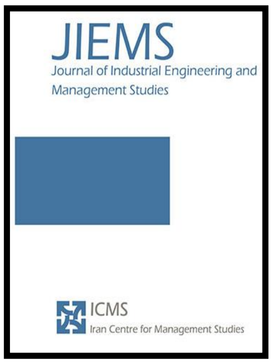
JIEMS

Journal of Industrial Engineering and Management Studies

Vol. 4, No. 2, 2017, pp. 52-63

DOI: 10.22116/JIEMS.2017.54606

www.jiems.icms.ac.ir


JIEMS
Journal of Industrial Engineering and Management Studies
ICMS
Iran Centre for Management Studies

A Cuckoo search algorithm (CSA) for Precedence Constrained Sequencing Problem (PCSP)

Mansoureh Maadi[1,*], Mohammad Javidnia[1]

Abstract

Precedence constrained sequencing problem (PCSP) is related to locate the optimal sequence with the shortest traveling time among all feasible sequences. In PCSP, precedence relations determine sequence of traveling between any two nodes. Various methods and algorithms for effectively solving the PCSP have been suggested. In this paper we propose a cuckoo search algorithm (CSA) for effectively solving PCSP. CSA is inspired by the life of a bird named cuckoo. As basic CSA at first was introduced to solve continuous optimization problem, in this paper to find the optimal sequence of the PCSP, some schemes are proposed with modifications in operators of the basic CSA to solve discrete precedence constrained sequencing problem. To evaluate the performance of proposed algorithm, several instances with different sizes from the literature are tested in this paper. Computational results show the good performance of the proposed algorithm in comparison with the best results of the literature.

Keywords: Precedence constrained sequencing problem, Mixed integer programming, Cuckoo search algorithm.

Received: May 2017-20

Revised: June 2017-18

Accepted: September 2017-06

1. Introduction

The aim of precedence constrained sequencing problem is finding the optimal sequence with the shortest traveling time through all nodes, visiting each node only once and considering the precedence relations between given pair of nodes. In PCSP, precedence relations determine sequence of traveling between any two nodes. PCSP has a number of practical applications especially in industrial planning including scheduling (Chen 1990, He and Kusiak 1992, Savelsbergh and Sol 1995, Renaud, Boctor and Ouenniche 2000, Lambert 2006), assembly flow (Duman and Or 2004), logistic (Moon et al. 2002, Chan and Chung, 2004, Altimarmak et al. 2006), project management and networking (Gen, Cheng and Lin 2008). The mentioned applications of the PCSP have attracted many researchers to this problem. Several methods that have been suggested to solve PCSP in the literature can be categorized into exact, heuristic and meta-heuristic algorithms. In this section we only tend to review heuristic and Meta – heuristic approaches to solve the PCSP.

* Corresponding Author; m_moadi@du.ac.ir

[1] School of engineering, Damghan University, Damghan, Iran.

He and Kusiak presented a heuristic algorithm to solve the scheduling model of a single machine with precedence constraint and sequence dependent changeover cost (He and Kusiak 1992). Savelsbergh and Sol proposed a heuristic algorithm to solve the Dial-A-Ride problem that can be formulated as a PCSP (Savelsbergh and Sol 1995). In this problem a vehicle should transport a number of passengers while each passenger is transported from a specific location to a distinct destination point. Renaud, Boctor and Ouenniche developed a heuristic model to solve a pickup and delivery traveling salesman problem that can be displayed as a PCSP (Renaud, Boctor and Ouenniche 2000). Duman and Or proposed a heuristic algorithm to solve a PCSP in an assembly problem of printed circuit board (Duman and Or 2004).

Pedersen, Rasmussen and Andersen proposed a Tabu search algorithm to solve practical large scale precedence scheduling problem with elastic jobs (Pedersen, Rasmussen and Andersen 2007). In this problem the jobs are processed on three servers and restricted by precedence constraints, time windows and capacity limitations. In this paper using a new method for approximating the server exploitation of the elastic jobs, a Tabu search algorithm is introduced to solve the problem. Li, Ong and Nee developed a hybrid genetic algorithm and simulated annealing approach for optimization of process plans for prismatic parts that can be formulated as a PCSP (Li, Ong and Nee 2002). In this paper genetic algorithm is carried out to generate some initial good process plans and simulated annealing algorithm is employed to search for alternative optimal or near optimal process plans. Moon et al. introduced a genetic algorithm with a priority based encoding method to solve the PCSP based on supply chain planning model (Moon et al. 2002). Chan and Chung presented a multi criterion genetic algorithm for order distribution in a demand driven supply chain that can be represented as PCSP (Chan and Chung 2004). Altiparmak et al. proposed a multi objective genetic algorithm for a single source and single product supply chain network design problem and Altiparmak et al. presented a multi objective genetic algorithm for a single source and multi product supply chain network design problem that can be represented as PCSP (Altiparmak et al. 2006, Altiparmak et al. 2007).

Su developed a unique reasoning method supported by the artificial intelligence technique of case based reasoning to solve assembly sequence planning problem that can be modeled as PCSP (Su 2007). In this paper to enhance the efficiency of the reasoning procedure, genetic algorithm is designed to automatically inferring of the reference assembly sequence. Gen, Cheng and Lin presented different types of network models with precedence constraints (Gen, Cheng and Lin. 2008). In this paper the models are shown as priority based representation and multi objective genetic algorithm are proposed to solve them. Gen, Lin and Zhang after introducing a network model with different pickup and delivery points as a PCSP, proposed a genetic algorithm to solve it (Gen, Lin and Zhang 2009). Yun and Moon introduced a topological sort based representation procedure to display PCSP and then proposed a genetic algorithm to solve various types of PCSP (Yun and Moon 2011). Yun, Chung and Moon developed a hybrid genetic algorithm with adaptive local search scheme to solve different types of PCSP (Yun, Chung and Moon 2013). Regarding numerous practical and theoretical papers with subject of the PCSP in the literature, it can be said that PCSP is a useful tool for variety of industrial planning and scheduling problems. However it can be also seen that the most approaches may not solve various types of PCSPs efficiently because of computational complexities and complicated precedence constraint (Yun and Moon 2011). In the literature, only two papers present algorithms to solve different types of PCSP instances with different number of nodes and this paper presents a cuckoo search algorithm that solve different complicated PCSP instances and has better performance rather than other algorithms of the literature.

This paper is organized as follows: In the next section, mathematical modeling of the PCSP is described. Basic cuckoo search algorithm is presented in the section 2. Proposed cuckoo search algorithm for solving PCSP is explained in section 3 following by computational results of proposed algorithm using various instances of the PCSP in section 4. Ultimately conclusions of the paper are investigated in section 5.

2. Problem modeling and description

As said before, the aim of the PCSP is finding the optimal sequence of nodes among all feasible sequences in a graph. A feasible sequence passes through each node in a given directed graph once and cannot visit any nodes at the same time. In PCSP there is no cycle because a linear sequence is impossible if a direct graph for the PCSP has a loop. In fact PCSP is a type of directed acyclic graph.
In this paper the proposed mixed- integer programming (MIP) model of the PCSP that was developed by Yun and Moon in 2011 is used for minimizing the total travelling time (Yun and Moon 2011).The following notations are used in this model.

i, j	Node index, i, j=1…I, where I is the number of nodes.
TT_{ij}	Travelling time from nodes i to j.
AT_i	Arrival time at node i.
S_1	Set of nodes (i, j), where node i is visited before node j.
S_2	Set of nodes (i, j), where nodes i and j can be visited in any feasible sequence.
LP	Arbitrary large positive number.

The variable of this model is defined as follows:

$$x_{ij} = \begin{cases} 1 & if\ node\ i\ is\ visited\ before\ j \\ 0 & Otherwise \end{cases}$$

Assume E be the finding a node with the maximal arrival time, so E can be presented as follows:

$$E = \max_{\forall i}\{AT_i\} \tag{1}$$

Using above definition the MIP model of the PCSP can be described as follows:

$$\text{Minimize} \quad E = \max_{\forall i}\{AT_i\} \tag{2}$$

Subject to:

$$AT_j - AT_i \geq TT_{i,j} \qquad \forall (i,j) \in S_1 and\ i \neq j \tag{3}$$

$$AT_i - AT_j + LPx_{ij} \geq TT_{ij}x_{ij} + TT_{ji}x_{ji} \qquad \forall (i,j) \in S_2 and\ i \neq j \tag{4}$$

$$x_{ij} + x_{ji} = 1 \qquad \forall (i,j) \in S_2 and\ i \neq j \tag{5}$$

$$x_{ij} = 0, 1 \qquad \forall (i,j) \in S_2\ \ and\ i \neq j \tag{6}$$

Constraints (3) and (4) ensure that each two nodes of the graph cannot be visited at the same time. Constraint (5) is related to the acyclic characteristic of the graph of PCSP. This constraint guarantees the existence of a feasible sequence among the nodes with precedence relations. Finally constraint 6 represents that all variables of the model are zero-one.

3. Basic cuckoo search algorithm

Cuckoo search algorithm (CSA) is a meta- heuristic algorithm, developed by Yang and Deb in 2009 and has been founded to be efficient in solving different optimization problems (Yang and Deb 2009). Recent studies show that CSA is potentially far more efficient than particle swarm optimization and genetic algorithms (Yang and Deb 2009). This algorithm has been applied in many areas of optimization and computational intelligence such as: scheduling (Dasgupta and Das 2015), design (Gandomi, Yang and Alavi 2013), and network (Dhivya, Sundarambal and Anand 2011; Dhivya, Sundarambal 2011; Moravej and Akhlaghi 2013) reliability (Valian, et al. 2013). For more information about different applications of CSA researchers are referred to review paper of Yang and Deb (Yang and Deb 2014). The excellent performance of CSA to solve different optimization problems can be a motivation to apply this algorithm in other scopes of optimization problems such as PCSP.
In the nature, Cuckoos are brood parasites, laying their eggs in the nest of other birds though they may remove other’s eggs to increase the hatching probability of their own eggs. In this process if a host bird discovers that the eggs are not its own, it will throw away the eggs of cuckoo or will build a new nest elsewhere and abandon its previous nest. Cuckoos use different strategies to increase the probability of having a new cuckoo by the host bird and reduce the probability of abandoning eggs by the host birds. In this way, using these facts in cuckoo’s life, in cuckoo search algorithm in each iteration worst solutions of the problem space as the nests that are identified by the host birds are discarded and new solutions as the new nests for host birds are searched using levy flight operator. Levy flight that has been used wildly in optimization problems is named by a French mathematician named levy. In fact levy flight represents a model of random walks characterized by their step length which obey a power-law distribution. Researchers have shown that the search for preys by hunters follows typically the same characteristics of levy flights (Yang and Deb 2009). Using these operations whole solution space of the optimization problem is being explored. To simplify the description of the basic CSA we use the following three idealized rules. 1- each cuckoo only lay one egg at a time and dumps it in a randomly chosen nest.2- the best nests with high-quality eggs will be carried over to the next generations 3- the number of available host nests is fixed and the egg laid by a cuckoo is discovered by the host bird with a probability of $p_a \in (0\ 1)$, in this case a host bird can either get rid of the egg or simply abandon the nest and build a completely new nest. Equation 7 is used to produce a new solution x_i^{t+1} for cuckoo i in basic CSA.

$$x_i^{t+1} = x_i^t + \alpha \oplus levy\ (s, \lambda) \tag{7}$$

Where α ($\alpha > 0$) is the step size. The step length follows the levy distribution:

$$levy\ (s, \lambda) \sim S^{-\lambda} \qquad (1 < \lambda \le 3) \tag{8}$$

Which has an infinite variance with an infinite mean (Yang and Deb 2009). In equation 8, S is step size drawn from a levy distribution. The pseudo- code of the basic CSA is represented in algorithm 1.

```
Algorithm1: basic cuckoo search algorithm
Input: objective function f(x), $x = (x_1, x_2, \dots, x_n)$, maxiteration,
Numbers of host nests (n), eggs discover probability ($p_a$),

Output: the best solution

Generate initial population of n host nests $x_i$ $(i = 1,2, \dots, n)$
While ($t < maxiter$) or (stop criterion) do
    Get a cuckoo randomly (say a) by levy flights
    Evaluate its quality (fitness function (f(x))
    Choose a nest among n (say b) randomly
    If $f(a)$ is better than f(b)
        Replace the solution b by new solution
    End
    Rank the fitness of the solutions/nests
    Abandon a fraction $p_a$of worse nests
    Keep the best solutions
    Find the current best solution
End
Post process results and visualization
```

4. Proposed CSA to solve the PCSP

In this section the basic CSA is developed to solve the PCSP. Since CSA at first was introduced for solving continuous problems, in this section to use CSA for the PCSP, we need a discrete CSA, hence to apply CSA to discrete search space, the standard arithmetic operators of the basic CSA need to be redefined according to the characteristics of the PCSP. The detailed procedures of implementing the proposed CSA are declared in the following subsections. Also the pseudo- code of the proposed CSA is described in algorithm 2.

```
Algorithm 2: proposed CSA to solve PCSP
Input: number of nodes ($n$), Traveling time Matrix (T), Number of host nests ($N$),maximum number of
iterations (maxiter) and eggs discover probability ($p_a$).
Output: An approximation of an optimal solution to the PCSP instance.

Initialize a population of N host nests by using section 4.1 as nests
for each nest ($x_i$) in nests
    Calculate cost (i)  using equation (10)
End
foriter=0 to maxiter
    Generate a cuckoo egg ($x_j$) using section 4.2 from random nest
    Calculate cost (j)using equation (10)
    if (cost (j) < cost (i))
        $x_i \leftarrow x_j$
        cost (i) ← cost (j)
    End
    Abandon a fraction $p_a$ of the worst nests
    Build new nests at new locations using section 4.3 to replace nests lost
    Keep the best solutions (or nests with quality solutions);
    Rank the solutions and find the current best
    iter =iter+1;
End
Return the best nest as the result
```

4.1. Generating initial population of host nests

In order to solve an optimization problem using CSA, like other meta-heuristic algorithms, the values of the problem should be formed as an array. This array is called host nest in CSA. In PCSP a host nest is defined as an array of $1 \times \mathrm{n}$ that n is the number of nodes of the problem. For treating precedence constraints, a priority–based representation procedure should be applied in the PCSP. Among different presented procedures, in this paper the topological sort (TS)-based representation procedure of Yun and Moon is used to represent host nests. This procedure can effectively treat precedence constraints in the PCSP (Yun and Moon 2011). Using this procedure, the various types of feasible sequences can be generated and subsequently effective and different initial host nests can be produced.

Generally, a precedence relation between nodes i and j is defined as predecessor or successor of each node in any sequence. If each node to be visited before itself in any sequence, it is called a cycle. To represent precedence relations in the PCSP a precedence matrix $D = [d_{ij}]$ should be defined where:

$$d_{ij} = \begin{cases} 1 & if\ node\ i\ is\ visited\ immedietely\ before\ j \\ null & Otherwise \end{cases} \tag{9}$$

In this matrix $d_{ij} = 1$ indicates the precedence relation (i, j). Regarding a precedence matrix, a directed graph can be depicted. An example of a precedence matrix and related directed graph is shown in table 1 and figure 1.

Table 1. An example of a precedence matrix.

Node number	1	2	3	4	5	6	7	8	9	10
1			1		1					
2				1	1					
3					1	1				
4					1		1			
5								1		
6									1	
7								1	1	
8										1
9										1
10										

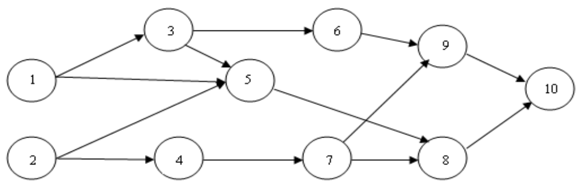


Figure 1. Related directed graph of table 1.

In table 1, for example $d_{35} = 1$ means that node 3 should be visited before node 5 in a linear sequence. Also d_{ij}=null indicates that there is no relationship between two nodes i and j. as can be seen in this table, columns 1 and 2 with values of null show that nodes 1 and 2 have no predecessors and row 10 with values of null indicates that node 10 doesn't have any successor.
To generate a feasible sequence of nodes as an initial host nest in the PCSP, considering a precedence matrix and its directed graph, at first a node with no predecessor is selected randomly. This node should be placed in the first cell of the host nest. For example in figure 1 nodes1 or 2 can be selected initially. Secondly the selected node and its arcs should be removed from the directed graph and in modified directed graph another random node with no predecessor is selected and placed in the second cell of the host bird. After that, the selected node and its arcs should be removed and new modified directed graph is gained. This process will be continued until all nodes of the graph to be removed from the graph and placed in the host nest. This host nest shows a feasible sequence of nodes and is named a feasible initial host nest. Figure 2 shows a feasible sequence of nodes of figure 1 as a host nest implementing TS-based representation procedure.

$$\text{host nest} = [1, 3, 6, 2, 4, 5, 7, 9, 8, 10\ \ ]$$

Figure 2. An example of a feasible host nest.

According to TS-based representation procedure, regarding the number of initial population, different initial host nests can be produced to start the proposed algorithm. In figure 3 considering 3 as the number of initial population, 3 feasible sequences or initial host nests related to directed graph of figure 1 as initial population are produced.

$$\text{host nest1} = [1, 2, 4, 3, 5, 7, 6, 8, 9, 10]$$
$$\text{host nest2} = [1, 3, 2, 4, 5, 6, 7, 9, 8, 10]$$
$$\text{host nest3} = [1, 3, 6, 2, 4, 5, 7, 8, 9, 10]$$

Figure 3. Initial population of habitats

In the PCSP, the cost function of the problem for the *k*th host nest is defined as follows that is considered as the objective function of the problem:

$$\text{cost (k)} = min\left\{\max_{\forall i}\{AT_i\}\right\} \tag{10}$$

4.2. Generating cuckoo eggs

After generating initial population of host nests, according to the basic CSA, cuckoos search for new nests via levy flight operator. Because levy flight is designed for searching in continuous space and PCSP is a discrete problem, in proposed algorithm, a new operator is developed to search new nests for cuckoo's egg laying. In this way considering a cuckoo in a host nest, following steps are designed for generating a cuckoo egg in new nest:
Step1: At first two cells of the current host nest array are selected randomly.
Step2: to generate a new nest for an egg, the node in the forward selected cell is placed in another selected cell. If this transfer doesn't disturb the precedence relations, other nodes of the host nest can be placed in new nest array sequentially, otherwise other two cells should be selected randomly and this process will be continued until a new nest array for an egg be generated.
Considering figure 2 as a primal host nest, figure 4 shows an example of implementation of generating cuckoo eggs operator.

$$\text{Primal host nest} = [1,3,6,2,4,5,7,9,8,10]$$
$$\text{New nest for egg} = [1,3,5,6,2,4,7,9,8,10]$$

Figure 4. implementation of generating cuckoo eggs operator

In figure 4 at first cell number 3 and cell number 6 are chosen randomly, then the node of 5 is transmitted to the cell number 3 and other nodes of the primal host nest are place in the new nest for egg array subsequently. This new nest is a feasible solution of the PCSP.

4.3. Building new nests for host birds after abandoning their nests

As said before, in the nature, if a host bird discovers that the eggs are not its own it will throw away the eggs of cuckoo or will build a new nest elsewhere and abandon its previous nest. In this way some new nests will be generated as the new nests of host birds. So, after generating new eggs in previous section, according to the basic CSA, a fraction of $p_a$of the worst nests will be omitted and new host nests are generated using following steps.

Step1: consider a detected host nest array and an array with n cells as the new host nest.

Step2: select a random number between 1 and $\left\lfloor \frac{\mathrm{n}}{2} \right\rfloor$as r_1.

Step3: copy the nodes in cell 1 to cell $\mathrm{r}_1$of the detected host nest array to the new host nest array.

Step4: from cell$\mathrm{r}_1 + 1,$ using TS-based representation procedure of section 4.1 set other nodes sequentially.

It is notable that new host nest is a feasible host nest. Figure 5 shows an example of a detected host nest and a new one. In this example $\mathrm{r}_1$is considered as 3.

$$\text{detected host nest} = [1,2,4,3,5,7,6,8,9,10]$$
$$\text{new host nest} \qquad = [1,2,4,7,3,5,8,6,9,10]$$

Figure 5: An example of Building new nest after abandoning detected nest with $\mathbf{r_1 = 3}$.

5. Computational results

In the literature, two types of PCSP instances have been introduced and applied to evaluate different proposed algorithms. The first type uses the conventional PCSP instance that is suggested by He and Kusiak (He and Kusiak 1992). For this type, the optimum value of the cost function is determined. The second type is consisting of various PCSP instances with different sizes of nodes that have not the optimum value of cost function. In the literature, instances of this type have been evaluated in terms of the value of the cost function and CPU time. In this section the results of applying proposed CSA on two types of PCSP instances are evaluated and compared to other recent proposed algorithms of the literature. Throughout the experiments the following parameter values are used for proposed CSA. The number of host nests is considered as $\left\lfloor \frac{3\times n}{2} \right\rfloor$ that n is the number of nodes of the problem and p_a is set 0.2. The number of iterations is considered as 150.

The results of applying the proposed algorithm on the first type instance is shown in table 2. In this table the results of the proposed algorithm is compared to the results of two algorithms of the literature. As that can be seen, the proposed algorithm is able to find the optimum value of total traveling time of 26 and the best sequence of 3-5-7-1-4-2-6.

Table 2. Computation results for the first type instance of PCSP

Approach	The best value of traveling time	The best sequence
CPLEX optimization approach (Yun and Moon, 2011)	26	3-5-7-1-4-2-6
He and Kusiak's approach 1992	31	1-3-5-2-4-7-6
HGA approach of Yun, Chung and Moon 2013	26	3-5-7-1-4-2-6
Proposed CSA approach	26	3-5-7-1-4-2-6

The computational results of proposed algorithm for second type of the PCSP instances are shown in table 3. In this table the number of nodes of different instances is presented in the first column. Other columns show the performances of two algorithms of the literature and the proposed CSA. According to table 3, the proposed algorithm has the better performance rather than other two algorithms and the best value of cost function of proposed CSA in all instances have a significant difference with other two algorithms. It is notable that CPU time generally is not comparable, but the proposed algorithm is able to solve PCSP instances in an acceptable time. Table 4 represents the best sequences of the second type of the PCSP instances that are related to the instances of table 3. The second type of PCSP instances with all information can be found in the paper of Yun and Moon (Yun and Moon 2011).

Table 3. Performance of the proposed CSA and two algorithms of the literature to solve the second type of PCSP instances

No. of node	GA approach of Yun and Moon 2011		HGA approach of Yun, Chung and Moon 2013		Proposed CSA	
	Best value	CPU time (Sec.)	Best value	CPU time (Sec.)	Best value	CPU time (Sec.)
20	127	2.62	127	2.11	**89**	**0.075**
30	196	3.07	196	2.79	**149**	**0.212**
40	212	3.62	212	3.11	**189**	**0.414**
50	271	3.98	269	3.45	**212**	**0.976**
60	332	5.77	329	5.23	**289**	**1.802**
70	392	6.21	384	5.88	**323**	**2.815**
80	426	7.23	423	6.96	**388**	**3.318**

Table 4. The best sequences of nodes related to table 3

No. of node	Best value	Best sequence
20	89	1 4 3 2 6 7 9 14 5 13 17 12 8 16 19 11 10 15 18 20
30	149	3 2 6 7 1 4 10 5 9 12 8 11 15 14 13 16 18 17 22 19 20 24 25 27 21 23 28 26 29 30
40	189	1 4 3 5 10 2 9 6 8 13 14 7 20 19 11 18 24 15 23 12 28 16 17 33 22 21 25 26 27 29 30 31 32 35 34 36 37 39 38 40
50	212	3 7 2 8 1 4 13 6 5 17 12 11 10 23 22 9 14 27 32 15 28 33 38 16 37 20 19 21 36 18 26 41 25 42 24 43 30 29 31 48 35 34 39 40 44 46 45 49 47 50
60	289	15 1 4 3 7 2 6 8 5 11 10 12 14 9 13 16 17 18 21 22 20 23 19 24 25 26 30 27 29 31 32 28 33 34 38 42 43 37 44 41 49 36 40 35 54 59 39 46 47 48 45 53 52 51 50 58 55 57 56 60
70	323	3 4 1 2 7 6 8 5 9 13 17 12 18 11 10 23 14 15 28 16 22 21 20 26 27 19 25 24 29 30 31 32 33 35 34 39 38 40 37 36 41 47 42 43 46 45 48 44 50 49 51 55 54 61 60 52 53 59 58 64 63 68 57 65 56 62 66 69 67 70
80	388	2 3 5 9 4 8 1 7 6 13 11 10 12 14 19 16 15 18 17 20 24 25 21 26 27 32 22 31 30 23 28 29 34 33 35 37 36 38 42 43 41 40 44 39 45 46 49 54 51 48 50 47 52 53 58 57 59 56 60 66 55 61 62 65 64 63 68 69 67 71 70 74 73 78 75 72 79 76 77 80

In table 5, for each second type of PCSP instance, the best and the worst value of cost function which is the total traveling time in 20 iterations are shown. Also the mean and standard deviation of results are presented. Results show the efficiency of the proposed algorithm in different times of runs.

Table 5. The results of the 20 times run of the proposed algorithm

No. of node	Objective values			
	Min	Max	Mean	SD
20	89	96	90.85	2.277
30	149	162	155.05	4.904
40	189	208	199.25	5.290
50	212	226	218	6.026
60	289	319	298.95	8.413
70	323	343	333.80	8.811
80	388	428	404.20	13.590

6. Conclusion

The objective of the PCSP is to locate the optimal sequence with the shortest traveling time among all feasible sequences. Because of various applications of this problem, many researchers have developed various approaches to treat and solve the PCSP. In this paper, because of the discrete nature of the PCSP, a cuckoo search algorithm is modified according the characteristics of the PCSP to solve different types of this problem. In fact in this paper a discrete CSA is proposed with new operators to solve PCSP. Results of proposed CSA on different instances of the PCSP show the excellent performance of the proposed algorithm in comparison to other algorithms of the literature. Generally it can be said that the proposed algorithm outperforms the other algorithms of the literature in both aspects of the best value of cost function and computational time.

References

Altiparmak, F., Gen, M., Lin, L. and Paksov T., (2006), “A genetic algorithm approach for multi-objective optimization of supply chain networks”, *Computers and Industrial Engineering*, Vol. 51, No. 1, pp. 196-215.

Altiparmak, F., Gen, M., Lin, L., and Karaoglan, I., (2007). “A steady-state genetic algorithm for multi-product supply chain network design”, *Computers and Industrial Engineering*, Vol. 56, No. 2, pp. 521–537.

Chen, C. (1990). “AND/OR precedence constraint traveling salesman problem and its application to assembly schedule generation”. IEEE international conference on Systems, Man and Cybernetics. Proceedings. 560–562.

Chan, F. T. S., and Chung, S. H., (2004). “A multi-criterion genetic algorithm for order distribution in a demand driven supply chain”, *International Journal of Computer Integrated Manufacturing*, Vol. 17, No. 4, pp. 339–351.

Dasqupta, P., and Das, S., (2015). “A Discrete Inter-Species Cuckoo Search for flow shop scheduling problems”, *Computers and Operations Research*, Vol. 60, No. 3, pp. 111-120.

Dhivya1, M., and Sundarambal, M., (2011). “Cuckoo search for data gathering in wireless sensor networks”, *International Journal of Mobile Communications*, Vol. 9, No. 6, pp. 642-656.

Dhivya1, M., Sundarambal, M. and Anand L.N., (2011). “Energy efficient computation of data fusion in wireless sensor networks using cuckoo based particle approach (CBPA)”, *International Journal of Communications, Network and System Sciences*, Vol. 4, pp. 249-255.

Duman, E., Or, I., (2004). “Precedence constrained TSP arising printed circuit board assembly”, *International Journal of Production Research*, Vol. 42, No. 1, pp. 67–78.

Gandomi, A. H., Yang Xin-She and Alavi A. H., (2013). “Cuckoo search algorithm: a meta-heuristic approach to solve structural optimization problems”, *Engineering with Computers*, Vol. 21, No. 1, pp. 17-35.

Gen, M., Cheng, R., and Lin, L., (2008). *Network models and optimization: multi objective genetic algorithm approach*, Springer. England.

Gen, M., Lin, L., and Zhang, H., (2009). “Evolutionary techniques for optimization problems in integrated manufacturing system: State-of-the-art-survey”, *Computers and Industrial Engineering*, Vol. 56, No. 3, pp. 779–808.

He, W., Kusiak, A., (1992). “Scheduling manufacturing systems”, *Computers in Industry*, Vol. 20, pp. 163–175.

Lambert, A. J. D., (2006). “Exact methods in optimum disassembly sequence search for problems subject to sequence dependent costs”, *Omega*, Vol. 34, pp. 538–549.

Li,W. D., Ong, S. K. and Nee, A. Y. C., (2002). “Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts”, *International Journal of Production Research*, Vol. 40, No. 8, pp. 1899-1922.

Moon, C., Kim, J., Choi, G., and Seo, Y., (2002). “An efficient genetic algorithm for the traveling salesman problem with precedence constraints”, *European Journal of Operational Research*, Vol. 140, No. 3, pp. 606–617.

Moravej, Z., Akhlaghi, A., (2013). “A novel approach based on cuckoo search for DG allocation in distribution network”, *Electrical Power and Energy Systems*, Vol. 44, pp. 672-679.

Pedersen, C. R., Rasmussen, R. V. and Andersen, K. A., (2007). “Solving a large-scale precedence constrained scheduling problem with elastic jobs using Tabu search”, *Computers and Operations Research*, Vol. 34, No. 7, pp. 2025-2042.

Renaud, J., Boctor, F. F., and Ouenniche, J., (2000). “A heuristic for the pickup and delivery traveling salesman problem”, *Computers and Operations Research*, Vol. 27, pp. 905–916.

Savelsbergh, M., and Sol, M., (1995). "The general pickup and delivery problem", *Transportation Science*, Vol. 29, pp. 17–29.

Su, Q., (2007). "Applying case-based reasoning in assembly sequence planning", *International Journal of Production Research*, Vol. 45, No. 1, pp. 29–47.

Valian, E., Tavakoli, S., Mohanna, S., and Haghi, A., (2013). "Improved cuckoo search for reliability optimization problems", *Computer and Industrial Engineering*, Vol. 64, pp. 459-468.

Yang, X. S., Deb, S., (2009). "Cuckoo search via le´vy flights. In: Nature & biologically inspired computing", NaBIC 2009. World congress on, IEEE, Proceedings. 210–214.

Yang, X. S., and Deb, S., (2014). "Cuckoo search: recent advances and applications", *Neural Computing and Applications*, Vol. 24, No. 1, pp. 169-174.

Yun, Y. S., and Moon, C., (2011). "Genetic algorithm approach for precedence constrained sequencing problems", *Journal of Intelligent Manufacturing*, Vol. 22, No. 3, pp. 379–388.

Yun, Y. S., Chung, H., and Moon, C., (2013). "Hybrid genetic algorithm approach for precedence-constrained sequencing problem", *Computers and Industrial Engineering*, Vol. 65, pp.137-147.