation of the strength values $S(j)$ of all the individuals j that dominate the individual i. $S(j)$ is equal to the number of individuals that the individual j dominates.
$D(i)$ = Density estimator calculated by inverse of the distance to the k-th nearest neighbor σ_i^k

3.1.3 Crossover and mutation operations
The crossover method of the activity mode uses the standard crossover method. However, for the activity priority values, the standard one-point crossover method cannot be used because the priority values should be kept unique in different gene positions. The crossover method developed by Hartmann (1998) was adopted to implement the crossover operation of the first section of the chromosome. The method starts with selecting a random point in the parent chromosome. Then scan the parent genes to select unique priority values that do not exist in the child chromosome [14]. For instance, in Fig. 4, the left part of the gene values in child 1 comes from parent 1 and the right part is from the parent 2 by left-to-right scan. The scan makes the right part of child 1 takes the first three gene values which are different from the left part of child 1. Similarly, the left part of child 2 takes genes from parent 2 and the right part of genes are taken from parent 1 by scanning different genes.

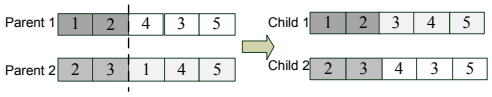


Fig. 4 Crossover of the first section of a chromosome

The mutation of the priority values of activities is achieved by swapping two randomly selected genes in a chromosome. For the second section of the chromosome, the mutation operator modifies modes at each position with an equal probability. If a position is selected, a random value within the range of the total number of modes is selected.

3.2 The project scheduling module

When the activity and mode lists are fed into the module. The schedule would be built up in the project scheduling module. Activities are scheduled according to the serial scheme scheduling approach [15]. Three activities sets are established, i.e., scheduled set, eligible set and decision set (see step 1 in Fig. 1). At first, all the activities are placed in the decision set. The decision set contains all the activities to be scheduled. The scheduled activity set contains activities that have already been scheduled (i.e., starting and finishing time is determined).The activities whose precedent activities are scheduled and included in the scheduled set are then moved to the eligible set from the decision set. The eligible set contains all the activities that are allowed to be scheduled because the precedent activities have been added into the schedule chart. Several activities are moved to the eligible set simultaneously. Only the activity in the eligible set showing the largest priority value is selected to be moved to the scheduled set from the eligible set. This activity is added to the project schedule chart and the starting time is adjusted to meet the resource constraint. This process is repeated until all the activities in the decision set are moved to the schedule set. This whole process is also called the forward pass.

The above-mentioned process starts from the first activity. The resource and precedent constraints are kept feasible in this process. On the other hand, it is possible to start from the last dummy activity. An activity is moved from decision set to eligible set when its succeeding activities are added to the schedule set. This process is regarded as backward pass.

The last task is to calculate the total float (TF) for each activity. The following method is used: the backward pass is used to ensure the activities to finish as late as possible. Then, by deducting the latest finish time by the earliest finish time, the TF of each activity is then calculated. This process may lead to the change of the total project duration. The reason is that some activities are constrained not only by the precedence but also resource availability. Shifting an activity would possibly cause an activity in the critical path to become a noncritical activity. Therefore, the final project duration should be re-calculated at the end of the resource leveling module by deducting the finish time of the last dummy activity by the start time of the first dummy activity.

3.3 The resource leveling module

This module would calculate the time and cost needed by the fitness calculation of the EMO module. The first task is to adjust the noncritical

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

activities (i.e., activities whose TF is larger than zero) to reduce the RFC. Where to locate these noncritical activities in the schedule is the critical work in the process of reducing RFC.

Reducing RFC can be done by the reduction of RRH + MDR. The noncritical activities were removed from the schedule and formed a noncritical activity list (see Step 1 in Fig. 1). In the subsequent steps, in terms of the priority values of the free activities, each activity will be added into a location where the maximum reduction of RRH + MRD should be achieved. After a noncritical activity is added to the schedule, the start and finish time of other noncritical activities are thus updated. Following this process, all the noncritical activities would be added to the schedule.

Where to place the noncritical activities can be achieved by aligning a newly added noncritical activity to existing activities in the schedule. An example is shown in Fig. 5. The grey highlighted bar denotes the resource usage of the existing activities in the schedule. Two new activities with oblique lines are added to the schedule. To achieve the maximum reduction of RFC, the two activities should be aligned with the edge of existing activities. Thus, if the first activity may be located at P_1, the reduction of RRH is R_1 and if the second activity is added at the position P_2, the reduction is $R_2 + R_3$.

After added to the existing schedule, the two activities are merged to the existing schedule. The next noncritical activity is added based on the newly generated schedule where the maximum reduction of RFC could be achieved. This approach uses the greedy algorithm which makes the locally optimal choice at each stage with the hope of finding the global optimum. Therefore, each noncritical activity is added to the schedule one by one. The sequence of adding noncritical activities depends on the priority values of these noncritical activities. For all available noncritical activities, an activity with highest priority value will be chosen first to add the existing schedule to achieve the maximum reduction of RRH + MDR. However, this method cannot guarantee the final reduction of RRH + MDR attains maximum because the greedy algorithm only finds the locally optimum at each step without considering previous steps. This issue is discussed in the validation section. In the end, the total cost and project duration would be calculated in this module and returned to the EMO module.

3.4 The local search module

The calculated results by the EMO module are further optimized by searching for the possible positions of the noncritical activities to ensure the maximum reduction of RRH + MRD. For each individual, the activity priority value and mode are determined. Then, apply the project scheduling module to calculate the total float and free float based on the generated schedule by using backward pass. Then to further minimize RRH + MRD, El-Rays' method is applied to search the potential positions of the noncritical activities [23]. El-Rays used a GA technique to search the possible locations of noncritical activities. From the "right" to "left" in the schedule chart, the most right noncritical activity (i.e., latest finish time) is shifted first by the calculated value in Eq.(13) [23]. After shifting this activity, update the FF and TF of preceding activities. In this way, each noncritical activity is shifted. One difference from the original El-Rays' method is that the shifting process should keep the resource feasible.

$$S_{i*} = \left\lfloor \frac{FF_{i*}+1}{TF_i+1} \times M_i \right\rfloor$$

$$M_i = P / N$$

Eq. (9)

S_{i*} = Actual shift-day for the activity $i*$.
M_i= Possible shift-day of the activity $i*$.
FF_{i*} = Free Float of the activities $i*$ when resource constraints are considered.
TF_i = Total Float of the activity i calculated by CPM without consideration of resource constraints.
T_{max}= Specified maximum project duration.
$\lfloor\ \rfloor$ = Fraction truncation.
P = priority value of activity i
N = the total number of activity.

Each individual of the population calculated by the EMO module would use the local search algorithm. If 30 individuals are finally generated by the EMO module, thus, the GA algorithm would run 30 times. Since the local search cannot guarantee finding the global optimum location, the results may be not necessarily better than results calculated by the EMO module. Therefore, the results calculated by the local search module will be compared with the results

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

of the EMO module. The better results would be recorded as the final results.

3.5 Additional consideration

RRH and RID are agreeable though each of them may have different applications. A strategy was proposed to reduce the cost incurred by both of the release/rehiring cost and idleness cost of resources. Here RIC is defined to measure the cost of resource idleness and RIC = unit cost for idle resource * RID. RHC has the same definition as RFC. In Fig. 6, when the resource level is maintained at the level 2, the total cost = RIC* + RHC* (see (a) in Fig. 6). RIC* is equal to the area indicated by the oblique lines. RHC* is equal to the length of the two arrow lines. The change of resource level may lead to different RIC and RHC.

In (b) of Fig. 6, assume the resource level increased to the resource level 2 from the level 1, RHC is reduced while RIC is increased. To keep the total value of RHC and RIC minimum, the resource amount at the level 2 should be selected because the increasing of RIC below the level 2 is smaller than the decreasing of the RHC. This strategy may apply after the final calculation and allow users to select the resource level that reduce both of RHC and RIC.

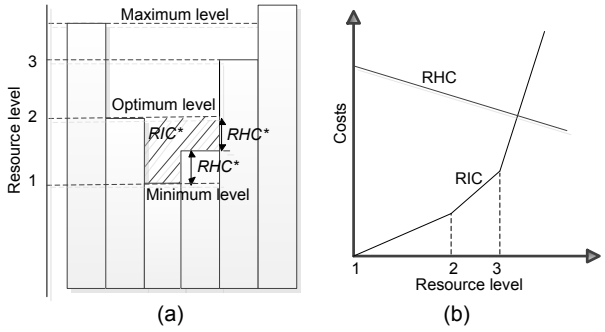


Fig. 5 Reduction of RHC and RIC

4 Model validation

The model validation includes the analyzing runtime, estimating the expected RFC error due to the proposed chromosome representation, and testing the effectiveness of the model through computational experiment.

4.1 Time complexity analysis

The time complexity is adopted to evaluate the runtime of the model in terms of the number of the activities. The local search module is independent of the number of activity and thus the time complexity analysis is not conducted for this module.

Zitzler and Laumanns [31] analyzed the time complexity of the SPEA2 algorithm. The runtime is mainly composed of two parts. (a) Fitness assignment. This runtime is dominated by the density estimator (O(M^2log$M$)), where M is the sum of the population and archive sizes. (b) Environment selection. The worst run-time complexity of the truncation operator is O(M^3), however, the average runtime is lower than O(M^2log$M$). For the project scheduling module, the time complexity is O($n^2 \bullet K$) where n is the number of activities and K is the number of resources [34].

The runtime of the project leveling module is mainly consumed by calculating TF, adding and deducting the free activities, and calculating the RRH. The runtime of calculating the total float is about O($\sum_{i=1}^{n} TF_i$) where TF_i is the total l float of the activity i. Since TF_i should be less than D denoting project duration). The maximum value of $\sum_{i=1}^{n} TF_i$ is nD. The runtime of the second part, adding and deducting the free activity is O($\sum_{i=1}^{n} (TF_i * d_i)$) where d_i is the duration of activity i. Each location between ES and FS of an activity should be checked to make sure the resource constraint is not violated and checking each location requires a runtime of O(d_i) to add or deduct the resources. The third part, calculation of RFC, takes O(D) where D denotes the total project duration. Together, the total runtime of the resource leveling module is O($\sum_{i=1}^{n} (TF_i \bullet d_i) + D$).

The total runtime for the three modules was thus calculated by summation of the runtime of each module. For each individual in the EMO module, the runtime is the sum of the project scheduling and resource leveling modules, i.e., O($\sum_{i=1}^{n} (d_i \bullet TF_i) + D + n^2 \bullet K$). There are a total of $G \bullet P$ individuals where G denotes the

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

generations and P denotes the population size for one generation. By adding the runtime of environmental selection, the total runtime of the worst case is $O(\{[\sum_{i=1}^{n}(d_i \bullet TF_i) + D + n^2 \bullet K] \bullet P + M^3\} \bullet G)$. Thus, the runtime shown here is acceptable because it is polynomial times of the number of activities.

The testing of the total runtime of the model was done in a laptop with configuration of Windows 7 OS, 2.0 GHz CPU, 2G RAM. The EMO module is coded by C language while the other modules are coded in MATLAB. The total runtime of the three modules is about 30s for a project with ten activities.

4.2 RFC error analysis

The arrangement of noncritical activities in the resource leveling module uses greedy algorithm as indicated in Section 3.3. It is known that the sequence of adding noncritical activities to the schedule affects the value of RFC. However, even by checking all possible combinations of these noncritical activity sequence (the maximum number is n!), the greedy algorithm cannot still guarantee finding the maximum reduction of RFC. All the possible combinations of activity sequence (i.e., the priority values in the chromosome) would represent the search space of the EMO module. The EMO module searches the space to find the maximum reduction of RFC (i.e., minimum of RFC) in the space. However, the EMO module cannot find the global optimum of RFC. Two reasons may cause this result. One reason is that EMO can only find the near-optimum solutions as other evolutionary algorithms. The second reason is that the global optimum is not located in the search space of the EMO module and therefore the EMO module cannot find the global optimum of RFC even if searching the whole space. This section would address two issues related to the second reason: (a) when the global optimum is located in the search space; (b) if the global optimum is not located in the search space, what the expected error is.

To allow the EMO module to find the globally optimum of RFC, the greedy algorithm must add the noncritical activities in terms of the activity sequence which represents one point in the search space of EMO, to the schedule. The key problem here is whether to find an activity sequence (i.e., priority values of activities) in which the noncritical activities can be added to the globally optimum locations of the noncritical activities by using the greedy algorithm. Only if the globally optimum locations of the noncritical activities can be found by the greedy algorithm can the global optimum of RFC also be found by the EMO module. The globally optimum locations of the noncritical activities can be found by the greedy algorithm if satisfying one of the following two conditions: (1) no more than two noncritical activities are adjacent to each other, or at least one most outside edge of the adjacent noncritical activities is not adjacent to any of the critical activities (see (a) and (b) in Fig. 7); or (2) if the globally optimum location of noncritical activities violates the condition (1), the global optimum is still located within the search space if there exists an activity sequence in which the noncritical activities could be added to the same locations as those optimum locations of noncritical activities are located (see (c) in Fig. 7)

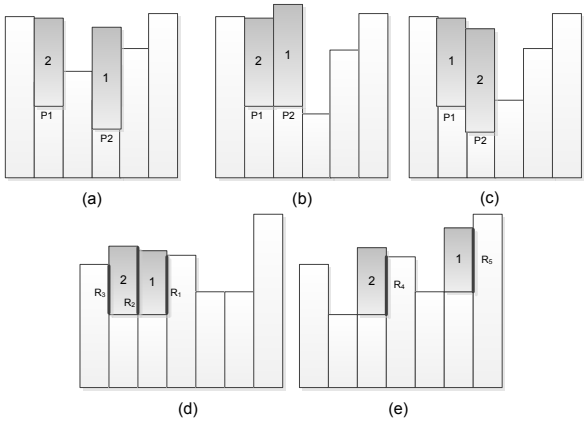


Fig. 6 Illustration of conditions (1) and (2)

The conditions (1) and (2) are illustrated in Fig. 7. The grey activities are noncritical and the activity 1 and the activity 2 are added to the schedule in order. According to the greedy algorithm, activity 1 should be added at position P_2 and activity 2 at position P_1 (see (a) and (b) in Fig. 7). This sequence of adding the two noncritical activities could lead to the maximum reduction of RFC. For (b) in Fig. 7, activities 1 and 2 are adjacent to each other but the right edge of the activity 2 is not adjacent to the critical activity. Thus, both (a) and (b) meet the condition (1). In (c) Fig. 7, the condition (1) is

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

violated. However by adding activity 1 and activity 2 in order, the global optimum reduction of RFC can still be achieved. Thus, (c) also meet the requirement of the condition (2). The arrangement of noncritical activities in (d) of Fig. 7 shows the globally optimum locations of activities 1 and 2. The total reduction is $R_1+R_2+R_3$ (see (d)). However, if the greedy algorithm is applied, the locally optimum locations of the two activities are shown in (e) of Fig. 7. The reduction is R_4+R_5, which is less than the value of $R_1+R_2+R_3$.

Condition (1) is briefly proved here. Assume the globally optimum locations of all noncritical activities are known, and then a noncritical activity with the smallest reduction of RFC can be first removed from the schedule while the optimum locations of the other noncritical activities are kept unchanged in the schedule chart, that is to say, these noncritical activities have larger RFC reduction in their original positions. All the noncritical activities can be removed from schedule in this way until only the critical activities are remained. Then if all the noncritical activities are added into the schedule (with only critical activities) in the reverse order by using greedy algorithm, the final locations of the noncritical activities would achieve the same reduction as the previous globally optimum locations of the noncritical activities.

When condition (1) is violated, the expected error is calculated to estimate the difference between the locally optimum locations found by EMO and the globally optimum locations. This can be shown in (c) and (d) of Fig. 7. If there are two activities are adjacent, the reduction is R_4+R_5 if the greedy algorithm is applied while the global optimum reduction should be $R_1+R_2+R_3$. The difference (or, error) is about 1/3 of the globally maximum reduction if the values of R_1, R_2, R_3. R_4 and R_5 are assumed to be close. Similarly, if there are three activities adjacent (i.e., violating condition (1)), the error is ¼ of the globally maximum reduction of RFC. If there are i activities adjacent, the error is $1/(i+1)$. Assume the average reduction of RFC for each noncritical activity is t, and each noncritical activity has an equal possibility of meeting or violating the condition (1). In other words, from the perspective of the expectation, there are a_1 noncritical activities that are not adjacent to noncritical activities and critical activities in the schedule chart (i.e, just like (a) in Fig. 7, meeting condition (1)), the error is 0%; also there are a_1 free activities that are only adjacent to one noncritical activity and critical activities (i.e., violating condition (1)) and the error is about $(a_1/2)$ x (1/3) because two noncritical activities are adjacent to form only one whole block (as seen from (e) of Fig. 7); similarly, there are a_1 free activities that are only adjacent to i noncritical activities and critical activities and the error is (a_i/i) x $[1/(i+1)]$. The total noncritical activities would be a_1 x i. Since the assumption is made that each noncritical activity has an equal possibility of adjacent to different number of other noncritical activities, the expected error can be calculated by Eq. (14).

$$e=\left[a_1\times 0\%+\frac{a_1}{2}\times\frac{1}{3}+\frac{a_1}{3}\times\frac{1}{4}+\ldots+\frac{a_1}{i}\times\frac{1}{i+1}\right]/(a_1\times i)=\frac{1}{i+1}-\frac{1}{2i}$$

Eq. (10)

If i is equal to 2 or 3, e would be about 8%, and if i becomes larger, e would become smaller. In addition, if the second condition (2) is considered, the value of e should be even smaller. Thus, it is concluded that the greed algorithm adopted in this research is reasonably accurate to ensure the locally optimum value of RFC calculated by the EMO module is close to the globally optimum reduction if it is assumes the EMO module has the ability to find the global optimum in the search space.

4.3 Computational experiment

The developed model except the EMO module was coded in MATLAB Version 7.9 in a laptop with a 2.0 GHz CPU, 2G RAM under Windows 7 OS. The EMO module was coded by C language to improve the computation efficiency. The external interface feature of MATLAB enables users to call C functions as MATLAB built-in functions. The dataset of the test cases came from PSPLIB developed by Kolisch and Sprecher [35]. Ten cases from the dataset "j10.mm" (note: this dataset was created originally for the multi-mode RCPSP) were selected. The case number selected are 1, 50, 100, 150,…. 450. The model developed in this research introduces some new capacities and is not used to solve the multi-mode RCPSP. Thus, a baseline model is proposed for comparison with the developed model.

4.3.1 Baseline model

The baseline model has two phases each of which would use the GA algorithm. In the first

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

phase, the minimum project duration was generated. In the second phase, the minimum total cost was selected based on the schedule generated in the first phase. The local search module was used to search the minimum RFC based on the calculated individuals in the first phase.

The objective of the first phase was to minimize the project duration only. This could be regarded as a standard multi-mode RCPSP. A number of similar researches were done in this topic [21, 33, 36]. The chromosome representation is the same as the tested model (i.e., the model developed by this research) mentioned in Section 3.1.1 and 3.1.2. In addition, an external archive was developed similar just like SPEA2. This archive was used to record the individuals that generated small project durations. In other word, during the GA running, the individuals that could generate the smaller project durations were store in the archive. In the end, the archive contained the individuals that could generate smaller project durations than other individuals. The capacity of archive was set to 30 initially. The individuals in the archive were ranked in order, which means the 1st individual generated the smallest duration while the 30th individual generated the largest duration. If additional individuals that were not included in the archive could generate the same duration as the 30th individual, the capacity of the archive would be increased to incorporate these additional individuals until 60. However, if there are still other individuals, a strategy similar with the SPEA II would be conducted to select those individuals that could generate the even distribution along the Pareto front.

If all the individuals that could generated the minimum schedule *T* were included in the archive in the first phase, and the GA algorithm could find the minimum cost *C* based on the schedule in the second phase, then some points (T, C) should be located in the Pareto front line. The points (T,C) would be compared with solutions calculated by the tested model. Four cases may occur after the comparison (see Fig. 8): "dominated", "nondominated", "dominate" and "equal". The black solid point denotes (T, C) that is calculated by the baseline model. The white points denoted the solution calculated by the tested model. P_1 dominates the point (T,C) and P_2 is dominated by (T, C). P_0 may dominate many points P_i, thus, the average point is calculated and then the percentage difference *E* is calculated by Eq. (15). In Fig. 8, the grey point denotes the average position of the two points. In the case of existing many points like P_0, the average *E* would be calculated.

$$E = \frac{\sum P_i / n - P_0}{P_0} \qquad \text{Eq. (11)}$$

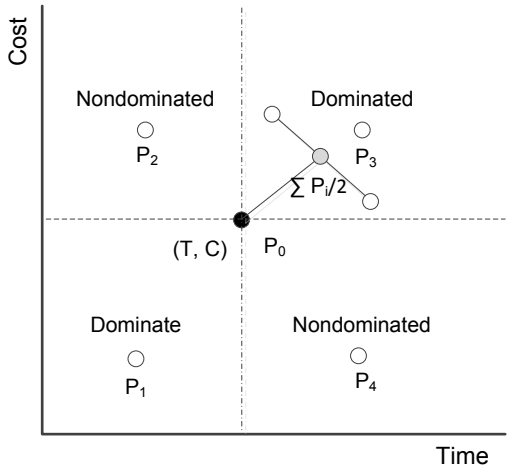


Fig. 7 Comparison of (T, C) and calculate solutions.

4.3.2 Experimental results

Two groups of resource unit cost were tested. The absolute number of unit cost is not important. The ratio between the two unit costs does matter. Thus, the resource unit cost is set to 10 all the time and the unit hiring/releasing cost (UHC) is set to 10 and 20, respectively. Preliminary tests were conducted to determine the simulation parameters. The test results showed that the mutation rate was set to be 0.05, which was the same as that recommended by Hartmann [14], the population and archive sizes were set to 30. The testing cases which had errors are shown in Table 1. Only cases #1 and #2 have errors calculated by Eq (15). If using the local search method, the errors were decreased significantly. Further analysis showed that the reason why most of the errors were zero was that for all the ten cases, more than 80% of the points generated by the tested model dominated the points generated by the baseline model. As shown in Fig. 8, most of the calculated points are located in the lower left corner of the coordinator system. This testing also showed that the most of points calculated by the baseline model only belonged to weak Pareto optimal. The reason is either that some individuals that could lead to the global optimum were missing in the archive in the first

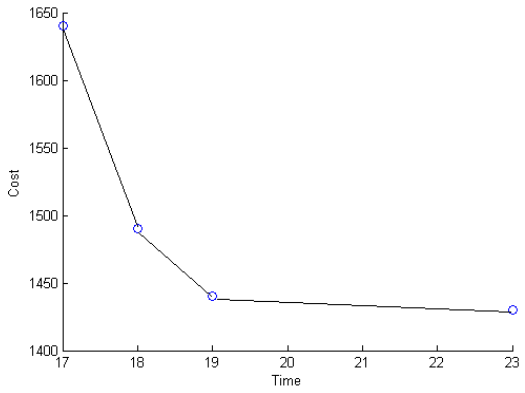
PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

phase of the baseline model or the GA only found the near-optimum solutions. The testing also showed that the RFC costs took up about 15% and 30% of the total cost for each group, respectively. Though the experiment could not justify the points generated by the tested model were located in the Pareto front line, the test still showed that the tested model had the potential of generating relative good results, and at least had much accurate results calculated by the baseline model.

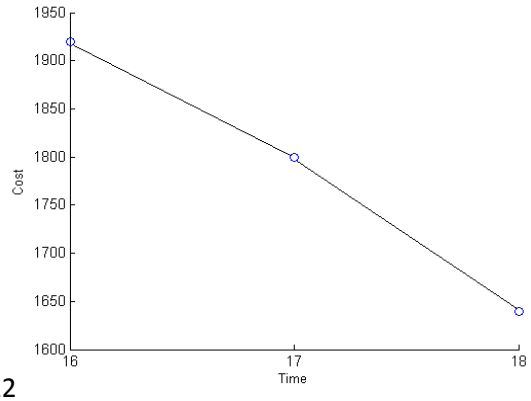
Table 1 Errors for cases #1 and #2

Cases having errors	Group 1 (URC = 10)				Group 1 (URC = 20)	
	No local search		Local search		No local search	Local search
	Time	Cost	Time	Cost	Time	Cost
#1	0.0 %	0.0 %	0.0 %	1.1 %	0.0%	4.1%
#2	0.0 %	0.6 %	0.0 %	0.0 %	0.0%	1.1%

The time and cost of case #2 is shown in Fig. 9. The figure (a) indicates a minimum duration of 17 days while the figure (b) shows a minimum duration of 16 days. When UHC = 10, the resource usage cost had a larger effect on the total cost and thus the increase of duration would lead to the greater decrease of the total cost. The results indicated that the RFC took up about 20% of the total cost when UHC was set to 10 and 25% when UHC was set 20. The minimum duration was the same as that calculated by the baseline model, which can be seen from Table 1. When the duration was increased by about 12% from the minimum duration, the total cost was decreased by 12% (UHC = 10) and 15% (UHC = 20), respectively.



(a) UHC =10



(b) UHC =20

Fig. 8 Time and cost for case #2

5 Conclusion

This research proposed a time-cost tradeoff model based on the SPEAII to optimize the project duration and resource cost by integrated consideration of the resource-constrained project scheduling and resource leveling. Resource rehiring/releasing cost was proposed to measure the resource fluctuation. The impact of resource fluctuation was thus converted to costs which could be added into the resource cost as an objective function.

The analysis of the time complexity of the model showed that the runtime was acceptable and had polynomial relationship with the number of activity. It took 30s to calculate the result for a ten-activity project. In additional, the model was evaluated in terms of the potential errors of resource fluctuation cost to justify the effectiveness of the chromosome representation. Ten testing cases were selected to evaluate the accuracy of the model. The test showed that in two out of the ten cases the model calculated some points that were dominated by the points calculated by baseline model. The errors were around 1%. In most cases, the model had a much better performance than the proposed baseline model and more than 80% of the solutions calculated by the model dominated those calculated by the baseline model. The performance of the model in the computational experiment indicated that the model was reasonably accurate.

However, due to the limitations of the baseline model, the developed model was not adequately validated. In addition, this model assumes the cost information such as unit cost of resource rehiring and releasing can be predetermined. In reality, it is sometime difficult to estimate all the cost information. Thus, the users have to run many times to compare the results if this cost

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

information is uncertainty. Future work can be done by introducing the stochastic cost model to develop a more realistic model.

References

[1] Chen Y, Hu J, Mo P. The Development of The Lifecycle Function Model By IDEF0 For Construction Projects., WiCOM 4th International Conference on Wireless Communications, Networking and Mobile Computing: IEEE; 2008. p. 1-4.

[2] Hu J, Flood I. A multi-objective scheduling model for solving the resource-constrained project scheduling and resource leveling problems., Interntional Conference on Computing in Civil Engineering, Clearwater Beach, Florida, United States; 2012. p. 49-56.

[3] Hu J, Patel M. Optimized Selection and Placement of Sensors using Building Information Models (BIM)., IES Annual Conference, Pittsburgh, PA; 2014.

[4] Hu J, Olbina S. Simulation-Based Model for Integrated Daylighting System Design. J COMPUT CIVIL ENG 2013:A4014003.

[5] Zdamar L, Ulusoy G. A survey on the resource-constrained project scheduling problem. IIE TRANS 1995;27(5):574-86.

[6] Herroelen W, Leus R. Project scheduling under uncertainty: Survey and research potentials. EUR J OPER RES 2005;165(2):289-306.

[7] Burns SA, Liu L, Feng CW. The LP/IP hybrid method for construction time-cost trade-off analysis. Construction Management and Economics 1996;14(3):265-76.

[8] Deineko VG, Woeginger GJ. Hardness of approximation of the discrete time-cost tradeoff problem. OPER RES LETT 2001;29(5):207-10.

[9] Hu J, Olbina S. Illuminance-based slat angle selection model for automated control of split blinds. BUILD ENVIRON 2011;46(3):786-96.

[10] Olbina S, Hu J. Daylighting and thermal performance of automated split-controlled blinds. BUILD ENVIRON 2012;56(0):127.

[11] Hu J, Olbina S. Radiance-based Model for Optimal Selection of Window Systems., ASCE 2012 International Conference on Computing in Civil Engineering, Clearwater Beach, FL, USA; 2012.

[12] Chen P, Shahandashti SM. Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints. AUTOMAT CONSTR 2009;18(4):434-43.

[13] Guo R, Guo W, Wang W. Optimization of multi-project in limited resources on the basis of improved genetic algorithm., 2008 Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Wuhan, China: Inst. of Elec. and Elec. Eng. Computer Society; 2008. p. 949-52.

[14] Hartmann S. A competitive genetic algorithm for resource-constrained project scheduling. NAV RES LOG 1998;45(7):733-50.

[15] Kolisch R. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. EUR J OPER RES 1996;90(2):320-33.

[16] Tsai DM, Chiu HN. Two heuristics for scheduling multiple projects with resource constraints. Construction Management & Economics 1996;14(4):325-40.

[17] Feng C, Liu L, Burns SA. Using Genetic Algorithms to Solve Construction Time-Cost Trade-Off Problems. J COMPUT CIVIL ENG 1997;11(3):184-9.

[18] Leu S, Chen A, Yang C. A GA-based fuzzy optimal model for construction time-cost trade-off. International Journal of Project Management 2001;19(1):47-58.

[19] Chen YQ, Zhang SJ, Liu LS, Hu J. Risk perception and propensity in bid/no-bid decision-making of construction projects. Engineering, Construction and Architectural Management 2015;22(1):2-20.

[20] Chen P, Weng H. A two-phase GA model for resource-constrained project scheduling. AUTOMAT CONSTR 2009;32(1):9-17.

[21] Wuliang P, Chengen W. A multi-mode resource-constrained discrete time – cost tradeoff problem and its genetic algorithm based solution. International Journal of Project Management 2009;201(2):409-18.

[22] Castro Lacouture D, Suer GA, Gonzalez Joaqui J, Yates JK. Construction project scheduling with time, cost, and material restrictions using fuzzy mathematical models and critical path method. J CONSTR ENG M 2009;135(10):1096-104.

[23] El-Rayes K, Jun DH. Optimizing Resource Leveling in Construction Projects. J CONSTR ENG M 2009;135(11):1172-80.

[24] Doulabi SHH, Seifi A, Shariat SY. Efficient Hybrid Genetic Algorithm for Resource

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015

Leveling via Activity Splitting. J CONSTR ENG M 2011;137(2):137-46.
[25] Chan WT, Chua D, Kannan G. Construction resource scheduling with genetic algorithms. J CONSTR ENG M 1996;122:125-32.
[26] Senouci AB, Eldin NN. Use of genetic algorithms in resource scheduling of construction projects. J CONSTR ENG M 2004;130:869-77.
[27] Hegazy T. Optimization of resource allocation and leveling using genetic algorithms. J CONSTR ENG M 1999;125:167.
[28] Liao TW, Egbelu PJ, Sarker BR, Leu SS. Metaheuristics for project and construction management - A state-of-the-art review. AUTOMAT CONSTR 2011;20(5):491-505.
[29] Leu SS, Yang CH. GA-based multicriteria optimal model for construction scheduling. J CONSTR ENG M 1999;125:420-7.
[30] Lucko G. Integrating Efficient Resource Optimization and Linear Schedule Analysis with Singularity Functions. J CONSTR ENG M 2010;1(1):164.
[31] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the strength pareto evolutionary algorithm (Tech. Rep. No. 103)., Eurogen: Gloriastrasse; 2001.
[32] Coello C, Lamont GB, Van Veldhuizen DA. Evolutionary algorithms for solving multi-objective problems. New York, US:Springer-Verlag New York Inc; 2007.
[33] Hartmann S. Project scheduling with multiple modes: A genetic algorithm. ANN OPER RES 2001;102(1):111-35.
[34] Pinson E, Prins C, Rullier F. Using tabu search for solving the resource-constrained project scheduling problem. EURO-WG PMS 1994;4:102-6.
[35] Kolisch R, Sprecher A. PSPLIB - A project scheduling problem library : OR Software - ORSEP Operations Research Software Exchange Program. EUR J OPER RES 1997;96(1):205-16.
[36] Gonçalves JF, Mendes J, Resende M. A genetic algorithm for the resource constrained multi-project scheduling problem. EUR J OPER RES 2008;189(3):1171-90.

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1299v1 | CC-BY 4.0 Open Access | rec: 13 Aug 2015, publ: 13 Aug 2015