A Honey Bee Swarm Optimization Algorithm for Minimizing the Total Costs of Resources in MRCPPSP

Mohammad Khalilzadeh*

Department of Industrial Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran; mo.kzadeh@gmail.com

Abstract

In this paper, we solve an extended type of multimode project scheduling problem considering renewable and non-renewable resource constraints and finish-to-start precedence relations among project activities. We suppose that renewable resources are hired and are not accessible in the whole duration of the project. Therefore, there is an assigned ready date as well as a due date for each renewable resource type so that no resource can be rented before its ready date. But, the resources are allowed to be used after their due dates by incurring penalty costs. The aim is to minimize the whole charges of both renewable and nonrenewable resource usage. This problem is introduced by Khalilzadeh et al.7 as multimode resource-constrained project scheduling problem minimizing the net present value of entire weighted resource tardiness penalty costs (MRCPPSP-DCTWRTPC) in which both renewable and nonrenewable resource restrictions rely on activity mode. The problem we proposed in this paper is the extended form of the problem introduced by Khalilzadeh et al.7 with considering the discounted cash flows of the resource costs. We call this problem multi-mode resource-constrained project scheduling problem, minimization of discounted cash flows of total weighted resource tardiness penalty cost (MRCPPSP-DCTWRTPC). For this problem, we propose a novel meta-heuristic algorithm based on a Honey Bee Swarm Optimization (HBSO) approach together with a prioritization rule for project tasks and numerous improvement and local search procedures. Computational results prove the effectiveness and efficiency of the presented method.

Keywords: Discounted Cash Flow, Honey Bee Swarm Optimization, Multi Modes, Project Scheduling, Resource Cost, Tardiness Penalty Cost

1. Introduction

The Resource-Constrained Project Scheduling Problem (RCPSP) is the arranging of project activities considering precedence constraints as well as renewable resource restrictions with the goal of minimization of the project duration. All activities are non-preemptive and can be accomplished in a single mode. Demeulemeester and Herroelen4 categorized well-known types of RCPSP problems and their different solution methods. In the multi-mode RCPSP known as MRCPPSP, a set of permissible modes can be expressed for each activity which is described by a fixed duration and related resource obligations. In this paper we consider MRCPPSP with the objective of minimizing discounted cash flows of total costs of all resources. Two kinds of resources, renewable and nonrenewable, are assumed. Nonrenewable resource charge of a task is a function of its resource requisites, relying on its modes. The restricted renewable resources are hired and each renewable resource is obtainable in a pre-defined successive time periods indicated by its ready time and due date and is not accessible before the ready time. Nevertheless, each renewable resource can be used after its due date with tardiness penalty cost. As the related renting fee of each renewable resource is constant, it is not required to add it in the objective function and only tardiness penalty cost is measured for every renewable resource. The MRCPPSP with minimization of total costs of resources (RCPSP-TWRTPC) is a real-world problem and an extended type of the MRCPPSP where all assumptions and constraints of the MRCPPSP are thought but the objective function is changed. We consider there are

*Author for correspondence
limited renewable resources like skillful human resources with high expertise levels, specific types of cranes, tunnel boring equipment and so on that should be hired from other firms providing these kinds of resources. As these scarce renewable resources are used in other projects, there is a required ready date besides a due date for each of them such that no resource can be leased before its ready date but these resources are permitted to be exploited after their due dates by charging penalty cost, based on the type of the resource. Also, we assume that there are limited nonrenewable resources such as budget, materials, energy, or other resources which are consumed through the project duration.

Ranjar et al. considered this problem with single mode for each project task and accessibility of one unit only for each kind of renewable resource. They called this problem Resource-Constrained Project Scheduling Problem, minimization of Total Weighted Resource Tardiness Penalty Cost (RCPSP-TWRTPC), which is an extended type of Resource-Constrained Project Scheduling Problem (RCPSP). They introduced a branch-and-bound method to tackle the problem.

Khalilzadeh et al. introduced the generalized form of the problem studied by Ranjar et al. with more realistic viewpoint of resource costs by considering both renewable and nonrenewable resources cost. They call the problem Multimode Resource-Constrained Project Scheduling Problem, minimization of Total Weighted Resource Tardiness Penalty Cost (MRCPSP-TWRTPC). The problem we proposed here is the extended form of the problem introduced by Khalilzadeh et al. with considering the discounted cash flows of the resource costs. We call this problem multi-mode resource-constrained project scheduling problem, minimization of discounted cash flows of total weighted resource tardiness penalty cost (MRCPSP-DCTWRTPC).

MRCPSP-DCTWRTPC is a comprehensive form of the RCPSP. Considering the NP-hardness of RCPSP, the MRCPSP-TWRTPC problem is NP-hard too. Therefore, as the most efficient and widely-used approach, a meta-heuristic algorithm can be applied to resolve this problem. In this paper, we present a meta-heuristic based HBSO algorithm for resolving this problem.

The remaining of this paper is structured as follows. In the next section, MRCPSP-DCTWRTPC is defined in details and is expressed in a mathematical model. In section 3, a definition of the HBSO method is proposed and subsequently the meta-heuristic HBSO algorithm is described in the next section. The Computational experiments are presented in section 5. Finally, section 6 concludes the research paper.

2. Problem Description

In MRCPSP-DCTWRTPC, we schedule the project activities in order to minimize its entire costs. Project resources required for implementing project activities can be classified as either renewable or nonrenewable. Activity $j$ may have some execution modes defined by $M_j$. Each activity mode indicates the activity duration and the activity requisites for the definite amount of renewable and nonrenewable resources. Each form of restricted renewable resource is leased for a constant time interval, starting from its ready time and finishing with its due date, and is not accessible before its ready time but may be used after its due date with tardiness penalty cost. Nonrenewable resources are not restricted. All tasks are ready at the starting of the project, and no preemption is allowed. If an activity is started under a specific mode, the activity mode cannot be changed. Activity $j$ executed in mode $m$ has duration $d_{jm}$ and requires $r_{jmk}$ units of renewable resource $k$ and $n_{jmk}$ units of nonrenewable resource $k$. The project network is shown by an Activity On Node (AON) illustration with finish-to-start precedence relations and zero time lag. Dummy activities 1 and $n$ relate to start and finish time of the project. The list of activities is topologically numbered, i.e., every single predecessor of each activity has a smaller number than the number of the activity. Also, we describe the earliest and latest start time of activity $j$ by $EST_j$ and $LST_j$ respectively. $EST_j$ and $LST_j$ are determined by CPM forward and backward pass methods applying the mode with shortest duration for each activity and allocating $LST_n = LFT_n = T$ where $T$ is the deadline for project duration calculated by any usable method, like the sum of the longest time of entire project activities plus the ready times of renewable resources. Subsequently, each activity $j$ can only be accomplished in the time window $[EST_j, LST_j]$.

We use the problem parameters as described by Khalilzadeh et al. incorporating the discounted cash flows of the resource costs:

$n$: number of project activities,
$NR$: number of nonrenewable resources,
$c_{jm}$: unit cost of nonrenewable resource $k$,
$R$: number of renewable resources,
$R_j$: renewable resource $k$ availability,
The resource k requirement for executing activity j under mode m, 

\( r_{jm} \) : renewable resource k requirement for executing activity j under mode m,

\( n_{jm} \) : nonrenewable resource k requirement for executing activity j under mode m,

\( E_{STj} \) : earliest start time of activity j,

\( L_{STj} \) : latest start time of activity j,

\( T \) : upper bound of the project makespan,

\( \alpha \) : daily discounted rate.

We also define the decision variables as the following:

\[
x_{jmt} = \begin{cases} 1 & \text{if activity j is started under mode m in period } \tau \\ 0 & \text{otherwise} \end{cases}
\]

\[
y_{k\tau} = \begin{cases} 1 & \text{if renewable resource k is used in period } \tau \\ 0 & \text{otherwise} \end{cases}
\]

\( l_k \) : renewable resource k tardiness, determined by:

\[ l_k = \max(0, CP_k - d_k) \]

where \( CP_k \) is the release time of resource k by the project.

The mixed integer programming model for this problem can be formulated as follows:

\[
\min \sum_{k=1}^{NR} C_k e^{-a \tau} \left( \sum_{j=1}^{M_j} \sum_{m=1}^{L_{ST_j}} n_{jm} x_{jmt} \right) + \sum_{k=1}^{R} P_{k \tau} l_k \tag{1}
\]

Subject to:

\[
\sum_{m=1}^{M_j} \sum_{\tau=E_{STj}}^{L_{STj}} x_{jmt} = 1 \quad j = 1, 2, \ldots, n \tag{2}
\]

\[
\sum_{m=1}^{M_j} \sum_{\tau=E_{STj}}^{L_{STj}} (\tau + d_{im}) x_{jmt} \leq \sum_{m=1}^{M_j} \sum_{\tau=E_{STj}}^{L_{STj}} \tau x_{jmt} \quad j = 1, 2, \ldots, n, \quad i \in P_j \tag{3}
\]

\[
\sum_{j=1}^{M_j} \sum_{m=1}^{L_{STj}} \sum_{\tau=E_{STj}}^{L_{STj}} x_{jmt} z_{\tau-d_{im}} \leq R_k \cdot y_{k\tau} \quad k = 1, 2, \ldots, R, \quad \tau = 1, 2, \ldots, T \tag{4}
\]

\[
\sum_{\tau=1}^{T} y_{k\tau} = 0 \quad k = 1, 2, \ldots, R \tag{5}
\]

In the above model, objective function (1) is project cost minimization where the first and second terms are total charges of nonrenewable resources and total penalty fees of renewable resources tardiness respectively. Constraint set (2) guarantees that each activity \( j \) is started with one of its modes within its identified start time periods, i.e. \([E_{STj}, L_{STj}]\). Constraint set (3) shows precedence relationship between activities. Constrains (4) bound renewable resource usage. According to constraints (5), renewable resources are not allowed to be used before their ready times and their tardiness periods are defined by constraints (6). Finally, constraint sets (7), (8) and (9) are non-functional ones.

### 3. Honey Bee Swarm Optimization

The HBSO algorithm is relatively recent, evolutionary, and population-based meta-heuristic procedure. HBSO inspired from the social behavior of natural swarms exploits a swarm of honey bees for search space that are updated from iteration to iteration. The model of quest selection that results in the emergence of group intelligence of honey bee swarms contains three essential components: food sources, employed foragers and unemployed foragers and the model describes two principal modes of the behavior: using a nectar source and leaving a source. A meta-heuristic algorithm should be able to explore search space effectively and efficiently.

### 4. HBSO for MRCSP-DCTWRTPC

In this part, we introduce a HBSO algorithm to tackle MRCSP-DCTWRTPC. The pseudo-code of this algorithm is displayed in Figure 1.

#### 4.1 Preprocessing

Sprecher et al. presented numerous preprocessing regulations in order to decrease feasible space of MRCSP. Subsequently, these rules have been applied to the other...
papers such as Lova et al.,10 Peteghem and Vanhoucke11, and Hartman5. Bearing in mind the likenesses between MRCPSP-DCTWRTPC and MRCPSP, we use two of these regulations in our recommended problem. The first one is the non-executable mode elimination rule for each activity. For a non-executable mode, the quantity of the resource required for completing the activity is more than the resource disposal. Another method is inefficient mode elimination technique. An assumed mode is inefficient for an activity if there is another mode for which the length of that activity is less, and that activity can be completed with less total quantity of both renewable and nonrenewable resources. Thus, the modes of activities are evaluated one after the other and non-executable and inefficient modes are removed.

4.2 Generating Preliminary Food Source

Preliminary food source is created randomly. Each element $j, j = 1,\ldots,n$, for food source is generated randomly from the ranges $[1, M]$. As there is no non-renewable resource constraint in the problem and all non-executable modes have been removed, preliminary mode assignments are feasible and no adjustment is required.

4.3 Activity Priority for Scheduling

In order to produce an answer in MRCPSP-DCTWRTPC, like MRCPSP-TWRTPC, two matters are to be determined: assigning modes to activities and scheduling them. By assigning modes to all project activities for a solution, the fees of nonrenewable resources are found and fixed. Then, scheduling of activities is conducted with the objective of minimizing the entire penalty costs of renewable resource tardiness. This procedure is explained in Khalilzadeh et al.7 as the following.

First, we describe the set of project activities which requires renewable resource $k$ for execution as the Activity Set of Resource $k$ (ASR$_k$). Each activity in this set may have immediate or non-immediate predecessors that may not be a member of this set. We outline the set of these predecessors which are not members of ASR$_k$ as Activities Predecessors of Resource $k$ (APR$_k$). Then, the pairs of ASR$_k$ and APR$_k$, $k = 1,\ldots, R$, are prioritized by index $k$ using the heuristic that activities in ASR$_k$ and APR$_k$ for the resource which has more possible of triggering tardiness penalty should have greater priority of being scheduled. To access the possible of the $k$-th resource tardiness penalty cost, we remind that this

---

**Figure 1.** Pseudo-code of HBSO algorithm for MRCPSP-DCTWRTPC.
penalty cost is equal to \( P \times \max\{0, CP_k - d\} \), where \( CP_k \) is the release time of resource \( k \) in the project, therefore \( P \times (CP_k - d) \) is a worthy measure for prioritizing the resources. Subsequently, we implement the following procedure to access \( CP_k \), \( k = 1, ..., R \), without knowing the schedule.

As no project activity can start sooner than the ready times of all the required resources, we add one dummy node \( k \) for each resource \( k \), \( k = 1, ..., R \), to the project network. Each dummy activity \( R \), with the duration of \( r_k \), is single mode with no resource constraint. All these dummy activities can be scheduled with the beginning of the project at time 0. Then for any activity \( j \) which requires renewable resource \( k \), we set dummy activity \( k \) as one of its predecessors. So accomplishing activity \( j \) is not possible before the time \( r_k \). Remind that dummy activity \( k \) is a member of \( APR_k \). The length of the critical path of sub-network \( k \) is represented by \( CP_k \) and is considered as a relative amount of \( CP_k \) for \( k = 1, ..., R \). \( CP_k \) is calculated by CPM method. After calculation of \( CP_k \) for all resources, the members of activity sets \( ASR_k \) and \( APR_k \), \( k = 1, ..., R \), are prioritizing with the value of \( P \times (CP_k - d) \).

According to this method, we give more priority to the activities in \( APR_k \) than activities in \( ASR_k \), as activities in \( ASR_k \) cannot be scheduled unless activities in \( APR_k \) are scheduled.

Lastly, for each resource \( k \) it is essential to prioritize the activities which are the members of \( ASR_k \) and \( APR_k \). We keep in mind while a set of activities requiring a renewable resource are to be scheduled, we actually deal with a RCPSP, as activities with identified modes are to be scheduled in order to complete within the shortest possible duration. Hence, in order to prioritize activities of a set of \( ASR_k \) or \( APR_k \), we use one of the most effective heuristics for scheduling activities in RCPSP. Lova et al. (2006) compared a number of most capable heuristics for prioritizing activities in RCPSP and revealed that prioritization of activities based on non-decreasing order of the sum of the Latest Start and Finish Time (LSTLFT) achieved the best results among single-pass heuristics. This heuristic has been amongst the best multi-pass methods as well. Multi-pass methods need much more computational times than single-pass methods, but they generally result in minor solution improvement, therefore, we apply the LSTLFT technique with single pass method for prioritizing activities of \( ASR_k \) and \( APR_k \).

### 4.4 Scheduling Activities

We apply parallel schedule generation scheme to this problem, since it works well with delay local search. Demeulemeester and Herroelen\(^3\) comprehensively described parallel schedule generation scheme.

Parallel schedule generation scheme repeats over the separate decision points at which activities can be added to the schedule. A decision point in time horizon corresponds with the completion times of already scheduled activities; thus, at most \( n \) decision points need to be considered. At each decision point, the unscheduled activities whose predecessors have been accomplished are taken into consideration in the order of priority list and are scheduled if no resource conflict exists at that time instant.

In this problem, all types of renewable resources have ready times and accessibility of them varies before and after these times and some new project activities may be eligible to be scheduled after these times. So, we consider ready times as well as the finish times of activities for selecting new point in project time horizon. A new point in project time horizon is the closest point to the current point amongst the ready times of renewable resources and the finish time of already scheduled activities.

### 4.5 Delay Local Search

This local search technique was executed by Chen et al.\(^3\) for the RCPSP problem to avoid local optimum. This method implements like the mutation operator in Genetic algorithm and may delay scheduling each activity despite its priority to allow other activities to be scheduled sooner and some resources requested by selected activity will be preserved for other activities.

In order to exploit resources more efficiently, scheduling some project activities are delayed despite their priority. Therefore, other project activities can be scheduled sooner. If these nominated activities are not delayed, could delay other activities for a rather lengthy time. So, each project activity is delayed, if \( q \leq q_0 \), where \( q \) is randomly chosen from uniform distribution \([0,1]\) and \( q_0 (0 < q_0 < 1) \) is the likelihood of delay and called “delay rate”\(^3\).

Since the efficiency of this delay local search in shortening project duration is proven by Chen et al.3, we use this technique for scheduling activities.

### 4.6 Mode Assignment Modification - Part I

In this paper, we apply mode assignment modification-part I introduced by Khalilzadeh et al.\(^7\) as a local search...
procedure to our algorithm. The existing schedule might have a set of activities with positive free slacks. Khalilzadeh et al.\textsuperscript{7} named this set FFA. For each j \in FFA it might be conceivable to amend its mode and reschedule it within its free float, so no other subsequent activities are delayed. Changing activity mode and rescheduling it may decrease nonrenewable resource fees and change CP\_k, release time of resource k, k = 1,\ldots, R. The change of CP\_k may modify renewable resource k tardiness, k = max \{0, CP\_k – d\_j\}, and its charges, p\_k l\_k. If we set the availability of resource k, k = 1,\ldots, R, for the periods after max \{CP\_k, d\_j\} equivalent to 0 and then reschedule activity j we are certain that this schedule will not rise renewable resource tardiness penalty costs.

4.7 Local Left Shift Improvement

Mode assignment modification - Part I may decrease the renewable resource requisites of the project for definite time periods in the current schedule. This is rational since the mode with less nonrenewable resources has usually longer duration and needs less renewable resources per time period. Therefore, the chance of local left shift for definite activities exists. Therefore, we implement the standard local left shift procedure after the mode assignment modification - Part I.

Having local left shift been performed, we may be able to change the modes of some activities by mode assignment modification-part I. So, these two procedures are executed one after another until no improvement can be obtained in the schedule.

4.8 Mode Assignment Modification - Part II

Khalilzadeh et al.\textsuperscript{7} consider the set of project activities with no successors and name it NSA. The direct predecessor activities of dummy activity n create NSA. If we change the mode of activity j which is a member of NSA and reschedule it, the schedule of no other activity changes and the project cost difference is \( \Delta c = \Delta NRC_j + \sum_{k=1}^{R} p_k \Delta l_k \),

where \( \Delta NRC_j \) the change of nonrenewable resource cost of mode for activity j is and \( \Delta l_k \) is the change of k-th resource tardiness. As activity j has no successor, \( \Delta l_k \) can be simply calculated. If \( \Delta c \) is negative, the mode change for activity j \( \in \) NSA is reasonable. We used this local search procedure described by Khalilzadeh et al.\textsuperscript{7} as "Mode assignment modification - Part II" in our algorithm.

4.9 Updating the Best Solutions

As aforementioned, the HBSO algorithm keeps the best food source achieved so far. Thus, after assessing all the food sources at iteration, if the existing food source is better than the best food source kept in the memory, the best food source is replaced by the current best food source.

5. Experimental Analysis

In this part, we analyze computational results of the algorithm. All procedures have been programmed and performed with C#.NET\textsuperscript{20} platform on a personal Computer with Core 2 Duo 2.53 GHz CPU and 4 GB RAM.

5.1 Sample Problems

We used sample problems library of PSPLIB\textsuperscript{8} and chose 3 sets of small-sized multimode project scheduling problems, j10, j16, j20 and one medium-sized problem of j30. Also, 2 sets of large-sized problems, j60 and j90 were created with the similar parameters as j30, but with 60 and 90 activities. Moreover, in order to detect the influence of resources in the problem, 2 extra sets of problems were produced by Progen9 named j30_r4_n4 and j60_r4_n4 respectively. All the parameters in these sets are same as those in sets j30 and j60, except for the number of resources, as there are 4 resources of each type.

Discrete Uniform Distribution (DUD) has been widely applied to considerable amount of papers in the related literature, e.g., Ranjbar et al.\textsuperscript{12} and Khalilzadeh et al.\textsuperscript{7} used DUD for resource ready times and due dates. Hence, in this paper we apply this type of distribution to select the parameters. The unit cost of nonrenewable resources were randomly chosen from DUD (2,6), the unit penalty cost of renewable resource tardiness were randomly chosen from DUD (10,30). The ready times of renewable resources were randomly generated from DUD (0,15), and finally; the renewable resource due dates were randomly picked from DUD (5,15) plus the amount of their ready time. Lastly, we set daily discount rate \( \alpha = 0.01 \) (\%).

5.2 Algorithm Validity

As there are no solved instances for this new type of project scheduling problem, we improved some instances whose optimal objective function rates are found. Then we
resolved these instance problems with our algorithm and examined the results. In order to create these instances, we used sample problems introduced in section 5.1, but we revised the due dates of renewable resources as the following: we produced a random feasible schedule for each instance, after allocating mode with the least nonrenewable resource costs to each activity of the project. In this schedule, we defined the release time of each renewable resource. Then, we set the due date for each renewable resource equal to its release time. Therefore, this schedule has zero tardiness penalty cost and the amount of its objective function is equal to the cost of nonrenewable resources. As we allocated the least nonrenewable cost modes to the activities, this schedule is optimum and the optimal rate of its objective function is known.

All instances were altered with the above process and resolved with the HBSO algorithm. The termination criterion was set to 600 schedules. In order to evaluate the validity of the algorithm, d, the percent deviation of the objective function rate from optimum, calculated for each resolved instance:

Where, \( d = 100 \times (Z_p - Z_{opt}) / Z_{opt} \), \( Z_p \) is the objective function rate of the best answer attained by the HBSO algorithm and \( Z_{opt} \), the optimal objective function value of the test problem. Table 1 illustrates the average and standard deviation of \( d \) for each instance. The low rates of average and standard deviation of \( d \) disclose worthy performance of the proposed algorithm with small CPU time.

### 5.3 Algorithm Robustness

In order to determine the robustness of the HBSO algorithm, we have applied it to numerous test problems.

<table>
<thead>
<tr>
<th>Problems set</th>
<th>Average CPU time (millisecond)</th>
<th>Average ( d )</th>
<th>Standard deviation of ( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>J10 (536 problems)</td>
<td>226</td>
<td>0.98</td>
<td>1.76</td>
</tr>
<tr>
<td>J16 (550 problems)</td>
<td>363</td>
<td>2.98</td>
<td>3.93</td>
</tr>
<tr>
<td>J20 (554 problems)</td>
<td>441</td>
<td>3.85</td>
<td>4.59</td>
</tr>
<tr>
<td>J30 (640 problems)</td>
<td>664</td>
<td>6.02</td>
<td>6.72</td>
</tr>
<tr>
<td>J60 (640 problems)</td>
<td>1443</td>
<td>8.73</td>
<td>8.92</td>
</tr>
<tr>
<td>J90 (640 problems)</td>
<td>2352</td>
<td>9.62</td>
<td>9.85</td>
</tr>
<tr>
<td>J30-r4-n4 (640 problems)</td>
<td>1237</td>
<td>6.38</td>
<td>7.11</td>
</tr>
<tr>
<td>J60-r4-n4 (640 problems)</td>
<td>2706</td>
<td>9.58</td>
<td>10.38</td>
</tr>
</tbody>
</table>

Table 2. Algorithm robustness check

<table>
<thead>
<tr>
<th>Problem set</th>
<th>120 schedules</th>
<th>900 schedules</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average CPU time (millisecond)</td>
<td>Average Percent deviation</td>
</tr>
<tr>
<td>J10</td>
<td>57</td>
<td>4.98</td>
</tr>
<tr>
<td>J16</td>
<td>91</td>
<td>3.82</td>
</tr>
<tr>
<td>J20</td>
<td>112</td>
<td>3.34</td>
</tr>
<tr>
<td>J30</td>
<td>176</td>
<td>3.36</td>
</tr>
<tr>
<td>J60</td>
<td>381</td>
<td>2.77</td>
</tr>
<tr>
<td>J90</td>
<td>627</td>
<td>1.72</td>
</tr>
<tr>
<td>J30-r4-n4</td>
<td>335</td>
<td>3.32</td>
</tr>
<tr>
<td>J60-r4-n4</td>
<td>718</td>
<td>2.23</td>
</tr>
</tbody>
</table>

Each test problem has been resolved a number of times and \( d \), the percent deviation of objective function rate for each test problem has been calculated.

15 test problems have been randomly chosen from each set of problems and have been solved thirty times by the HBSO algorithm, and each time stopped by two termination criteria, generation of 120 and 900 schedules.

To examine the robustness of the algorithm, \( d' \) for each test problem has been calculated where \( d' = 100 \times Sd(Z) / E(Z) \), \( Z \) is the objective function value of the best solution obtained by solving the problem for \( i \)th run, \( Sd(Z) \) standard deviation of \( Z \), and \( E(Z) \) is the mean of \( Z \). The average of \( d' \) for the problems of each set has been displayed in Table 2.

### 6. Conclusion

In this study, we presented MRCPS-P-DCTWRTPC problem as a resource-constrained project scheduling problem with the objective of minimization of the discounted cash flows of both renewable and nonrenewable resource costs. We developed the mixed integer programming model of the similar problem introduced by Khalilzadeh et al. Successively, we applied HBSO algorithm to tackle the proposed project scheduling problem. In order to create feasible schedules, we implemented the HBSO algorithm for activity mode assignment and improved a novel heuristic method to prioritize activities for parallel scheduling scheme. Two improvement heuristics, delay local search and local left shift, in agreement with two mode assignment modification methods, were used to develop the answers. The experimental results
showed appropriate algorithm robustness in resolving diverse test problems particularly with great number of iterations. Also, the validity analysis revealed small deviations from the optimal results for the problem instances in rational operating time.

7. References