INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2015; 5(1):67-77

METAHEURISTIC ALGORITHMS FOR MINIMUM CROSSING NUMBER PROBLEM

A. Kaveh*,† and M. Ilchi Ghazaan
Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran

ABSTRACT

This paper presents the application of metaheuristic methods to the minimum crossing number problem for the first time. These algorithms including particle swarm optimization, improved ray optimization, colliding bodies optimization and enhanced colliding bodies optimization. For each method, a pseudo code is provided. The crossing number problem is NP-hard and has important applications in engineering. The proposed algorithms are tested on six complete graphs and eight complete bipartite graphs and their results are compared with some existing methods.

Received: 12 October 2014; Accepted: 14 December 2014

KEY WORDS: graph layout; crossing number; particle swarm optimization; improved ray optimization; colliding bodies optimization; enhanced colliding bodies optimization.

1. INTRODUCTION

In recent years, various kinds of metaheuristic algorithms are proposed to solve instances of problems that are believed to be hard in general. These algorithms achieve this by reducing the effective size of the search and exploring that space efficiently. The metaheuristic algorithms are easy to implement and the basic idea behind these methods is usually natural phenomena. In this paper, particle swarm optimization (PSO) [1], improved ray optimization (IRO) [2], colliding bodies optimization (CBO) [3] and enhanced colliding bodies optimization (ECBO) [4] are applied to the minimum crossing number problem.

---

*Corresponding author: Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran
†E-mail address: alikaveh@iust.ac.ir (A. Kaveh)

The study of crossing numbers began during the Second World War [5]. The crossing number of a graph G is the smallest number of pairwise crossings of edges among all drawings of G in the plane. The crossing numbers for basic graphs such as the complete graphs and complete bipartite graphs is still unknown. This problem has important applications such as printed circuit board layout, VLSI circuit routing, automated graph drawing [6] and finding the degree of statical indeterminacy in structure, Kaveh [7,8]. In this paper, the problem involves placing the vertices of the graph along a horizontal “node line” in the plane and then adding edges as specified by the interconnection pattern. The objective of this problem is to embed the edges so that the total number of crossings is minimized [7]. This kind of problem is studied by Shahrokhi et al. [8], Cimikowski and Shope [9] and Wang and Okazaki [7].

The remaining sections of this paper are organized as follows. In Section 2, the mathematical formulation of the minimum crossing number problem is presented. Optimization algorithms are briefly described in Section 3. To show the efficiency and robustness of metaheuristic algorithms to the minimum crossing number problem, these are applied to some complete graphs and complete bipartite graphs in Section 4. Finally the paper is concluded in Section 5.

2. FORMULATION OF THE OPTIMIZATION PROBLEM

A *graph K* consists of a set of elements called *nodes*, a set of elements called *edges*, together with a relation of incidence which associates two distinct nodes with each edge, known as its *ends*. Two nodes of a graph are called *adjacent* if these nodes are the end nodes of an edge. An edge is called *incident* with a node if it is an end node of the edge. A graph is called a *complete graph* if all its r nodes are connected to each other, denoted by K_r. A graph is a complete bipartite graph if its nodes consist of two sets A and B with all nodes of A being connected to all nodes of B. A graph containing r nodes in A and s nodes in B, is denoted by $K_{r,s}$.

A *drawing* K^p of a graph K in the plane is a mapping of the nodes of K to distinct points of K^p, and the members of K to open arcs of K^p such that:

(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 *good* if the members are such that:

(iii) no two edges with a common end point meet;

(iv) no two edges meet in more than one point;

(v) 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* $cr(K^p)$ of a graph K is the minimum number of crossings in any good drawing of K in the plane.

The crossing number for a complete graphs K_r is calculated by [12]:

$$cr(K_r) = \frac{1}{4}\left[\frac{r}{2}\right]\left[\frac{r-1}{2}\right]\left[\frac{r-2}{2}\right]\left[\frac{r-3}{2}\right] \tag{1}$$

This formula holds only for $r \leq 12$.

The crossing number for a complete bipartite graph $K_{r,s}$ can be evaluated by [13]:

$$cr(K_{r,s}) = \left[\frac{r}{2}\right]\left[\frac{r-1}{2}\right]\left[\frac{s}{2}\right]\left[\frac{s-1}{2}\right] \tag{2}$$

This is known to be true for $r \leq 6$ and all s, and also for $r = 7$ when $s \leq 10$.

In the 2-page drawing representation used here, each edge is embedded in either the upper page or the lower page [9]. Embedding of the complete graph K_6 is shown in Fig. 1.

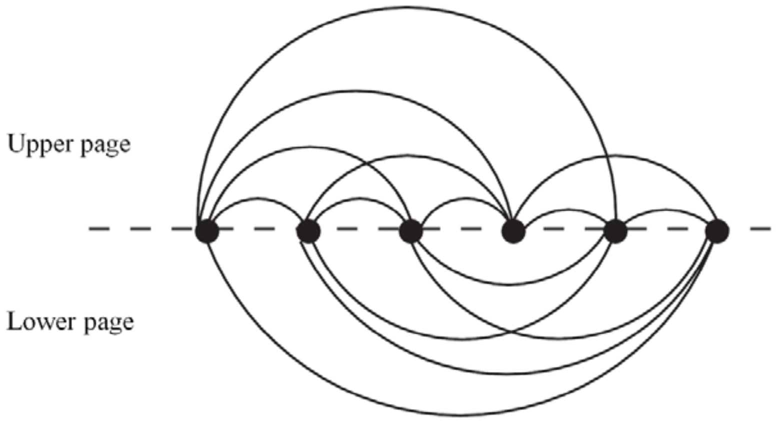

Figure 1. Embedding of the complete graph K_6

The state y_{ij}=1 indicates that the edge *ij* is embedded in the upper page, and the state y_{ij}=0 indicates that the edge *ij* is embedded in the lower page. Therefore, the number of variables is equal to the number of edges in a given graph. The linear crossing number problem can be formulated as [9]:

$$\text{crossing number} = \frac{1}{2}.\sum_{ij}\sum_{kl}(g_{ij}.g_{kl}.d_{ijkl}.y_{ij}.y_{kl}) + \frac{1}{2}.\sum_{ij}\sum_{kl}(g_{ij}.g_{kl}.d_{ijkl}.(1-y_{ij}).(1-y_{kl})) \tag{3}$$

where d_{ijkl} is crossing condition and calculated by:

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

g_{ij} indicates whether the edge *ij* exist.

$$g_{ij} = \begin{cases} 1 & if \quad edge \quad ij \quad exists \\ 0 & otherwise \end{cases} \tag{5}$$

3. OPTIMIZATION ALGORITHMS

A brief overview of four optimization algorithms will be provided in the subsequent sub-sections (PSO, IRO, CBO and ECBO). All of these methods are population-based stochastic algorithms that start with a set of randomly selected candidate solutions. According to a series of rules, mainly inspired from natural phenomena, the existing solutions are perturbed iteratively in order to improve their objective function values [14].

3.1 Particle swarm optimization (PSO)

The particle swarm optimization (PSO), as one of the successful stochastic optimization algorithms, is based on simulation of the social behavior of bird flocking and fish schooling that was introduced by Eberhart and Kennedy [15, 16]. The PSO is a population based technique that involves a number of particles which represent the swarm being initialized randomly in the search space. Each particle represents a candidate solution of the optimum design problem and iteratively moves across the search space. During each generation, each particle updates its velocity and position by learning from the best position achieved so far by the particle itself (*pBest*) and the location of the best fitness achieved so far across the whole population (*gBest*). In 1998, Shi and Eberhart [1] first introduced a new parameter, namely the inertia weight ω to influence convergence. In this paper, the PSO with a fixed inertia weight is utilized. The general structure of a PSO algorithm is as follows:

```
procedure Particle Swarm Optimization (PSO)
    Initialize algorithm parameters
    for each particle
        Initial position is created randomly
        Fitness value is evaluated
    end for
    pBest and gBest are updated
    While maximum iterations is not fulfilled
        for each particle
            Velocity is updated
            $new\_position_i = current\_position_i + velocity_i$
        end for
        pBest and gBest are updated
    end while
end procedure
```

3.2 Improved ray optimization (IRO)

The ray optimization (RO) algorithm proposed by Kaveh and Khayatazad [17,18] was conceptualized using the relationship between the angles of incidence and fraction based on Snell's law. In this method, each agent is modeled as a ray of light that moves in the search space in order to find the global or near-global optimum solution. Improved ray optimization (IRO), proposed by Kaveh et al. [2], employed a new approach for generating new solution vectors which has no limitation on the number of variables, so in the process of algorithm there is no need to divide the variables into groups like RO. In order to generate new solution vectors in the IRO, dynamic parameters are utilized that make a better balance between exploration and exploitation. The procedure which returns the violated agents into feasible search space is also modified in IRO. Instead of changing all the components of violating agents, only the components that violate the boundary are refunded. As a pseudo code, the IRO method has the following form:

```
procedure Improved Ray Optimization (IRO)
    Initialize algorithm parameters
    for each agent
        Initial position and movement vector are created randomly
        Fitness value is evaluated
    end for
    /* LBM and GB are local best memory and global best, respectively*/
    LBM and GB are updated
    While maximum iterations is not fulfilled
        for each agent
            $new\_position_i = current\_position_i + movement\_vector_i$
            Violated components are regenerated
        end for
        LBM and GB are updated
        for each agent
            The direction of movement vector is calculated
            The magnitude of movement vector is calculated
        end for
    end while
end procedure
```

3.3 Colliding bodies optimization (CBO) and its enhanced version (ECBO)

As a newly developed type of meta-heuristic algorithm, colliding bodies optimization (CBO) was introduced by Kaveh and Mahdavi [3,19]. CBO is a population-based stochastic optimization algorithm based on the governing laws of one dimensional collision between two bodies from the physics. Each agent is modeled as a colliding body (CB) with a specified mass and velocity. One object collides with other object and they move toward minimum energy level. The CBO has a simple formulation, and it requires no internal parameter tuning. CBO codes in MATLAB and C++ are presented in [20]. The basic structure of a CBO algorithm is as follows:

```
procedure Colliding Bodies Optimization (CBO)
    Initialize algorithm parameters
    for each CB
        Initial position is created randomly
    end for
    While maximum iterations is not fulfilled
        for each CB
```

```
            Fitness value is evaluated
            The value of mass is calculated
        end for
        Stationary and moving groups are created
        for each CB
            The velocity before collision is calculated
            The velocity after collision is calculated
            Position is updated
        end for
    end while
end procedure
```

The enhanced colliding bodies optimization (ECBO) is introduced by Kaveh and Ilchi Ghazaan [4]. In order to improve the exploration capabilities of the CBO and to prevent premature convergence, a stochastic approach is employed in ECBO that changes some components of CBs randomly. Colliding memory (CM) is also considered to save some historically best CB vectors and their related mass and objective function values to improve the performance of the CBO and reduce the computational cost. MATLAB and C++ codes for ECBO are provided in [20]. The general structure of an ECBO algorithm is as follows:

```
procedure Enhanced Colliding Bodies Optimization (ECBO)
    Initialize algorithm parameters
    for each CB
        Initial position is created randomly
    end for
    While maximum iterations is not fulfilled
        for each CB
            Fitness value is evaluated
            The value of mass is calculated
        end for
        CM is updated
        Population is updated
        Stationary and moving groups are created
        for each CB
            The velocity before collision is calculated
            The velocity after collision is calculated
            Position is updated
        end for
    end while
end procedure
```

4. NUMERICAL EXAMPLES

Six complete graphs (K_8, K_9, …, 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 considered to verify the efficiency of the metaheuristic algorithms. For all algorithms, the population of 40 agents are utilized in K_8, K_9, $K_{3,10}$, $K_{4,5}$, $K_{4,10}$, $K_{5,5}$ problems and for the remaining problems 50 agents are considered. In PSO, the value of acceleration coefficients c_1 and c_2 are both set to 2; the legal velocity range is set to 50% of the search range; and the inertia weight is set to $\omega = 0.4$. In ECBO, the size of the colliding memory is taken as n/10 (n is the number of colliding bodies) and the value of ***Pro*** set to 0.35. Because of the stochastic nature of the algorithms, each example has been solved 20 times independently.

The optimal values of crossing number on complete graphs obtained by metaheuristic algorithms and some other previous studies reported in the literature are presented in Table 1.

Table 1: Best results comparison on complete graphs

Graph	Optimum	Shahrokhi et al. [10]	Cimikowski and Shope [11]	Wang and Okazaki [9]	Present work			
					PSO	IRO	CBO	ECBO
K_8	18	19	18	18	18	18	18	18
K_9	36	36	36	36	36	36	36	36
K_{10}	60	62	60	60	60	60	60	60
K_{11}	100	100	100	100	104	100	100	100
K_{12}	150	154	150	150	158	153	151	150
K_{13}	-	265	225	225	245	227	231	225

It can be seen that the best answers are achieved by Cimikowski and Shope [11], Wang and Okazaki [9]; and ECBO. These methods obtained the optimum values for K_8 through K_{12} and for K_{13} their results are identical. Shahrokhi et al. [9] and PSO has the worst performance and the results achieved by IRO and CBO are approximately identical. The statistical results are shown in Table 2 (The parameters "success rate" and "FEs" indicate the percentage of successful runs for which the optimum value could be found and the number of function evaluations for the best result, respectively). Wang and Okazaki [9]; and ECBO obtain the optimum values in all independent runs. The performance of the remaining methods except the PSO is nearly the same. Embedding of the complete graph K_8 obtained by ECBO is shown in Fig. 2.

Table 2: Statistical results on complete graphs

Graph		Cimikowski and Shope [11]	Wang and Okazaki [9]	Present work			
				PSO	IRO	CBO	ECBO
K_8	Success rate	19	100	15	50	40	100
	Average	19.8	18	20.1	18.95	19.1	18
	Std. dev.	N/A	N/A	1.89	1.14	1.09	0
	FEs	N/A	N/A	240	280	240	280
K_9	Success rate	44	100	45	65	75	100
	Average	38.9	36	38.4	37.8	36.8	36
	Std. dev.	N/A	N/A	7.84	7.56	2.56	0
	FEs	N/A	N/A	200	400	200	280
K_{10}	Success rate	12	100	5	10	5	100
	Average	63.6	60	69.8	63.75	64.6	60
	Std. dev.	N/A	N/A	22.36	4.58	7.34	0

	FEs	N/A	N/A	280	1100	1650	1000
K_{11}	Success rate	29	100	0	5	5	100
	Average	108	100	115.4	106.6	105.5	100
	Std. dev.	N/A	N/A	24.4	29.24	11.95	0
	FEs	N/A	N/A	300	950	1750	950
K_{12}	Success rate	13	100	0	0	0	100
	Average	166	150	183.8	164.45	161.15	150
	Std. dev.	N/A	N/A	84.46	109.24	59.53	0
	FEs	N/A	N/A	550	1850	1350	2300
K_{13}	Success rate	-	-	-	-	-	-
	Average	239	225	278.2	245.1	244.5	225
	Std. dev.	N/A	N/A	180.96	187.39	115.15	0
	FEs	N/A	N/A	500	3350	1650	2150

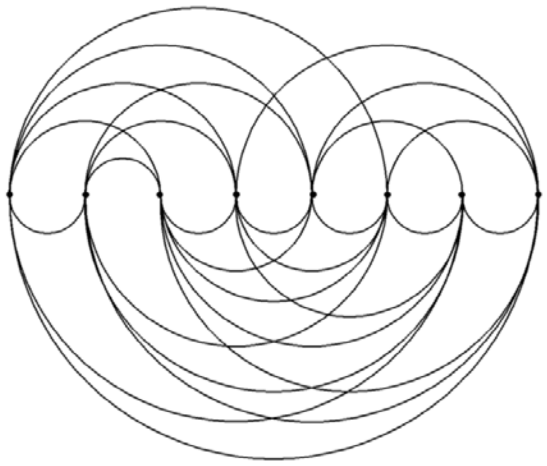

Figure 2. Embedding of the complete graph K_8 obtained by ECBO

Table 3 provides a comparison between the best results obtained by proposed algorithms on complete bipartite graphs. The best results found by CBO and ECBO are identical and they perform better than PSO and IRO.

Table 3: Best results comparison on complete bipartite graphs

Graph	Optimum	PSO	IRO	CBO	ECBO
$K_{3,10}$	20	28	20	20	20
$K_{3,15}$	49	71	49	49	49
$K_{4,5}$	8	10	10	10	10
$K_{4,10}$	40	63	54	54	54
$K_{4,15}$	98	217	133	130	130
$K_{5,5}$	16	20	20	20	20
$K_{5,10}$	80	109	100	100	100
$K_{5,15}$	196	360	264	244	244

The optimum values can be achieved by IRO, CBO and ECBO only for $K_{3,10}$ and $K_{3,15}$.

The average and standard deviation of results for twenty independent runs and number of function evaluations for the best results are shown in Table 4. ECBO performs better than other algorithms because its standard deviation is zero in all examples. Embedding of the complete bipartite graph $K_{3,10}$ found by ECBO is depicted in Fig. 3.

Table 4: Statistical results on complete bipartite graphs

Graph		PSO	IRO	CBO	ECBO
	Average	37.25	24.35	26.45	20
$K_{3,10}$	Std. dev.	31.28	39.62	22.34	0
	FEs	560	680	960	450
	Average	94.9	64.7	66.8	49
$K_{3,15}$	Std. dev.	214.99	230.91	233.16	0
	FEs	350	3350	1500	1150
	Average	12.3	11.35	10.95	10
$K_{4,5}$	Std. dev.	8.61	6.32	1.94	0
	FEs	200	240	320	320
	Average	86.2	66.8	67	54
$K_{4,10}$	Std. dev.	93.16	108.76	130.7	0
	FEs	440	1680	880	520
	Average	235.5	160.85	157.7	130
$K_{4,15}$	Std. dev.	200.55	336.55	587.41	0
	FEs	350	1490	1650	1750
	Average	26.2	22.3	23	20
$K_{5,5}$	Std. dev.	25.16	11.71	9.4	0
	FEs	160	360	480	240
	Average	152.05	122.45	118.85	100
$K_{5,10}$	Std. dev.	276.25	317.34	193.42	0
	FEs	500	8750	1500	1150
	Average	406.4	316.6	307.2	244
$K_{5,15}$	Std. dev.	463.44	1326.44	1634.96	0
	FEs	400	8850	2050	2700

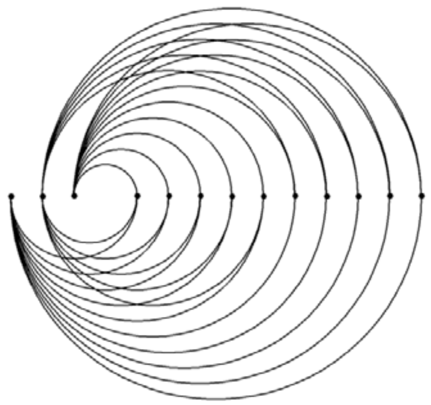

Figure 3. Embedding of the complete graph $K_{3,10}$ obtained by ECBO

5. CONCLUSION

Four metaheuristic algorithms are considered for the fixed linear crossing number problem. In this problem, the vertices of a graph are placed in a fixed order along a horizontal "node line" in the plane, each edge is drawn as an arc in one of the two half-planes (pages), and the objective is to minimize the number of edge crossings. The experimental results indicate that all algorithms nearly have an acceptable performance for the crossing number problem. ECBO yields near-optimal solutions and outperforms the other methods. It is superior over other methods in terms of reliability and solution accuracy. However, the performance of IRO and CBO is approximately the same and better than PSO.

REFERENCES

1. Shi Y, Eberhart RC. A modified particle swarm optimizer, *In Proc IEEE Congr Evol Comput*, May 1998, pp. 69-73.
2. Kaveh A, Ilchi Ghazaan M, Bakhshpoori T. An improved ray optimization algorithm for design of truss structures, *Period Polytech*, No. 2, 2013; **57**:1-15.
3. Kaveh A, Mahdavai VR. Colliding bodies optimization: A novel meta-heuristic method, *Comput Struct* 2014; **139**: 18-27.
4. Kaveh A, Ilchi Ghazaan M. Enhanced colliding bodies optimization for design problems with continuous and discrete variables, *Adv Eng Softw* 2014; **77**: 66-75.
5. Tur´an P. A Note of Welcome, *J Graph Theory* 1977; **1**: 7-9.
6. Leighton FT. New lower bound techniques for VLSI, *Math Sys Theory* 1984; **17**: 47-70.
7. Kaveh A. Space structures and the crossing number of their graphs, *Mech Struct Mach* 1993; **21**: 151-66.
8. Kaveh A. *Structural Mechanics: Graph and Matrix Methods*, UK, 2nd edition, 1996.
9. Wang RL, Okazaki K. Artificial neural network for minimum crossing number problem, *Fourth Inter Conf Machine Learning Cyber*, August 2005; pp. 18-21.
10. Shahrokhi F, Szekely LS, Sykora O, Vrto I. The book crossing number of a graph, *J Graph Theory* 1996; **21**: 413-24.
11. Cimikowski R, Shope P, A neural network algorithm for a graph layout problem, IEEE *Trans Neural Netwoek* 1996; **4**: 341-5.
12. Pan S, Richter RB. The crossing number of K_{11} is 100, *J Graph Theory* 2007; **56**: 128-34.
13. Woodall DR. Cyclic-order graphs and Zarankiewicz's crossing-number conjecture, *J Graph Theory* 1993; **17**: 657-71.
14. Kaveh A, Zolghadr A. Comparison of nine meta-heuristic algorithms for optimal design of truss structures with frequency constraints, *Adv Eng Softw* 2014; 76: 9-30.
15. Eberhart RC, Kennedy J. A new optimizer using particle swarm theory, In *Proc 6th Int Symp Micromach Hum Sci* 1995, pp. 39-43.
16. Kennedy J, Eberhart RC. Particle swarm optimization. In *Proc IEEE Int Conf Neural Networks* 1995, pp. 1942-8.
17. Kaveh A, Khayatazad M. A new meta-heuristic method: ray optimization, *Comput Struct* 2012; **112-113**: 283-94.

18. Kaveh A, Khayatazad M. Ray optimization for size and shape optimization of truss structures, *Comput Struct* 2013; **117**: 82-94.
19. Kaveh A, Mahdavai VR. Colliding Bodies Optimization for discrete optimal design of truss structures, *Comput Struct* 2014;**139**:43-53.
20. Kaveh A, Ilchi Ghazaan M. Computer codes for colliding bodies optimization and its enhanced version, *Int J Optim Civil Eng* 2014; **4(3)**: 321-339.