International Journal of Civil Engineering

Structural Engineering

Charged system search adopted for solution of traveling salesman problem: An application to single-row facility layout problem

A. Kaveh[1,*], H. Safari[2]

Received: November 2013, Accepted: February 2014

Abstract

This paper presents an algorithm based on CSS for discrete problems with the focus on traveling salesman problem. The CSS algorithm, based on some principles from physics and mechanics, utilizes the governing Coulomb law from electrostatics and Newtonian laws of mechanics. However, the CSS is more suitable for continuous problems compared to discrete problems. In this paper, a local search method and nearest neighbor are added to CSS for discrete problems with the focus on traveling salesman problem (TSP). The proposed algorithm is used to solve the TSP, and a method is presented for the solution of the single row facility layout problem (SRFLP). To show the efficiency of the new algorithm, the results are compared to those of some benchmark problems reported in the recent literatures.

Keywords: *Charged system search (CSS), Traveling salesman problem, Discrete problems, Single row facility layout problem.*

1. Introduction

Traveling salesman problem (TSP) is an NP-hard problem, where for a number of cities with specified distance between them, a tour must be found such that the salesman goes only once to each of these cities and returns to the starting one [1], travelling a minimum distance. Finding an exact solution to this problem is very hard, and the exact solution may be obtained only for small-sized problems. To obtain the solution of problems with large number of cities, it is preferable to use heuristic algorithms. In recent years, many researchers have proposed various methods to solve this problem [2-8].

The CSS algorithm based on some principles from physics and mechanics, utilize the governing Coulomb law from electrostatics and Newtonian laws of mechanics [9].

After the introduction, Section 2 presents an overview of the standard CSS algorithm. The modified CSS algorithm for the solution of TSP is introduced in Section 3. Experimental results for TSP are presented in Section 4. A Review of the single row facility layout problem (SRFLP) is presented in Section 5. The modified CSS algorithm for solving SRFLP is introduced in Section 6, and experimental results for SRFLP are presented in Section7. Finally, conclusion is provided in Section 8.

** Corresponding author: alikaveh@iust.ac.ir*

1 Professor, Centre of Excellence for Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, Iran

2 Graduate student, School of Civil Engineering, University of Science and Technology, Narmak, Tehran-16, Iran

2. A Review of the Charged System Search Algorithm

The Charged system search (CSS) algorithm, proposed by Kaveh and Talathari [9], is a meta-heuristic algorithm for optimization problems. This algorithm takes its inspiration from the physic laws governing a group of charged particles, CPs. These charge particles are sources of the electric fields, and each CP can exert electric force on other CPs. Using the Newtonian mechanic laws, the movement of each CP due to the electric force can be determined. Some other applications of the CSS can be found in Refs. [10-13]. The CSS algorithm is summarized in a systematic form as follows:

Stage 1. Initialization

The initial positions of the CPs are randomly determined using a uniform source, and the initial velocities of the particles are set to zero. A memory is utilized to save a number of best results. This memory is called the Charged Memory (CM).

Stage 2. Determination of electric forces and the corresponding movements

- Force Determination. Each charged particle imposes electric forces on the other CPs according to the magnitude of its charge. The charge of the each CP is:

$$q_i = \frac{fit(i) - fitworst}{fitbest - fitworst} \tag{7}$$

Where fit (i) is the objective function value of the ith CP, *fitbest* and *fitworst* are the so far best and worst

fitness among all of the CPs, respectively.

In addition to electric charge, the magnitude of the electric forces exerted on the CPs is depended on their separation distance that is,

$$r_{ij} = \frac{\| \mathbf{X}_i - \mathbf{X}_j \|}{\| (\mathbf{X}_i + \mathbf{X}_j)/2 - \mathbf{X}_{best} \| + \varepsilon} \quad (8)$$

Where $\mathbf{X}_i$ and $\mathbf{X}_j$ are the position of the *i*th and *j*th CPs, and r_{ij} is the separation distance these CPs. $\mathbf{X}_{best}$ is the position of the best current CP, and ε is a small positive number to prevent singularity.

The probability of the attraction of the *i*th CP by the *j*th CP is expressed as:

$$p_{ij} = \begin{cases} 1 \Leftrightarrow \dfrac{fit(i) - fitbest}{fit(i) - fit(j)} > rand, or, fit(j) > fit(i) \\ 0 \Leftrightarrow else. \end{cases} \quad (9)$$

The electric resultant force $\mathbf{F}_{E,j}$, acting on the *j*th can be calculated by the following equation,

$$\mathbf{F}_{E,j} = q_j \sum_{i,i \neq j} \left(\frac{q_i}{a^3} r_{ij} \cdot w_1 + \frac{q_i}{r_{ij}^2} \cdot w_2 \right) \cdot p_{ji} \cdot (\mathbf{X}_i - \mathbf{X}_j),$$
$$\begin{cases} w_1 = 1, w_2 = 0 \Leftrightarrow r_{ij} < R \\ w_1 = 0, w_2 = 1 \Leftrightarrow r_{ij} \geq R \\ j = 1,2,...,N \end{cases} \quad (10)$$

- Movements Calculations. According to the determined forces, each CP moves to its new position, and attain velocity as:

$$\mathbf{X}_{j,new} = rand_{j1} \cdot k_a \cdot \frac{\mathbf{F}_j}{m_j} \cdot \Delta t^2 + rand_{j2} \cdot k_v \cdot \mathbf{V}_{j,old} \cdot \Delta t + \mathbf{X}_{j,old}, \quad (11)$$

$$\mathbf{V}_{j,new} = \frac{\mathbf{X}_{j,new} - \mathbf{X}_{j,old}}{\Delta t} \quad (12)$$

Where $rand_{j1}$ and $rand_{j2}$ are two random numbers that uniformly distributed in the range (0, 1). k_a is the acceleration coefficient, k_v is the velocity coefficient, and m_j is the mass of particle that is considered equal to q_j. The velocity coefficient controls the influence of the previous velocity of the particles. In other words, this coefficient is related to the exploration ability of the algorithm. The acceleration coefficient affects the force acting on each CP, or it influences the exploitation ability of the algorithm. An efficient optimization algorithm should perform good exploration in early iterations and good exploitation in last iterations. Thus, the magnitude of the k_a and k_v is set to 0.5 which are linearly increased and decreased, respectively. Thus, k_a and k_v are expressed as:

$$k_a = 0.5(1 + iter / iter_{\max}),\ k_v = 0.5(1 - iter / iter_{\max}) \quad (13)$$

Where *iter* is the current iteration number and $iter_{max}$ is the maximum number of iterations.

Stage 3. Boundaries Constraints Handling

For handling boundary constraints, a harmony search-based approach is used. According to method, any variable of each solution ($x_{i,j}$) that violates its corresponding boundary can be regenerated from CM as

- w.p. CMCR
- select a new value for variable from CM,
- w.p. (1−PAR) do nothing,
- w.p. PAR choose a neighboring value,
- w.p. (1−CMCR)
- select a new value randomly,

Where "w.p." is the abbreviation of "with the probability", CMCR (the Charge Memory Considering Rate) varying between 0 and 1 sets the rate of selecting a value in the new vector from historic values stored in CM, and (1−CMCR) sets the rate of randomly choosing one value from possible range of values. The value (1−PAR) sets the rate of doing nothing, and PAR sets the rate of choosing a value from neighboring the best CP. For further details, the reader may refer to Ref. [9].

Stage 4. Charged Memory (CM) Updating

If among all of the new CPs, there are better CP or CPs that have better objective function value than the worst ones in the CM, these should be included in the CM, and the worst ones in the CM are excluded from the CM.

Stage 5. Checking the Termination Criteria

Stages 2 and 3 are re-iterated until one of the specified terminating criteria is satisfied.

3. A Hybrid-Modified Charged System Search Algorithm

3.1. A modified charged system search algorithm

To introduce the modified charged system search algorithm, in the following a paragraph is quoted from [14]:

"One of the assumptions of meta-heuristics is that the time alters discretely. This means that all alterations in space−time are performed when all agents have created their solutions. For example, in the CSS algorithm, when the calculations of the amount of forces are completed for all CPs, the new locations of agents are determined (Stage 2).

In addition, CM updating is fulfilled after moving all CPs to their new locations. All these conform to discrete time concept. In the optimization problems, this is known as an iteration. In other words, the modification of the space−time for the multi-agent algorithms is often performed when an iteration is completed and the new iteration is not started yet.

Here, this assumption is discarded for the CSS algorithm and therefore a modified CSS is obtained. In the modified CSS, time changes continuously and after creating just one solution, all updating processes are performed. Using this version of CSS, the new position of

each agent can affect on the moving process of the subsequent CPs while in the standard CSS unless an iteration is completed, the new positions are not utilized. Based on this inference, the modified CSS is as follows:

- Stage 1: Initialization.

• This step is similar to the one provided previously.

The initial positions and velocities of CPs as well as the CM are initialized. A number associated to each CP is considered.

- Stage 2: Solution construction.

• Forces determination. The force vector for the *j*th CP is calculated by Eq. (7).

• New position creation. Each CP moves to the new position as defined in Eq. (11) and Eq. (12). It should be noted that in order to determine the location of each CP using Eq. (11), the recent location of the previous agents is utilized instead of the previous ones leading to the use of the pervious information directly after their generation. After moving the CP to its new position, the objective function is evaluated.

- Stage 3: CM (sources) updating.

• If the new CP vector is better than the worst one in the CM, it is included in the CM.

- Stage 4: Terminating criterion control.

• Stages 2 and 3 are repeated until a terminating criterion is satisfied."

3.2. Hybrid-modified charged system search algorithm for solving TSP

Changes have been made to the CSS algorithm as follows:

1. Both standard CSS and modified CSS, initial with positions are generated randomly and best of these solutions are stored in the CM (charge memory), but to improve the initial positions in the proposed algorithm, two series of solutions are generated, one of them, the main solutions and the other series are stored in CM. It is noteworthy that in this proposed algorithm, the particles in the CM can attract other particles. Thus, the main particles, unlike the two previous algorithms where particles are randomly generated, in this algorithm, the particles are placed on the CM are generated by NN (Nearest Neighbor) methods. NN starts with selecting a random number for the first city of a tour, then for selecting next city, the minimum distance in the rows of the previous city in distances matrix is found and selected, and this stage is repeated until all cities are chosen. The proposed algorithm uses a modified NN so that all possible scenarios are evaluated by the NN and the best of them are selected.
2. In this method, as the modified CSS, time is not assumed to be discrete, and after each displacement for a particle, CM update process is performed. Thus the impact of the particle on other particles by the new location and a new charge will be applied.
3. Each time the particles are evaluated by the objective function, a local search method is used to generate other solutions around this solution and if a better solution is found, the best solution is to replace the current solution. The method that has been used for this purpose, is a combination of mutation operators that can be utilized in genetic algorithms. The method used here is presented in Fig. 1.
4. In most cases, termination condition of the previous algorithms is based the maximum number of iterations. In the proposed algorithm, this condition has been changed, and if at a stage no improvement is observed, after certain number of non-improving iterations, the particles are reallocated. Then the same number of iterations is continued, and if still no improvement is achieved, then the algorithm stops.
5. At the end, when the algorithm is stopped, again using the local search described in the previous section, the best solution is examined for improvement, with the difference that, at this stage all possible solutions for improvement are investigated and the local search is carried out to a certain number performing replacement if any better solution is found.

4. Experimental Results for TSP

To show the efficiency of the proposed HMCSS+LS algorithm, results are compared with Co-adaptive Net [15], CONN [16], RABNET-TSP [5], ELC-LR [17], GA+GSTM [6], Chen & Chien [18], ASA-GS [19], and CGAS [7]; for benchmark problem from TSPLIB [20]; each problem is run for 10 trials. Results of average error and best error from best-known solutions are presented in Tables 2 and 3. The results of average error are also graphically shown in Figures 2 and 3. In addition, algorithms parameters are chosen such that the number of CP's equal to 30, radius of CP's is one, step for reallocate the CP's equal to 120 iteration and HS parameters, CMCR, PAR, k_a and k_v are shows in Table 1.

Table 1 The parameters of HMCSS+LS

	Problem size			
	<60	60-120	120-300	>300
CMCR	0.98	0.99	1	1
PAR	0.1	0.2	0.1	0.1
k_a	0.6	1	1	2
k_v	0.5	0.5	0.5	0.5

```
Tour: One solution for TSP.
dist: The objective function evaluation of Tour.
n: size of the problem.
According to the random (Rnd) number produced between (0-1);
if (Rnd<.2)
{
    T=Tour;
    I=a random city between solutions city;
    J=a random city between solutions city;
    I<J;
    According to the random (Rnd) number produced between (0-1);
    if (Rnd<.9)
    {
        Ts=T(J:-1:I); (selection a part of solution and inversion it)
    }Else
    {
        Ts=T(I:J); (selection a part of solution)
    }
    Tt=T-Ts; (remove Ts from Tour)
    for (counter=1:length(Tt)-1)
    {
        newTour= [Tt(1:counter),Ts,Tt(counter+1:nTt)]; (insert Ts into Tt)
        newdist= the objective function evaluation of newTour;
        if (newdist<dist)
        {
            dist=newdist;
            Tour=newTour;
        }
    }
    if (n<=30)
    {
        Repeat this process 1 times
    }Elseif (n>30 && n<=60)
    {
        Repeat this process 2 times
    }Elseif (n>60 && n<=120)
    {
        Repeat this process 3 times
    }Elseif (n>120 && n<=300)
    {
        Repeat this process 4 times
    }Elseif (n>300)
    {
        Repeat this process 5 times
    }
}
```

Fig. 1 Local search using a combination of mutation operators.

Table 2 Comparison of the average error and best error of HMCSS+LS with Co-adaptive Net [15], CONN [16], RABNET-TSP [5] and ELC-LR [17]

TEST PROBLEM	# OF NODE	CO-ADAPTIVE NET (2003)		CONN (2007)		RABNET-TSP (2009)		ELC-LR (2010)		HMCSS+LS (CURRENT METHOD)			
		Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Rank in Ave Error	Rank in Best Error
EIL51	51	2.89	0.94	2.85	-	2.69	0.23	0.94	-	0.47	0.00	1	1
BERLIN52	52	7.01	0.00	8.18	-	5.18	0.00	-	-	0.01	0.01	1	2
KROA100	100	1.31	0.57	2.57	-	1.13	0.24	1.63	-	0.41	0.00	1	1
PR144	144	-	-	2.34	-	-	-	-	-	0.31	0.00	1	1
CH150	150	3.23	1.78	5.50	-	3.22	1.13	0.61	-	0.66	0.38	2	1
KROA200	200	3.27	0.92	5.16	-	2.80	0.79	1.04	-	1.11	0.89	2	2
TS225	225	-	-	-	-	-	-	8.84	-	1.19	0.25	1	1
LIN318	318	4.31	2.65	-	-	3.97	1.92	0.32	-	2.25	1.84	1	1
PCB442	442	-	-	5.72	-	-	-	-	-	2.30	1.83	1	1
										Frequently rank		**1**	**1**

Table 3 Comparison of the average error of the HMCSS+LS, GA+GSTM[6], Chen & Chien [18], ASA-GS [19], and CGAS [7]

TEST PROBLEM	# OF NODE	GA+GSTM (2011)		CHEN&CHIEN (2011)		ASA-GS (2011)		CGAS (2012)		HMCSS+LS (CURRENT METHOD)			
		Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Ave Error (%)	Best Error (%)	Rank in Ave Error	Rank in Best Error
EIL51	51	-	-	0.30	0.23	0.67	0.67	-	-	0.47	0.00	2	1
BERLIN52	52	0.00	0.00	0.00	0.00	0.03	0.03	1.22	0.00	0.01	0.01	2	2
KROA100	100	1.18	0.00	0.42	0.00	0.01	0.01	0.73	0.00	0.41	0.00	2	1
PR144	144	1.08	0.00	-	-	0.01	0.00	-	-	0.31	0.00	2	1
CH150	150	0.64	0.46	0.55	0.00	0.16	0.04	-	-	0.66	0.38	4	3
KROA200	200	1.54	0.87	1.26	0.05	0.23	0.14	1.97	0.00	1.11	0.89	2	4
TS225	225	0.50	0.25	-	-	0.00	0.00	-	-	1.19	0.25	3	2
LIN318	318	3.31	0.98	-	-	0.84	0.66	-	-	2.25	1.84	3	3
PCB442	442	2.78	2.05	-	-	0.96	0.56	-	-	2.30	1.83	2	2
										Frequently rank		**2**	**1**

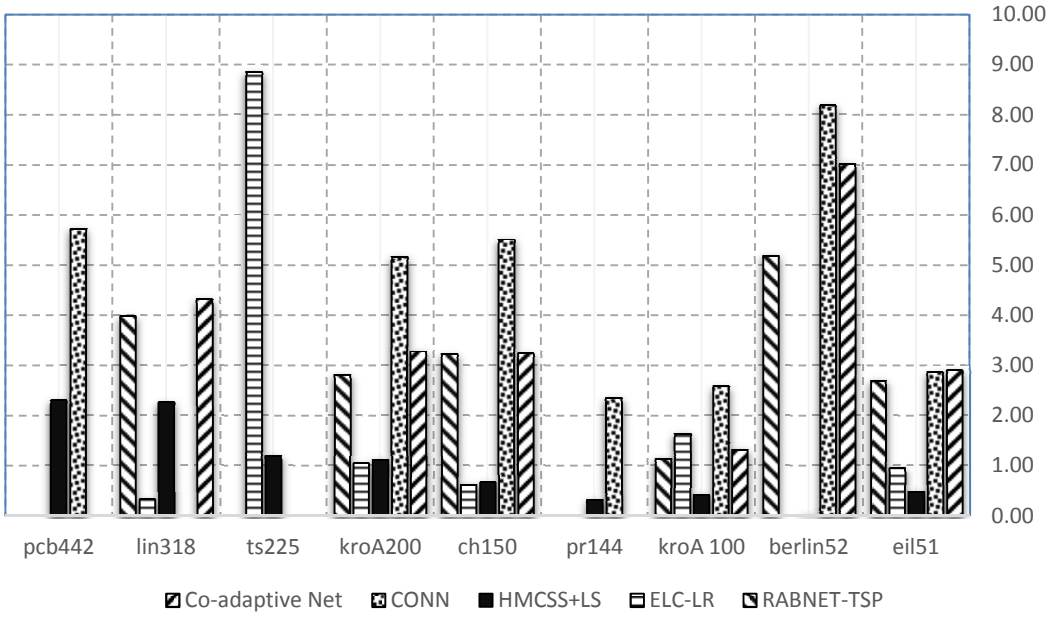

Fig. 2 Comparison of average error of HMCSS+LS with Co-adaptive Net [15], CONN [16], RABNET-TSP [5] and ELC-LR [17]

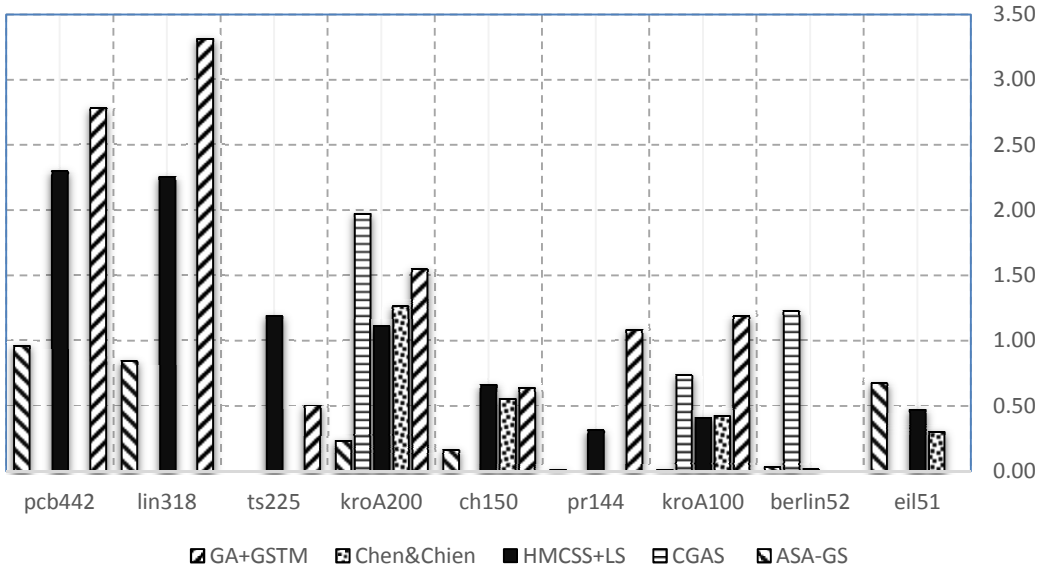

Fig. 3 Comparison of average error and best error of HMCSS+LS with GA+GSTM[6], Chen & Chien [18], ASA-GS [19], and CGAS.

5. Introduction of Single Row Facility Layout Problem

Single row facility layout problem (SRFLP) is an NP-hard problem, where the solution must be found a permutation of facilities on a line that minimize the weighted sum of the distances between all pairs of facilities**.**

A set F=(1, 2. . . n) of n > 2 facilities, the length lj of each facility j∈F, and weights $c_{i\ j}$ for each pair (i, j) of facilities are given. (i, j ∈ F, i≠j)

Objective is to find a permutation P = (p_1,p_2, . . . ,p_n) of facilities in F to minimizes the cost of the permutation:

$$z(p) = \sum_{1 \le i \le j \le n_{p_i p_j}} c_{i,j}.d_{p_i p_j}$$

Where $d_{p_i p_j} = l_{p_i}/2 + \sum_{i<k<j} l_{p_k} + l_{p_j}/2$ is the distance between the centroids of facilities p_i and p_j.

For many years, researchers have tried to provide better ways to solve this problem, which can be considered as the first exact method to solve the SRFLP proposed in [21, 22]. After them, other methods have been proposed, such as those of Refs. [23-24].

6. Hybrid-Modified Charged System Search Algorithm for Solving SRFLP

Changes that have been considered in developing the HMCSS algorithm for solving TSP are as follows:

1. The results showed that using NN to solve SRFLP is not useful, so this part of the algorithm for solving SRFLP was removed and algorithm initials with positions are generate randomly for solutions stored in the CM.
2. Obviously, the objective function of this algorithm changes.
3. In the local search method, the number of repeating process is sets to 3.

7. Experimental Results for SRFLP

In order to show the efficiency of the HMCSS+LS for solving SRFLP, results are compared to those of the benchmark problems; each problem being run for 100 trials. Results are presented in Table 4 and Table 5; algorithms parameters are chosen such that the number of CP's become equal to 24, radius of CP's is one, step for reallocate the CP's is equal to 25 iteration and HS parameters, CMCR, PAR, k_a and k_v are shown in Table 4.

Table 4 The parameters of HMCSS+LS

CMCR	0.99
PAR	0.1
k_a	0.5
k_v	0.5

Table 5 Comparison of the best cost of HMCSS+LS with S & E [26], DA&F [27], H & R [28] and K & G [24]

Instance	Size	S&E (2010)	DA&F (2011)	H&R (2011)	K&G (2013)	HMCSS
Anjos-60-05	60	318805.0	318805.0	318805.0	318805.0	**318805.0**
Anjos-70-05	70	4218230.0	4218017.5	4218002.5	4218002.5	**4218002.5**
Anjos-75-05	75	1791408.0	1791408.0	1791469.0	1791408.0	1791410.0
Anjos-80-05	80	1589061.0	1588901.0	1590847.0	1588885.0	**1588885.0**

Table 6 Comparison of the best cost of HMCSS+LS, A & L [23] and K & G [24]

Instance	Size	A&L (2012)	K&G (2013)	HMCSS
sko-64-05	64	501922.5	501922.5	**501922.5**
sko-72-05	72	428305.5	428248.5	**428230.5**
sko-81-05	81	1303756.0	1302833	**1302733**
sko-100-05	100	1034922.5	1033338.5	1033422.5

8. Conclusions

As was seen from the results of the previous sections, the hybrid-modified CSS algorithm in many cases, in both TSP and SRFLP, reached to satisfactory solutions compared to other algorithms. However, the CSS algorithm is very sensitive to the initial parameters such as particle radius, harmony search parameters and coefficients k_a and k_v. Thus, with further change of these parameters, one expects to obtain better results from this algorithm.

References

[1] Flood MM. The traveling-salesman problem, Operations Research, 1956, Vol. 4, pp. 61-75.

[2] Bhattacharyya M, Bandyopadhyay AK. Comparative study of some solution methods for traveling salesman problem using genetic algorithms, Cybernetics and Systems, 2008, Vol. 40, pp. 1-24.

[3] Reinelt G. The Traveling Salesman: Computational solutions for TSP Applications, Springer-Verlag, 1994.

[4] Shi XH, Liang YC, Lee HP, Lu C, Wang Q. Particle swarm optimization-based algorithms for TSP and generalized TSP, Information Processing Letters, 2007, Vol. 103, pp. 169-176.

[5] Masutti TA, Castro de LN. A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem, Information Sciences, 2009, Vol. 179, pp. 1454-1468.

[6] Albayrak M, Allahverdi N. Development a new mutation operator to solve the Traveling Salesman Problem by aid of Genetic Algorithms, Expert Systems with Applications, 2011, Vol. 38, pp. 1313-1320.

[7] Dong G, Guo WW, Tickle K. Solving the traveling salesman problem using cooperative genetic ant systems, Expert Systems with Applications, 2012, Vol. 39, pp. 5006-5011.

[8] Tsai C-F, Tsai C-W, Tseng C-C. A new hybrid heuristic approach for solving large traveling salesman problem, Information Sciences, 2004, Vol. 166, pp. 67-81.

[9] Kaveh A, Talatahari S. A novel heuristic optimization method: charged system search, Acta Mechanica, 2010, Vol. 213, pp. 267-289.

[10] Kaveh A, Farahani M, Shojaei N. Optimal design of barrel vaults using charged search system, International Journal of Civil Engineering, IUST, 2012, Vol. 10, pp. 301-308.
[11] Kaveh A, Nikaeen M. Optimum design of irregular grillage systems using CSS and ECSS algorithms with different boundary conditions, International Journal of Civil Engineering, IUST, 2013, Vol. 11, pp. 143-153.
[12] Kaveh A, Talatahari S. Charged system search for optimal design of frame structures, Applied Soft Computing, 2012, Vol. 12, pp. 382-393.
[13] Kaveh A. Advances in Metaheuristic Algorithms for Optimal Design of Structures, Springer International Publishing, Wien, 2014.
[14] Kaveh A, Talatahari S. An enhanced charged system search for configuration optimization using the concept of fields of forces, Structural and Multidisciplinary Optimization, 2011, Vol. 43, pp. 339-351.
[15] Cochrane E, Beasley J. The co-adaptive neural network approach to the Euclidean travelling salesman problem, Neural Networks, 2003, Vol. 16, pp. 1499-1525.
[16] Saadatmand-Tarzjan M, Khademi M, Moghaddam H. A novel constructive-optimizer neural network for the traveling salesman problem, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 2007, Vol. 37, pp. 754-770.
[17] Zamani R, Lau SK. Embedding learning capability in Lagrangean relaxation: An application to the travelling salesman problem, European Journal of Operational Research, 2010, Vol. 201, pp. 82-88.
[18] Chen S-M, Chien C-Y. Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, Expert Systems with Applications, 2011, Vol. 38, pp. 14439-14450.
[19] Geng X, Chen Z, Yang W, Shi D, Zhao K. Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search, Applied Soft Computing, 2011, Vol. 11, pp. 3680-3689.
[20] http://comopt.ifi.uni-heidelberg.de/software/tsplib95/.
[21] Simmons DM. One-dimensional space allocation: an ordering algorithm, Operations Research, 1969, Vol. 17, pp. 812-826.
[22] Love RF, Wong JY. On solving a one-dimensional space allocation problem with integer programming, Infor, 1976, Vol. 14, pp. 139-144.
[23] Amaral AR, Letchford AN. A polyhedral approach to the single row facility layout problem, Mathematical Programming, 2013, Vol. 141, pp. 453-477.
[24] Kothari R, Ghosh D. Tabu search for the single row facility layout problem using exhaustive 2-opt and insertion neighborhoods, European Journal of Operational Research, 2012, Vol. 224, pp. 93-100.
[25] Hungerländer P, Rendl F. Semidefinite relaxations of ordering problems, Mathematical Programming, 2013, Vol. 140, pp. 77-97.
[26] Samarghandi H, Eshghi K. An efficient tabu algorithm for the single row facility layout problem, European Journal of Operational Research, 2010, Vol. 205, pp. 98-105.
[27] Datta D, Amaral AR, Figueira JR. Single row facility layout problem using a permutation-based genetic algorithm, European Journal of Operational Research, 2011, Vol. 213, pp. 388-394.
[28] Hungerländer P, Rendl F. A computational study and survey of methods for the single-row facility layout problem, Computational Optimization and Applications, 2013, Vol. 56, pp. 1-20.