INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2020; 10(1):35-51

META-HEURISTIC ALGORITHMS FOR MINIMIZING THE NUMBER OF CROSSING OF COMPLETE GRAPHS AND COMPLETE BIPARTITE GRAPHS

A. Kaveh[*,†] and K. Biabani Hamedani
School of Civil Engineering, Iran University of Science and Technology, Tehran, Iran

ABSTRACT

The minimum crossing number problem is among the oldest and most fundamental problems arising in the area of automatic graph drawing. In this paper, eight population-based meta-heuristic algorithms are utilized to tackle the minimum crossing number problem for two special types of graphs, namely complete graphs and complete bipartite graphs. A 2-page book drawing representation is employed for embedding graphs in the plane. The algorithms consist of Artificial Bee Colony algorithm, Big Bang-Big Crunch algorithm, Teaching-Learning-Based Optimization algorithm, Cuckoo Search algorithm, Charged System Search algorithm, Tug of War Optimization algorithm, Water Evaporation Optimization algorithm, and Vibrating Particles System algorithm. The performance of the utilized algorithms is investigated through various examples including six complete graphs and eight complete bipartite graphs. Convergence histories of the algorithms are provided to better understanding of their performance. In addition, optimum results at different stages of the optimization process are extracted to enable to compare the meta-heuristics algorithms.

Keywords: crossing number; meta-heuristic algorithms; optimization; 2-page book drawing; complete graph; complete bipartite graph.

Received: 10 September 2019; Accepted: 25 November 2019

1. INTRODUCTION

Given a drawing S^p of graph S, the intersections which occur in the interiors of members are the crossings. The crossing number of a graph is the minimum number of crossings over all possible drawings of S. Minimum crossing number problem is a classic and very important problem in graph drawing. This problem has important applications such as determining the

[*] Corresponding author: School of Civil Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran
[†]E-mail address: alikaveh@iust.ac.ir (A. Kaveh)

statical indeterminacy of skeletal structures, design of printed circuit board layout, VLSI circuit routing, automated graph drawing. The objective of the minimum crossing number problem is to embed the members of a graph so that the total number of crossings is minimized. For a general graph, there is no known formula by which the crossing number can be calculated. There is also no algorithm by means of which an optimal drawing can be obtained. In fact it is proven that the minimum crossing number problem is NP-complete [1]. Therefore, an approach for estimating crossing numbers, and the exact value of crossing number should be restricted to special graphs like complete graphs, bipartite graphs and cubic graphs. For these graphs only upper and lower bounds for crossing numbers are usually conjectured, but not proved to be exact. Graph theoretical studies for number of crossing have been performed by Kaveh [2-4], and Kaveh and Rahami [5].

Meta-heuristic algorithm can be applied to the minimum crossing number problem. Meta-heuristics try to combine randomization and rule-based theories which are almost always taken from natural phenomena such as evolution, characteristics of biological systems, social systems, swarm intelligence, and governing laws in different phenomena like basic physical laws [6]. Meta-heuristic algorithms are easy to implement and have found many applications in different areas of applied mathematics, engineering, medicine, economics and other sciences [7,8]. Many researchers have applied meta-heuristic algorithms for the minimum crossing number problem. Makinen and Sieranta [9] utilized genetic algorithms for drawing bipartite graphs. Valls et al. [10] applied a tabu thresholding algorithm to arc crossing minimization in bipartite graphs. Shahrokhi et al. [11] proposed two polynomial time algorithms to generate near optimal drawing of a graph on book. Cimikowski and Shope [12] employed a neural network algorithm for a graph layout problem. Laguna et al. [13] performed arc crossing minimization in hierarchical design with tabu search. Utech et al. [14] and Tettamanzi [15] utilized evolutionary algorithms for drawing graphs. Marti [16] employed a greedy randomized adaptive search procedure (GRASP) for minimum crossing number in graphs. Wang and Okazaki [17] proposed an improved Hopfield neural network for solving the minimum crossing number problem. Kaveh and Ilchi Ghazaan [18] applied particle swarm optimization, improved ray optimization, colliding bodies optimization, and enhanced colliding bodies optimization to the minimum crossing number problem.

In this research, we try to solve the minimum crossing number problem for complete and complete bipartite graphs utilizing eight meta-heuristic algorithms. In other words, the aim of optimization is to find drawings with the least possible crossings. A 2-page book drawing is used for drawing the graphs. The algorithms consist of Artificial Bee Colony, Big Bang-Big Crunch, Teaching-Learning-Based Optimization, Cuckoo Search, Charged System Search, Tug of War Optimization, Water Evaporation Optimization, and Vibrating Particles System. The codes for these algorithms are those of Kaveh and Bakhshpoori [8]. The objective of optimization is to minimize the number of crossing members. Various examples are provided to demonstrate the effectiveness of the meta-heuristic algorithms and to compare their performance.

2. MATERIALS AND METHODS

2.1 Meta-heuristic algorithms

Eight meta-heuristic algorithms are utilized to minimize crossing number of graphs. These algorithms are as follows: 1) Artificial Bee Colony (ABC) algorithm, 2) Big Bang-Big Crunch (BB-BC) algorithm, 3) Teaching-Learning-Based Optimization (TLBO) algorithm, 4) Cuckoo Search (CS) algorithm, 5) Charged System Search (CSS) algorithm, 6) Tug of War Optimization (TWO) algorithm, 7) Water Evaporation Optimization (WEO) algorithm, and 8) Vibrating Particles System (VPS) algorithm. Kaveh and Bakhshpoori [8] coded these algorithms and performed some experimental evaluations to assess the performance of the algorithms in both aspects of convergence rate and accuracy. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithms. The algorithms are introduced briefly in the following sections.

2.1.1 Artificial bee colony algorithm (ABC)

Swarm intelligence and group behavior of honey bees is the basic inspiration of some meta-heuristics. The first one is the Artificial Bee Colony (ABC) algorithm which was introduced by Karaboga [19] based on the foraging behavior of honey bees. In ABC algorithm each candidate solution is represented by a food source, and its nectar quality represents the objective function of that solution. These food sources are modified by honey bees in a repetitive process manner with the aim of reaching food sources with better nectar. In ABC honey bees are categorized into three types: employed or recruited, onlooker, and scout bees with different tasks in the colony. Bees perform modification with different strategies according to their task. Employed bees try to modify the food sources and share their information with onlooker bees. Onlooker bees select a food source based on the information from employed bees and attempt to modify it. Scout bees perform merely random search in the vicinity of the hive. Hence ABC algorithm searches in three different sequential phases in each iteration.

2.1.2 Big bang-big crunch algorithm (BB-BC)

The Big Bang-Big Crunch (BB-BC) algorithm was developed by Erol and Eksin [20]. BB-BC is taken from the prevailing evolutionary theory for the origin of universe: the Big Bang Theory. According to this theory, in the Big Bang phase, particles are drawn toward irregularity by losing energy, while in the Big Crunch phase, they converged toward a specific direction. Like other population-based meta-heuristics, BB-BC starts with a set of random initial candidate solutions, as the initial Big Bang. After each Big Bang phase, a Big Crunch phase should take place to determine a convergence operator by which particles will be drawn into an orderly fashion in the subsequent Big Bang phase. The convergence operator can be the weighted average of the positions of the candidate solutions or the position of the best candidate solution. These two contraction (Big Crunch) and dispersing (Big Bang) phases are repeated in the cyclic body of the algorithm in succession to satisfy a stopping criteria with the aim of steering the particles toward the global optimum.

2.1.3 Teaching-learning-based optimization algorithm (TLBO)

Teaching-learning-based optimization (TLBO) algorithm was developed by Rao et al. [21] based on the classical school learning process. TLBO consists of two stages: effect of a teacher on learners and the influence of learners on each other. TLBO is initialized with a population of random solutions, named students or learners. The smartest student with the best objective function is assigned as the teacher in each iteration. Students are updated iteratively to search the optimum within two phases: based on the knowledge transfer from the teacher (teacher phase) and from interaction with other students (learner phase). In TLBO the performance of the class in learning or the performance of teacher in teaching is considered as a normal distribution of marks obtained by the students. TLBO improves other students in the teacher phase by using the difference between the teacher's knowledge and the average knowledge of all the students. The knowledge of each student is obtained based on the position taken place by that student in the search space. In a class, students also improve themselves via interacting with each other after the teaching is completed by the teacher. In the learner phase, TLBO improves each student by the knowledge interaction between that student and another randomly selected one.

2.1.4 Cuckoo search algorithm (CS)

Cuckoo Search (CS) algorithm was developed by Yang and Deb [22] as an efficient population-based meta-heuristic inspired by the behavior of some cuckoo species. Cuckoos are fascinating birds because of their aggressive reproduction strategy. These species lay their eggs in the nests of other host birds. The host takes care of the eggs presuming that the eggs are its own. However, some of host birds are able to combat with this parasites behavior of cuckoos and throw out the discovered alien eggs or build their new nests in new locations. All the nests or eggs whether they belong to the cuckoos or host birds represent the candidate solutions in the search space. Cuckoos and host birds try to breed their own generation. In the cyclic body of the algorithm, two sequential search phases are performed by cuckoos and host birds. Firstly, cuckoos produce the eggs. In this phase eggs are produced by guiding the current solutions toward the best known solution. Then these new eggs are intruded to the nests of host birds based on the replacement strategy. After cuckoo breeding, it turns to the host birds. If a cuckoo's egg is very similar to a host's egg, then this cuckoo's egg is less likely to be discovered. In this phase host birds discover a fraction of alien eggs and update them by adding them a random permutation-based step size. Based on the replacement strategy, host bird replaces the produced egg with the current one. These two search phases are repeated in the cyclic body of the algorithm until reaching to a stopping criteria.

2.1.5 Charged system search algorithm (CSS)

Charged System Search (CSS) algorithm was developed by Kaveh and Talatahari [23] as an efficient population-based meta-heuristic using some principles from physics and mechanics. CSS utilizes the governing Coulomb laws from electrostatics and the Newtonian laws of mechanics. In this algorithm each agent is a charged particle with a predetermined radius. The charge of magnitude of particles is considered based on their quality. Each particle creates an electric field, which exerts a force on other electrically charged objects.

Therefore, charged particles can affect each other based on their fitness values and their separation distance. The quantity of the resultant force is determined by using the electrostatics laws, and the quality of the movement is determined using Newtonian mechanics laws. In each iteration, transitions of particles can be induced by electric fields leading to particle-particle electrostatic interactions with the aim of attracting or repelling the particles toward the optimum position.

2.1.6 Tug of war optimization algorithm (TWO)

Inspired by the game tug of war, Kaveh and Zolghadr [24] developed a novel population-based meta-heuristic algorithm denoted as Tug of War Optimization (TWO) algorithm. TWO considers each candidate solution as a team participating in a series of rope pulling competitions. The teams exert pulling forces on each other based on the quality of the solutions they represent. The competing teams move to their new positions according to Newtonian laws of mechanics. TWO starts from a set of randomly generated initial candidate solutions. Each candidate solution is considered as a team, and all population form a league. The weight of teams is determined based on the quality of the corresponding solutions, and the amount of pulling force that a team can exert on the rope is assumed to be proportional to its weight. Naturally, the opposing team will have to maintain at least the same amount of force in order to sustain its grip on the rope. The lighter team accelerates toward the heavier team, and this forms the convergence operator of TWO. In each iteration of the algorithm, the league is updated by a series of team-team rope pulling competitions with the aim of attracting teams toward the optimum position.

2.1.7 Water evaporation optimization algorithm (WEO)

Inspired by evaporation of a tiny amount of water molecules on the solid surface with different wettability, Kaveh and Bakhshpoori [25] developed a novel meta-heuristic called Water Evaporation Optimization (WEO). This algorithm considers water molecules as algorithm individuals. Solid surface or substrate with variable wettability is reflected as the search space. Decreasing the surface wettability (substrate changed from hydrophilicity to hydrophobicity) reforms the water aggregation from a monolayer to a sessile droplet. Such a behavior is consistent with how the layout of individuals changes to each other as the algorithm progresses. Decreasing wettability of the surface can represent the decrease of objective function for a minimizing optimization problem. Evaporation flux rate of the water molecules is considered as the most appropriate measure for updating the individuals which its pattern of change is in good agreement with the local and global search ability of the algorithm and can help WEO to have significantly well-converged behavior and simple algorithmic structure.

2.1.8 Vibrating particles system algorithm (VPS)

Vibrating Particles System (VPS) algorithm is a new meta-heuristic search algorithm developed by Kaveh and Ilchi Ghazaan [26]. VPS is motivated based on the free vibration of single degree of freedom systems with viscous damping. Like other population-based meta-heuristics, VPS starts from a random set of initial solutions and considers them as the free vibrated single degree of freedom systems with viscous damping. Considering underdamped conditions, each free vibrated system or vibrating particle will oscillate and return to its equilibrium position. By utilizing a combination of randomness and exploitation of the obtained results, VPS improves the quality of the particles iteratively by oscillating them toward the equilibrium position, as the optimization process proceeds. Consider the equilibrium position of each particle, consisting of three parts, the best position achieved so far across the entire population (HP), a good particle (GP), and a bad particle (BP). In this way the essence of VPS stands on three essential concepts, self-adaptation (particle moves toward HB), cooperation (the GP and BP, which are selected from particles themselves, can influence the new position of particles), and competition (the influence of GP will be more than that of BP).

2.2 Definition of the optimization problem

A graph S consists of a set of elements called nodes and a set of elements called members together with a relation of incidence which associates each member with a pair of nodes, called its ends. Two nodes of a graph are called adjacent if these nodes are the end nodes of a member. A member is called incident with a node if it is an end node of that member. Two members are called incident if they have a common end node. A complete graph is a graph in which every two distinct nodes are connected by exactly one member, Fig. 1. A complete graph with N nodes is denoted by K_N.

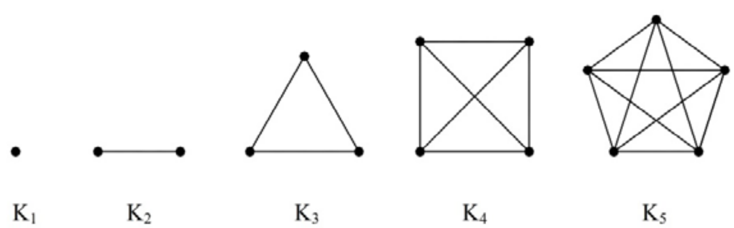

Figure 1. Some complete graphs

A graph is called bipartite, if the corresponding node set can be split into two sets N_1 and N_2 in such a way that each member of S joins a node of N_1 to a node of N_2. A complete bipartite graph is a bipartite graph in which each node N_1 is joined to each node of N_2 by exactly one member. If the number of nodes in N_1 and N_2 are denoted by r and s, respectively, then a complete bipartite graph is denoted by $K_{r,s}$. Examples of bipartite and complete bipartite graphs are shown in Fig. 2.

(a) A bipartite graph (b) A complete bipartite graph ($K_{3,4}$)

Figure 2. Two bipartite graphs

A drawing S^p of a graph S in the plane is a mapping of the nodes of S to distinct points of S^p, and the members of S to open arcs of S^p such that [3]:
(i) the image of no member contains that of any node;
(ii) the image of a member (n_i, n_j) joins the points corresponding to n_i and n_j.

A drawing is called admissible if the members are such that:
(i) no two arcs with a common end point meet;
(ii) no two arcs meet in more than one point;
(iii) no three arcs meet in a common point.

A point of intersection of two members in a drawing is called a crossing, and the crossing number $c(S^p)$ of a graph S is the number of crossings in any admissible drawing of S in the plane. An optimal drawing in a given surface is one which exhibits the least possible crossings. It is proven that for given graph S and an integer k, the question $c(S^p) \leq k$ is NP-complete [1]. There are conjectures for the crossing number of both the complete and complete bipartite graphs [27]:

$$c(K_N) = \frac{1}{4}\left\lfloor\frac{N}{2}\right\rfloor\left\lfloor\frac{N-1}{2}\right\rfloor\left\lfloor\frac{N-2}{2}\right\rfloor\left\lfloor\frac{N-3}{2}\right\rfloor \tag{1}$$

$$c(K_{r,s}) = \frac{1}{4}\left\lfloor\frac{r}{2}\right\rfloor\left\lfloor\frac{r-1}{2}\right\rfloor\left\lfloor\frac{s}{2}\right\rfloor\left\lfloor\frac{s-1}{2}\right\rfloor \tag{2}$$

where $c(K_N)$ is the crossing number of the complete graph K_N. It should be noted that the equation (1) has been confirmed only for $N \leq 12$. The smallest unsolved case is K_{13} with conjectured crossing number 225. Furthermore, $c(K_{r,s})$ is the crossing number of the complete bipartite graph $K_{r,s}$. The equation (2) is known to be true for $r \leq 6$ and all s, and also for $r = 7$ when $s \leq 10$ [18].

Towards the end of the century, substantially different drawings of K_N were found, such as cylindrical drawing, 2-page book drawing, monotone drawing, shellable drawing, etc. To this date, no drawing of K_N with fewer than $c(K_N)$ crossings is known [28]. In the 2-page book drawing representation which is used here, all the nodes of the graph are on a line and each member is embedded in either the upper page or the lower page defined by the line [28]. A 2-page book drawing of K_8 is shown in Fig. 3.

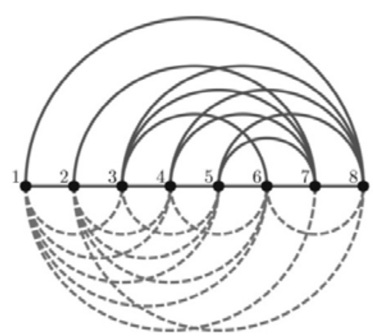

Figure 3. A 2-page book drawing of K_8

Any pair of members ij and kl cross in a drawing if $i < k < j < l$ and both lie in the same page. The state $y_{ij} = 1$ indicates that the member between nodes i and j (the member ij) is embedded in the upper page, and the state $y_{ij} = 0$ indicates that the the member ij is embedded in the lower page. The number of members in a given graph determines the number of design variables of the optimization problem. The linear crossing number problem can be mathematically stated as finding the minimum of the following objective function [17]:

$$crossing\ number = \frac{1}{2}\sum_{ij}\sum_{kl}\left(g_{ij}.\, g_{kl}.\, d_{ijkl}.\, y_{ij}.\, y_{kl}\right) + \frac{1}{2}\sum_{ij}\sum_{kl}\left(g_{ij}.\, g_{kl}.\, d_{ijkl}.\, (1 - y_{ij}).\, (1 - y_{kl})\right) \quad (3)$$

where d_{ijkl} is crossing condition and can be stated as follows:

$$d_{ijkl} = \begin{cases} 1, & if \quad i < j < k < l \quad or \quad k < i < l < j \\ 0, & otherwise \end{cases} \quad (4)$$

g_{ij} indicates whether the member ij exist.

$$g_{ij} = \begin{cases} 1, & if\ member\ ij\ exist \\ 0, & otherwise \end{cases} \quad (5)$$

3. RESULTS AND DISCUSSION

Six complete graphs ($K_8, K_9, K_{10}, K_{11}, K_{12}, K_{13}$) and eight complete bipartite graphs ($K_{3,10}, K_{3,15}, K_{4,5}, K_{4,10}, K_{4,15}, K_{5,5}, K_{5,10}, K_{5,15}$) are studied to illustrate the efficiency of the algorithms. Optimization results of all algorithms are presented in Tables 1 to 6. Tables 1 to 3 shows the optimum results for complete graphs, and the results in Tables 4 to 6 are those of the complete bipartite graphs. Optimum solutions at four different stages of the optimization process are provided in Tables 3 and 6 for complete and complete bipartite graphs, respectively. These two tables enable us to compare the performance of meta-heuristics algorithms. The average and standard deviation of results are shown in the Tables 2 and 5 for complete and complete bipartite graphs, respectively. Convergence histories of all graphs are depicted in Figs. 7 to 20. The maximum number of objective function evaluations is different for each graph. For instance, this parameter is set to 500 for K_8 and K_9, while this is set to 2000 for K_{12} and K_{13}. Embeding of the complete graph K_8 obtained by TLBO is shown in Fig. 4. Fig. 5 shows the embeding of the complete bipartite graph $K_{3,10}$ obtained by CSS. In addition, embeding of the complete graph $K_{4,10}$ obtained by WEO is shown in Fig. 6. The optimization results show that the algorithms have relatively close performances for small complete and complete bipartite graphs which demonstrates the high performance of the algorithms. A careful examination of Figs. 11, 12, 17, and 20 reveals that TWO, CS, CSS, and WEO have better performance for larger complete and complete bipartite graphs (K_{12}, K_{13}, $K_{4,15}$, $K_{5,15}$) in both aspects of convergence rate and accuracy compared to other used algorithms. These algorithms also have better results in terms of the average and standard deviation of results. As it can be seen from Tables 1, TWO, CS, CSS, and WEO converge to the global optimum (optimal drawing) for all complete graphs. On the contrary, an examination of Table 4 reveals that for some of the complete bipartite graphs (e.g. $K_{4,5}$, $K_{5,5}$, $K_{5,10}$), TWO, CS, CSS, and WEO converge to identical solutions close to the optimal drawing, but not equal. The reason is that these drawings are the best possible ones which can be found in a 2-page book drawing. In other words, by mean of the 2-page book drawing, it is impossible to find an embedding with fewer crossings for the above mentioned graphs.

Table 1: Optimal results obtained for the complete graphs

Graph	Minimum	ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
K_8	18	18	18	18	18	18	18	18	18
K_9	36	36	36	36	36	36	36	36	36
K_{10}	60	60	60	60	60	60	60	60	60
K_{11}	100	100	100	100	100	100	100	100	100
K_{12}	150	150	150	153	150	150	150	150	155
K_{13}	-	227	225	229	225	225	225	225	231

Table 2: Statistical results of the complete graphs

Graph		ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
	Average	19.42	20.37	18.57	19.28	18.44	19.21	18.56	19.66
K_8	Std. dev.	2.20	2.22	1.17	1.21	1.22	0.88	0.81	1.87
	No. of analyses	500	500	500	500	500	500	500	500
	Average	37.24	37.4	37.04	37.2	37.24	36.48	37.44	37.81
K_9	Std. dev.	3.05	3.73	3.16	1.94	2.70	1.69	3.06	4.88
	No. of analyses	500	500	500	500	500	500	500	500
	Average	62.85	63.46	64.86	63.19	65.06	63.05	63.53	67.69
K_{10}	Std. dev.	5.53	6.79	7.76	5.28	6.90	5.27	6.88	6.96
	No. of analyses	1000	1000	1000	1000	1000	1000	1000	1000
	Average	102.44	104.81	103.14	103.18	107.40	104.47	102.92	107.51
K_{11}	Std. dev.	5.72	7.43	7.37	9.08	11.05	10.64	7.71	8.00
	No. of analyses	1500	1500	1500	1500	1500	1500	1500	1500
	Average	159.67	158.84	160.62	157.13	156.49	156.25	157.40	177.44
K_{12}	Std. dev.	12.61	15.94	15.71	16.03	15.01	16.73	16.66	16.11
	No. of analyses	2000	2000	2000	2000	2000	2000	2000	2000
	Average	237.92	236.33	251.43	236.18	234.25	230.45	234.66	248.34
K_{13}	Std. dev.	18.91	23.52	17.83	22.63	22.99	15.92	22.23	24.69
	No. of analyses	2000	2000	2000	2000	2000	2000	2000	2000

Table 3: Optimum results at different stages of optimization (complete graphs)

Graph	No. of analyses	ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
K_8	125	19	19	18	20	18	20	19	21
	250	19	18	18	19	18	20	18	19
	375	18	18	18	18	18	18	18	18
	500	18	18	18	18	18	18	18	18
K_9	125	36	36	36	40	38	36	38	36
	250	36	36	36	36	36	36	36	36
	375	36	36	36	36	36	36	36	36
	500	36	36	36	36	36	36	36	36
K_{10}	250	63	63	63	63	64	63	66	68
	500	61	61	62	61	62	61	60	67
	750	60	60	61	60	61	60	60	63
	1000	60	60	60	60	60	60	60	60
K_{11}	375	102	100	100	102	108	102	104	116
	750	100	100	100	100	102	100	100	104
	1125	100	100	100	100	102	100	100	102
	1500	100	100	100	100	100	100	100	100
K_{12}	500	160	162	161	156	154	153	154	178
	1000	157	150	153	153	150	150	150	176
	1500	152	150	153	150	150	150	150	167
	2000	150	150	153	150	150	150	150	155
K_{13}	500	233	231	259	233	233	225	231	249
	1000	233	225	251	227	225	225	225	241
	1500	227	225	235	225	225	225	225	231
	2000	227	225	229	225	225	225	225	231

Table 4: Optimal results obtained for the complete bipartite graphs

Graph	Minimum	ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
$K_{3,10}$	20	20	20	20	20	20	20	20	20
$K_{3,15}$	49	49	49	49	49	49	49	49	49
$K_{4,5}$	8	10	10	10	10	10	10	10	10
$K_{4,10}$	40	54	54	54	54	54	54	54	54
$K_{4,15}$	98	130	130	132	130	130	130	130	133
$K_{5,5}$	16	20	20	20	20	20	20	20	20
$K_{5,10}$	80	100	100	100	100	100	100	100	100
$K_{5,15}$	196	252	244	262	244	244	244	244	252

Table 5: Optimum results at different stages of optimization (complete bipartite graphs)

Graph	No. of analyses	ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
$K_{3,10}$	125	23	22	23	31	20	31	28	29
	250	20	20	20	20	20	21	20	20
	375	20	20	20	20	20	20	20	20
	500	20	20	20	20	20	20	20	20
$K_{3,15}$	250	67	49	55	57	53	61	59	91
	500	51	49	49	49	49	49	49	85
	750	49	49	49	49	49	49	49	57
	1000	49	49	49	49	49	49	49	49
$K_{4,5}$	100	10	10	10	10	11	10	10	10
	200	10	10	10	10	10	10	10	10
	300	10	10	10	10	10	10	10	10
	400	10	10	10	10	10	10	10	10
$K_{4,10}$	200	57	54	71	60	54	60	54	75
	400	54	54	54	54	54	54	54	66
	600	54	54	54	54	54	54	54	57
	800	54	54	54	54	54	54	54	54
$K_{4,15}$	375	165	140	176	156	132	130	130	185
	750	148	130	156	133	130	130	130	174
	1125	130	130	156	130	130	130	130	148
	1500	130	130	132	130	130	130	130	133
$K_{5,5}$	100	20	20	24	24	26	24	20	20
	200	20	20	20	20	20	20	20	20
	300	20	20	20	20	20	20	20	20
	400	20	20	20	20	20	20	20	20
$K_{5,10}$	375	102	103	113	100	100	103	101	133
	750	101	100	100	100	100	100	100	100
	1125	100	100	100	100	100	100	100	100
	1500	100	100	100	100	100	100	100	100
$K_{5,15}$	500	336	262	370	348	252	262	270	384
	1000	278	256	342	302	244	244	244	312
	1500	262	244	294	256	244	244	244	280
	2000	252	244	262	244	244	244	244	252

Table 6: Statistical results of the complete bipartite graphs

Graph		ABC	BB-BC	TLBO	CS	CSS	TWO	WEO	VPS
	Average	23.50	26.66	23.10	26.77	22.33	25.1	25.70	24.68
$K_{3,10}$	Std. dev.	8.54	8.92	6.44	12.59	5.87	8.49	10.10	6.69
	No. of analyses	500	500	500	500	500	500	500	500
	Average	59.70	55.85	60.18	58.82	55.00	61.74	58.30	81.95
$K_{3,15}$	Std. dev.	16.98	19.28	20.05	21.24	15.14	22.85	16.37	24.34
	No. of analyses	1000	1000	1000	1000	1000	1000	1000	1000
	Average	10.65	11.58	11.22	11.53	10.40	10.67	11.32	10.64
$K_{4,5}$	Std. dev.	2.55	4.23	3.00	3.99	0.99	2.32	3.32	2.22
	No. of analyses	400	400	400	400	400	400	400	400
	Average	59.58	60.94	61.63	62.14	59.4	60.98	59.03	67.88
$K_{4,10}$	Std. dev.	14.51	15.27	11.51	17.22	13.32	16.85	14.51	15.36
	No. of analyses	800	800	800	800	800	800	800	800
	Average	156.88	151.43	170.70	150.92	141.29	149.98	144.48	176.66
$K_{4,15}$	Std. dev.	36.62	38.16	35.54	34.99	30.60	41.77	35.05	40.72
	No. of analyses	1500	1500	1500	1500	1500	1500	1500	1500
	Average	21.56	22.06	22.74	22.22	22.40	21.82	22.41	22.76
$K_{5,5}$	Std. dev.	5.26	5.26	4.84	4.51	3.48	3.53	5.45	7.01
	No. of analyses	400	400	400	400	400	400	400	400
	Average	108.17	109.58	110.83	106.71	109.09	108.32	109.62	114.52
$K_{5,10}$	Std. dev.	19.59	23.34	22.84	20.33	23.35	20.51	23.96	25.39
	No. of analyses	1500	1500	1500	1500	1500	1500	1500	1500
	Average	302.24	269.13	340.48	310.51	274.27	269.93	270.15	333.34
$K_{5,15}$	Std. dev.	61.40	47.14	49.61	69.55	63.20	51.69	54.43	57.64
	No. of analyses	2000	2000	2000	2000	2000	2000	2000	2000

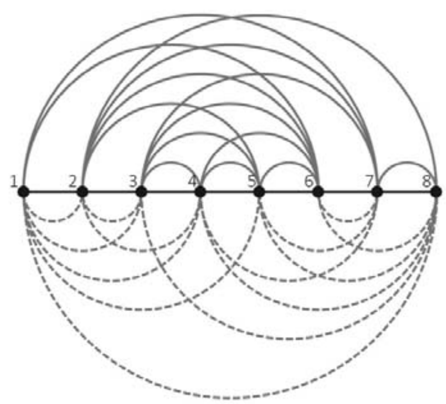

Figure 4. Embeding of the complete graph K_8 obtained by TLBO

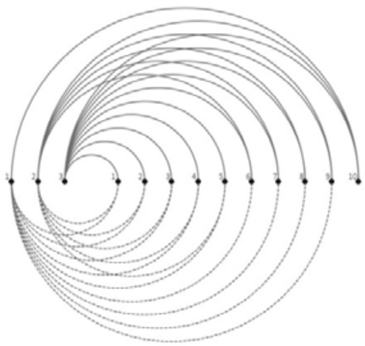

Figure 5. Embeding of the complete bipartite graph $K_{3,10}$ obtained by CSS

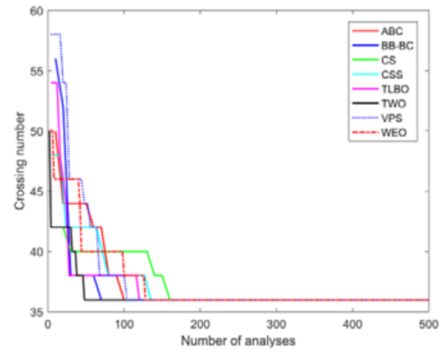

Figure 8. Convergence histories for K_9

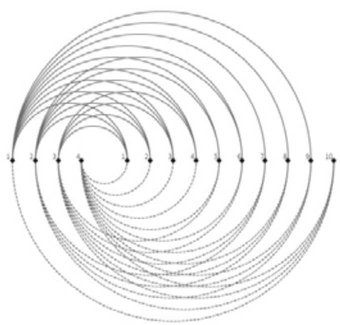

Figure 6. Embeding of the complete bipartite graph $K_{4,10}$ obtained by WEO

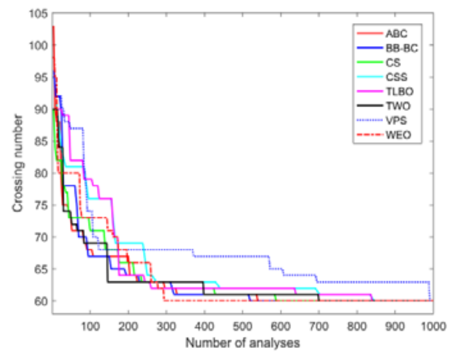

Figure 9. Convergence histories for K_{10}

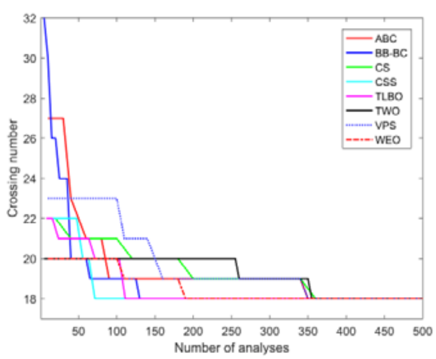

Figure 7. Convergence histories for K_8

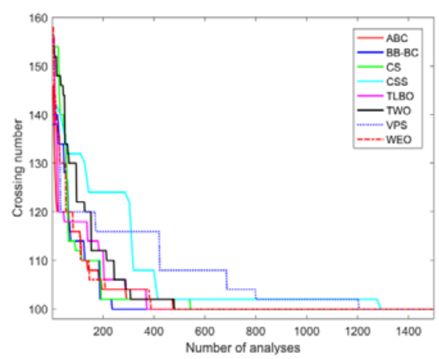

Figure 10. Convergence histories for K_{11}

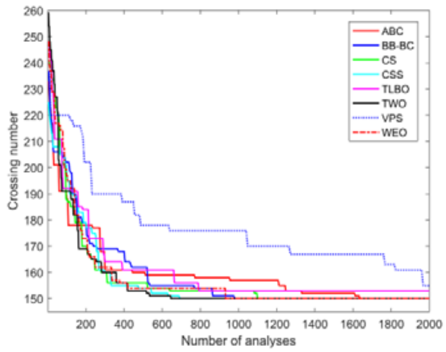

Figure 11. Convergence histories for K_{12}

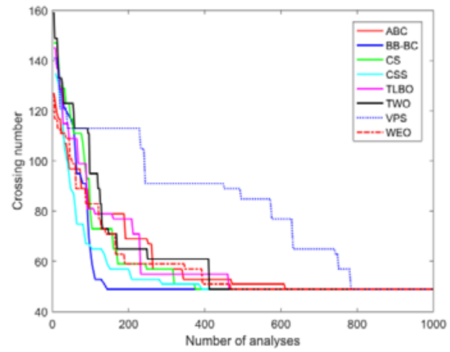

Figure 14. Convergence histories for $K_{3,15}$

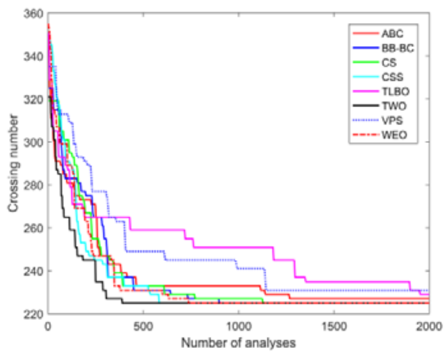

Figure 12. Convergence histories for K_{13}

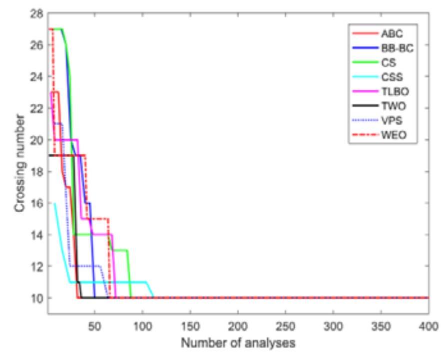

Figure 15. Convergence histories for $K_{4,5}$

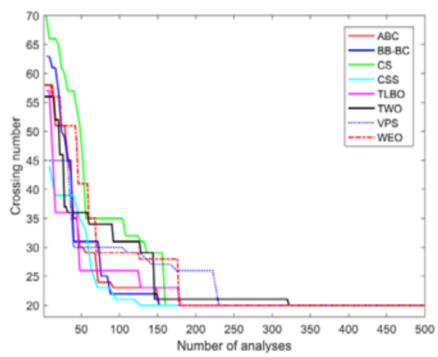

Figure 13. Convergence histories for $K_{3,10}$

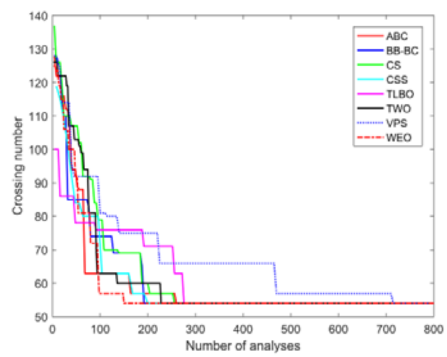

Figure 16. Convergence histories for $K_{4,10}$

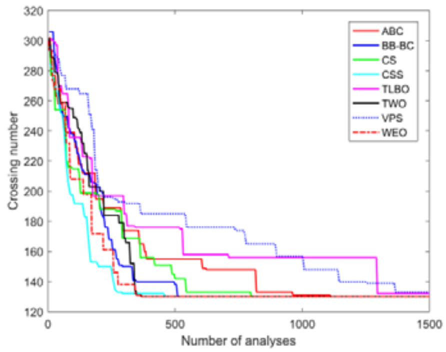

Figure 17. Convergence histories for $K_{4,15}$

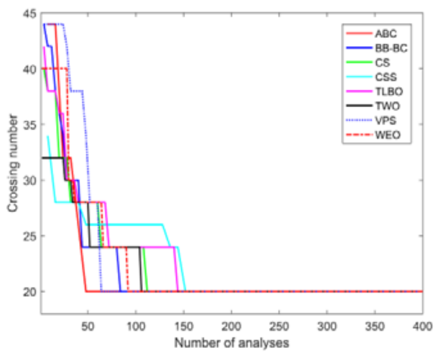

Figure 18. Convergence histories for $K_{5,5}$

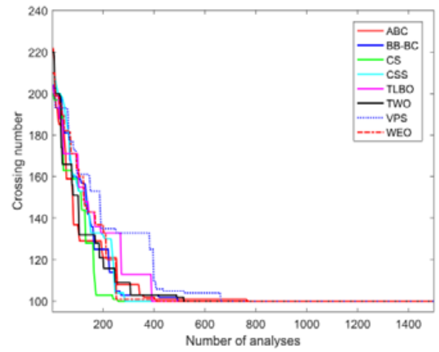

Figure 19. Convergence histories for $K_{5,10}$

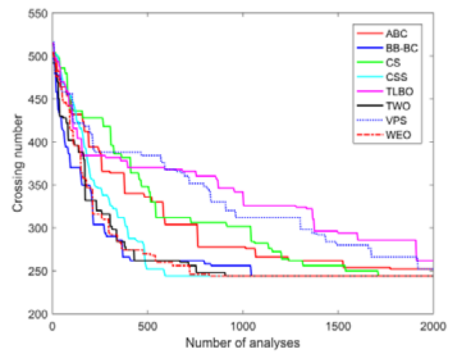

Figure 20. Convergence histories for $K_{5,15}$

4. CONCLUSION

In this paper, eight population-based meta-heuristic algorithms are employed for the minimum crossing number problem in complete graphs and complete bipartite graphs. The algorithms consist of Artificial Bee Colony, Big Bang-Big Crunch, Teaching-Learning-Based Optimization, Cuckoo Search, Charged System Search, Tug of War Optimization, Water Evaporation Optimization, and Vibrating Particles System. A 2-page book drawing representation is used for embedding the graphs. The objective of optimization is to minimize the crossing number of complete and complete bipartite graphs. All the utilized algorithms can find optimal or near optimal drawings rapidly and have an acceptable performance for the minimum crossing number problem. The results indicate superiority of the CS, CSS, TWO, and WEO algorithms in both aspects of convergence rate and accuracy compared to other employed algorithms. The convergence histories of the mentioned algorithms indicate that they have a close performance.

REFERENCES

1. Gary MR, Johnson DS. Crossing number is NP-complete, *SIAM J Alg Disc Meth* 1983; **4**(3): 312-16.
2. Kaveh A. Space structures and crossing number of their graphs, *Mech Struct Mach* 1993; **21**(2): 151-66.
3. Kaveh A. *Structural Mechanics: Graph and Matrix Methods*, Research Studies Press, 3rd edition, Baldock, Hertfordshire, England, 2004.
4. Kaveh A. Crossing number, genus and thickness of a graph for the flexibility analysis of structures, In *Trends in Application of Mathematics to Mechanics*, Edits: Schneider W, Troger H, Ziegler F. Longman Scientific & Technical, New York, 1991, pp. 333-338.
5. Kaveh A, Rahami H. An efficient algorithm for embedding non-planar graphs in planes, *J Math Model Algor* 2002; **1**(4): 257-68.
6. Kaveh A. *Advances in Metaheuristics Algorithms for Optimal Design of Structures*, Springer, 2nd edition, Cham, Switzerland, 2017.
7. Kaveh A. *Applications of Metaheuristic Optimization Algorithms in Civil Engineering*, Springer, 1st edition, Cham, Switzerland, 2017.
8. Kaveh A, Bakhshpoori T. *Metaheuristics: Outlines, MATLAB Codes and Examples*, Springer, 1st edition, Cham, Switzerland, 2019.
9. Makinen E, Sieranta M. Genetic algorithms for drawing bipartite graphs, *Int J Comput Math* 1994; **53**(3-4): 157-66.
10. Valls V, Marti R, Lino P. A tabu thresholding algorithm for arc crossing minimization in bipartite graphs, *Ann Oper Res* 1996; **63**(2): 233-51.
11. Shahrokhi F, Szekely LA, Sykoro O, Vrto I. The book crossing number of a graph, *J Graph Theory* 1996; **21**(4): 413-24.
12. Cimikowski R, Shope P. A neural network algorithm for a graph layout problem, *IEEE T Neural Networ* 1996; **7**(2): 341-5.
13. Laguna M, Marti R, Valls V. Arc crossing minimization in hierarchical design with tabu search, *Comput Oper Res* 1997; **24**(12): 1175-86.
14. Utech J, Branke J, Schmeck H, Eades P. An evolutionary algorithm for drawing directed graphs, *Proceedings of the International Conference on Imaging Science, Systems, and Technology*, Las Vegas, Nevada, USA, 1998, pp. 154-60.
15. Tettamanzi AGB. Drawing graphs with evolutionary algorithms, *Proceedings of 1998 Conference on Adaptive Computing in Design and Manufacture*, Plymouth, England, 1998, pp. 325-37.
16. Marti R. Arc crossing minimization in graphs with GRASP, *IIE Trans* 2001; **33**(10): 913-9.
17. Wang RL, Okazaki K. Artificial neural network for minimum crossing number problem, *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, Guangzhou, China, 2005, pp. 4201-4.
18. Kaveh A, Ilchi Gazaan M. Metaheuristic algorithms for minimum crossing number problem, *Int J Optim Civ Eng* 2015; **5**(1): 67-77.
19. Karaboga D. An idea based on honey bee swarm for numerical optimization, Technical Report, Department of Computer Engineering, Faculty of Engineering, Erciyes University, Erciyes, Turkey, 2005.
20. Erol OK, Eksin I. New optimization method: big bang-big crunch, *Adv Eng Softw* 2006; **37**(2): 106-11.

21. Rao RV, Savsani VJ, Vakharia DP. Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems, *Comput Aided Des* 2011; **43**(3): 303-15.
22. Yang XS, Deb S. Engineering optimisation by cuckoo search, *Int J Math Model Numer Optim* 2010; **1**(4): 330-43.
23. Kaveh A, Talatahari S. A novel heuristic optimization: charged system search algorithm, *Acta Mech* 2010; **213**(3-4): 267-89.
24. Kaveh A, Zolghadr A. A novel meta-heuristic algorithm: tug of war optimization, *Int J Optim Civ Eng* 2016; **6**(4): 469-92.
25. Kaveh A, Bakhshpoori T. Water evaporation optimization: a novel physically inspired optimization algorithm, *Comput Struct* 2016; **167**(20): 69-85.
26. Kaveh A, Ilchi Ghazaan M. A new meta-heuristic algorithm: vibrating particles system, *Sci Iran Trans A Civ Eng* 2017; **24**(2): 551-66.
27. Harary F. *Graph Theory*, Addison-Wesley, Reading, Massachusetts, USA, 1972.
28. Abrego BM, Aichholzer O, Fernandez-Merchant S, Ramos P, Salazar G. Shellable drawings and the cylindrical crossing number of K_n, *Discrete Comput Geom* 2014; **52**(4): 743-53.