2011

Ordering for bandwidth and profile minimization problems via charged system search algorithm

Ali Kaveh

Pezhman Sharafi
University of Wollongong, psharafi@uow.edu.au

IJST, Transactions of Civil Engineering, Vol. 36, No. C1, pp 39-52
Printed in The Islamic Republic of Iran, 2012

ORDERING FOR BANDWIDTH AND PROFILE MINIMIZATION PROBLEMS VIA CHARGED SYSTEM SEARCH ALGORITHM*

A. KAVEH[1**] AND SHARAFI[2]

[1]Centre of Excellence for Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, I. R. of Iran
Email: alikaveh@iust.ac.ir
[2]School of Civil, Mining and Environmental Engineering, University of Wollongong, Northfields Avenue, Wollongong, NSW 2522, Australia

Abstract– In this paper the recently developed meta-heuristic optimization method, known as charged system search (CSS), is utilized for optimum nodal ordering to minimize bandwidth and profile of sparse matrices. The CSS is an optimization algorithm, which is based on the governing laws of Coulomb and Gauss from electrostatics and the Newtonian mechanics of motion. The bandwidth and profile of some graph matrices, which are pattern equivalent to structural matrices, are minimized using this approach. This shows the applicability of the meta-heuristic algorithms in bandwidth and profile optimization. Comparison of the results with those of some existing methods, confirms the robustness of the new algorithm.

Keywords– Profile reduction, bandwidth reduction, ordering, graphs, charged system search, optimization

1. INTRODUCTION

The analysis of many problems in structural engineering involves the solution of simultaneous equations. Such non-singular systems of linear algebraic equations are in the form $Ax = b$ arising from the finite element method. These kinds of equations usually involve a positive definite, symmetric, and sparse matrix coefficient A. For large structures a great deal of the computer execution time and memory are devoted to the solution of these equations. Hence some appropriate specified patterns for the solutions of the corresponding equations have been provided, like banded form, profile form and partitioned form. These patterns are obtained by nodal ordering of the corresponding models.

In finite element analysis, for the case of one degree of freedom per node, performing nodal ordering is equivalent to reordering the equations. In a more general problem with m degree of freedom per node, there are m coupled equations produced for each node. In this case re-sequencing is usually performed on the nodal numbering of the graph models, to reduce the bandwidth, profile or wavefront, because the size of these problems are m fold smaller than those for m degree of freedom numbering. In this article, the mathematical model of FEM is considered as an element clique graph, and nodal ordering is performed for reducing the bandwidth and profile of the corresponding matrices [1, 2].

Nodal ordering has an important role in the solution of sparse systems. This is achieved by permuting the rows and columns of a matrix by proper renumbering of the nodes of the associated graph. Two important subjects in nodal ordering are bandwidth optimization and profile reduction. In fact, for sparse matrices the size can be considered as the bandwidth or profile of such matrices. These problems have created considerable interest over the years because of its practical relevance for a significant range of global optimization applications. Since the nature of the problem of nodal ordering is NP-Complete, many

*Received by the editors December 29, 2010; Accepted May 17, 2011.
**Corresponding author

approximate methods and heuristics are proposed like, Cuthill-McKee [3], Souza and Murray [4], Sloan [5], Gibbs-King [6], Kaveh [7, 8], Kaveh and Rahami [9] and Kaveh and Rahimi Bondarabady [10] algorithms.

Meta-heuristic algorithms mostly tend to perform properly for the optimization problems [11-13]. This is because these methods avoid simplifying or making shortening assumptions about the original form. Evidence of this is their successful applications to a vast variety of fields, such as engineering, art, biology, economics, marketing, genetics, operations research, robotics, social sciences, physics, politics and chemistry. As a newly developed type of meta-heuristic algorithm, the charged system search (CSS) has recently been introduced for design of structural problems (Kaveh and Talatahari [14, 15]). This method utilizes the governing laws of Coulomb and Gauss from electrostatics and the Newtonian law of mechanics. Inspired by these laws, a model is created to formulate the nodal ordering optimization problem. CSS contains a number of agents called charged particle (CP). Each CP is considered as a charged sphere which exerts an electric force on other CPs according to the Coulomb and Gauss laws. The resultant forces and the motion laws determine the new location of the CPs. Using these laws provides a good balance between the exploration and the exploitation of the algorithm. As a result, CSS can easily be utilised for both discrete and continuous optimization problems. In this paper, the CSS developed in [14-15], is employed for the nodal ordering as a discrete optimization problem.

2. BACKGROUND

a) Electrostatic laws

In physics, the space surrounding an electric charge has a property known as the electric field. This field exerts a force on other electrically charged objects. The electric field surrounding a point charge is given by Coulomb's law. Coulomb has confirmed that the electric force between two small charged spheres is proportional to the inverse square of their separation distance r_{ij}. Therefore, this law provides the magnitude of the electric force (Coulomb force) between the two point charges. This force on a charge, q_j at position r_i, experiencing a field due to the presence of another charge, q_j at position r_i, can be expressed as

$$F_{ij} = k_e \frac{q_i q_j}{{r_{ij}}^2} \frac{r_i - r_j}{\left\| r_i - r_j \right\|} \tag{1}$$

where k_e is a constant known as the Coulomb constant; r_{ij} is the separation of the two charges [16].

Consider an insulating solid sphere of radius "*a*" which has a uniform volume charge density and carries a total charge of magnitude q_i. The magnitude of the electric force at a point outside the sphere is defined as Eq. (1), while this force can be obtained using Gauss's law at a point inside the sphere as

$$F_{ij} = k_e \frac{q_i q_j}{a^3} r_{ij} \frac{r_i - r_j}{\left\| r_i - r_j \right\|} \tag{2}$$

In order to calculate the electric force on a charge (q_j) at a point (r_j) due to a group of point charges, the principle of superposition is applied to electric forces as

$$F_j = \sum_{i=1, i \neq j}^{N} F_{ij} \tag{3}$$

where N is the total number of charged particles and F_{ij} is equal to

$$F_{ij} = \begin{cases} k_e \dfrac{q_i}{a^3} r_{ij} \dfrac{r_i - r_j}{\left\| r_i - r_j \right\|} & if\ r_{ij} < a \\ k_e \dfrac{q_i}{r_{ij}^2} \dfrac{r_i - r_j}{\left\| r_i - r_j \right\|} & if\ r_{ij} \geq a \end{cases} \tag{4}$$

Therefore, the resulted electric force can be obtained as [2]

$$F_j = k_e q_j \sum \left(\frac{q_i}{a^3} r_{ij} . i_1 + \frac{q_i}{r_{ij}^2} . i_2 \right) \frac{r_i - r_j}{\left\| r_i - r_j \right\|} \qquad \begin{Bmatrix} i_1 = 1, i_2 = 0 \Leftrightarrow r_{ij} < a \\ i_1 = 0, i_2 = 1 \Leftrightarrow r_{ij} \geq a \end{Bmatrix} \tag{5}$$

b) Newtonian mechanics laws

Newtonian mechanics studies the motion of objects. In the study of motion, the moving object is described as a particle regardless of its size. In general, a particle is a point-like mass having infinitesimal size. The motion of a particle is completely known if the particle's position in space is known at all times. The displacement of a particle is defined as its change in position. As it moves from an initial position r_{old} to a final position r_{new}, its displacement is given by

$$\Delta r = r_{new} - r_{old} \tag{6}$$

The slope of tangent line of the particle position represents the velocity of this particle as

$$v = \frac{r_{new} - r_{old}}{t_{new} - t_{old}} = \frac{r_{new} - r_{old}}{\Delta t} \tag{7}$$

The acceleration of the particle is defined as the change in the velocity divided by the time interval Δt during which that change has occurred:

$$\alpha = \frac{v_{new} - v_{old}}{\Delta t} \tag{8}$$

The displacement of any object can now be obtained as a function of time as

$$r_{new} = \frac{1}{2} \alpha . \Delta t^2 + v_{old} . \Delta \Delta + r_{old} \tag{9}$$

Also, according to Newton's second law, we have

$$F = m . \alpha \tag{10}$$

where *m* is the mass of the objective. Substituting Eq. (10) in Eq. (9), we have

$$r_{new} = \frac{1}{2} \frac{F}{m} . \Delta t^2 + v_{old} . \Delta t \ + r_{old} \tag{11}$$

3. THE RULES OF THE CHARGED SYSTEM SEARCH

In this section, the recently developed optimization algorithm in [14, 15], which is called Charged System Search, is briefly presented utilizing the aforementioned physics laws. In the CSS, each solution candidate X_i containing a number of decision variables (i.e. $X_i = \{ x_{i,j} \}$) is considered as a charged particle. The charged particle is affected by the electrical fields of the other agents. The quantity of the resultant force is determined by using the electrostatics laws as discussed in Section 2.1, and the quality of

the movement is determined using the Newtonian mechanics laws in Section 2.2. Thus an agent with good results must exert a stronger force than the bad ones, so the amount of charge will be defined considering the objective function value, *fit(i)*. In order to introduce CSS, the following rules are introduced:

Rule 1: In CSS each CP has a magnitude of charge (q_i), and as a result creates an electrical field around its space. The magnitude of the charge is defined considering the quality of its solution, as follows:

$$q_i = \frac{fit(i) - fit_{best}}{fit_{best} - fit_{worst}} \quad i=1, 2, \ldots, N \tag{12}$$

where fit_{best} and fit_{worst} are so far the best and the worst fitness of all particles; *fit(i)* represents the objective function value or the fitness of the agent *i*; and *N* is the total number of CPs. The separation distance r_{ij} between two charged particles is defined as follows:

$$r_{ij} = \frac{\left\| X_i - X_j \right\|}{\left\| (X_i + X_j)/2 - X_{best} \right\| + \varepsilon} \tag{13}$$

where X_i and X_j are the positions of the *i*th and *j*th CPs, X_{best} is the position of the best current CP, and ε is a small positive number to avoid singularities.

Rule 2: The initial positions of CPs are determined randomly in the search space and the initial velocities of charged particles are assumed to be zero.

Rule 3: Electric forces between any two charged particles are attractive. Utilizing this rule increases the exploitation ability of the algorithm. Though it is possible to define the repelling force between CPs as well, for our problems this does not seem to be unnecessary. When a search space is a noisy domain, having a complete search before converging to a result is required; in such conditions the addition of the ability of repelling forces to the algorithm may improve its performance.

Rule 4: Good CPs can attract the other agents and bad CPs repel the others, proportional to their rank, that is

$$c_{ij} \propto rank(CP_j) \wedge \begin{cases} 0 < c_{ij} \le +1 & \text{if the CP is above average} \\ -1 \le c_{ij} < 0 & \text{if the CP is below average} \end{cases} \tag{14}$$

where c_{ij} is a coefficient determining the type and the degree of influence of each CP on the other agents, considering their fitness and apart from their charges. This means that good agents are awarded the capability of attraction and bad ones are given the repelling feature, which will improve the exploration and exploitation abilities of the algorithm. On the one hand, when a good agent attracts a bad one, the exploitation ability for the algorithm is provided. On the other hand, when a bad agent repels a good CP, the exploration is provided.

Rule 5: The value of the resultant electrical force affecting a CP is determined using Eq. (5), as

$$F_j = q_j \sum_{i,i\neq j} \left(\frac{q_i}{a^3} r_{ij} i_1 + \frac{q_i}{r_{ij}^2} i_2 \right) c_{ij} (X_i - X_j) \quad \begin{cases} j = 1, 2, \ldots, N \\ i_1 = 1, i_2 = 0 \Leftrightarrow r_{ij} < a \\ i_1 = 0, i_2 = 1 \Leftrightarrow r_{ij} \ge a \end{cases} \tag{15}$$

where F_j is the resultant force acting on the *j*th CP, as illustrated in Fig. 1. In this algorithm, each CP is considered to be a charged sphere with radius *a* having a uniform volume charge density. In this paper, "*a*" is set to unity.

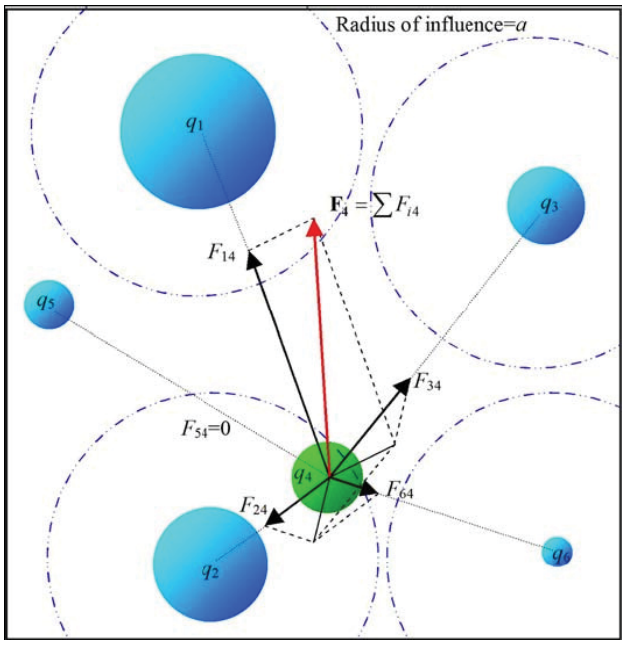

Fig. 1. The resultant electrical force acting on a CP [14]

Rule 6: The new position and velocity of each CP is determined considering Eq. (7) and Eq. (11), as follows

$$x_{j,new} = Fix(rand_{j1} k_a \frac{F_j}{m_j} \Delta t^2 + rand_{j2} k_v v_{j,old} \Delta t + x_{j,old}) \tag{16}$$

$$v_{j,new} = \frac{x_{j,new} - x_{j,old}}{\Delta t} \tag{17}$$

where $Fix(X)$ is a function which rounds each element of $\boldsymbol{X}$ to the nearest permissible discrete value; k_a is the acceleration coefficient; k_v is the velocity coefficient to control the influence of the previous velocity; and $rand_{j1}$ and $rand_{j2}$ are two random numbers uniformly distributed in the range of (0,1). m_j is the mass of the CPs which is equal to q_j in this paper. Δt is the time step and is set to one. Fig. 2 illustrates the movement of a CP to its new position using this rule.

The effect of the previous velocity and the resultant force acting on a CP can be decreased or increased based on the values of the k_v and k_a, respectively. Excessive search in the early iterations may improve the exploration ability; however, it must be deceased gradually, as described before. Since k_a is the parameter related to the attracting forces, selecting a large value for this parameter may cause a fast convergence, and a small value can increase the computational time. In fact, k_a is a control parameter of the exploitation. Therefore, choosing an incremental function can improve the performance of the algorithm. Also, the direction of the previous velocity of a CP is not necessarily the same as the resultant force. Thus, it can be concluded that the velocity coefficient k_v controls the exploration process and therefore a decreasing function can be selected. Thus, k_v and k_a are defined as

$$k_v = 0.5(1 - iter/iter_{max}), \quad k_a = 0.5(1 + iter/iter_{max}) \tag{18}$$

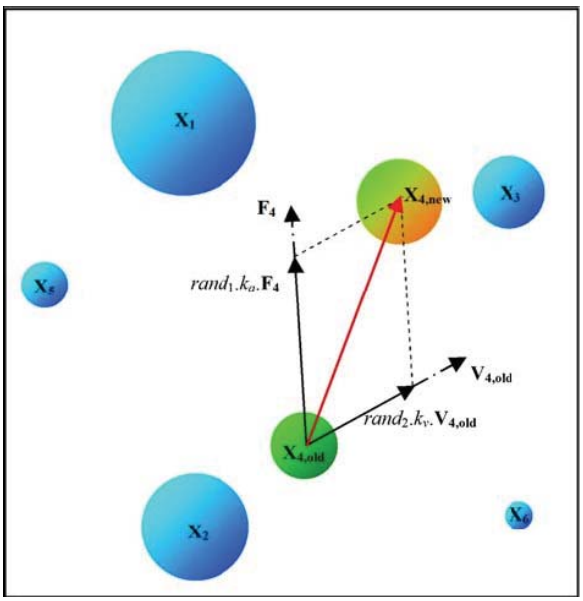

Fig. 2. movement of a CP to its new position [14]

Rule 7: Charged memory (CM) is utilized to save a number of the best so far solutions. Here, the size of the CM is taken as *N/4*. The vectors stored in the CM can influence the CPs. This may increase the computational cost, and therefore it is assumed that the same number of the worst particles cannot attract others.

Rule 8: The agents violating the limits of the variables are regenerated using the harmony search-based handling approach as described by Kaveh and Talatahari [14, 15].

Rule 9: Maximum number of iterations is considered as the terminating criterion.
The general flowchart of the CSS algorithm is illustrated in Fig. 3.

4. DEFINITIONS OF THE PROBLEM

a) Nodal ordering for bandwidth reduction

Let *G(N,A)* be a graph with members set $A(|A| = a)$ and nodes set $N(|N| = n)$. The aim is to label or assign the set of integers *{1,2,3,...,n}* to the nodes of *G*. Let *As(I)* be the label or the integer, assigned to node *i*, where each node has a different label. The bandwidth of node *i* for this assignment, $bw_{AS}(i)$, is the maximum of the difference between *As(i)* and *As(j),* where *As(j)* is the label of adjacent nodes of node *i or* the number assigned to its adjacent nodes. Then

$$bw_{As}(i) = max\{|As(i) - As(j)| : j \in N(i)\} \tag{19}$$

where *N(i)* is the set of nodes adjacent to *i*. The bandwidth of the graph with respect to the assignment, *As(i)* , is then

$$BW_{As}(G) = max\{bw(i) : i \in G\} \tag{20}$$

The bandwidth of the graph is the minimum value of BW_{AS} over all possible assignments:

$$BW(G) = min\{BW_{As}(G) : \forall As(i)\} \tag{21}$$

Therefore, in the bandwidth reduction problem, one seeks an assignment *As(i)* that minimizes $BW_{As}(G)$. Such an assignment keeps all the non-zero elements of the matrix in a band, which is as close as possible to the main diagonal.

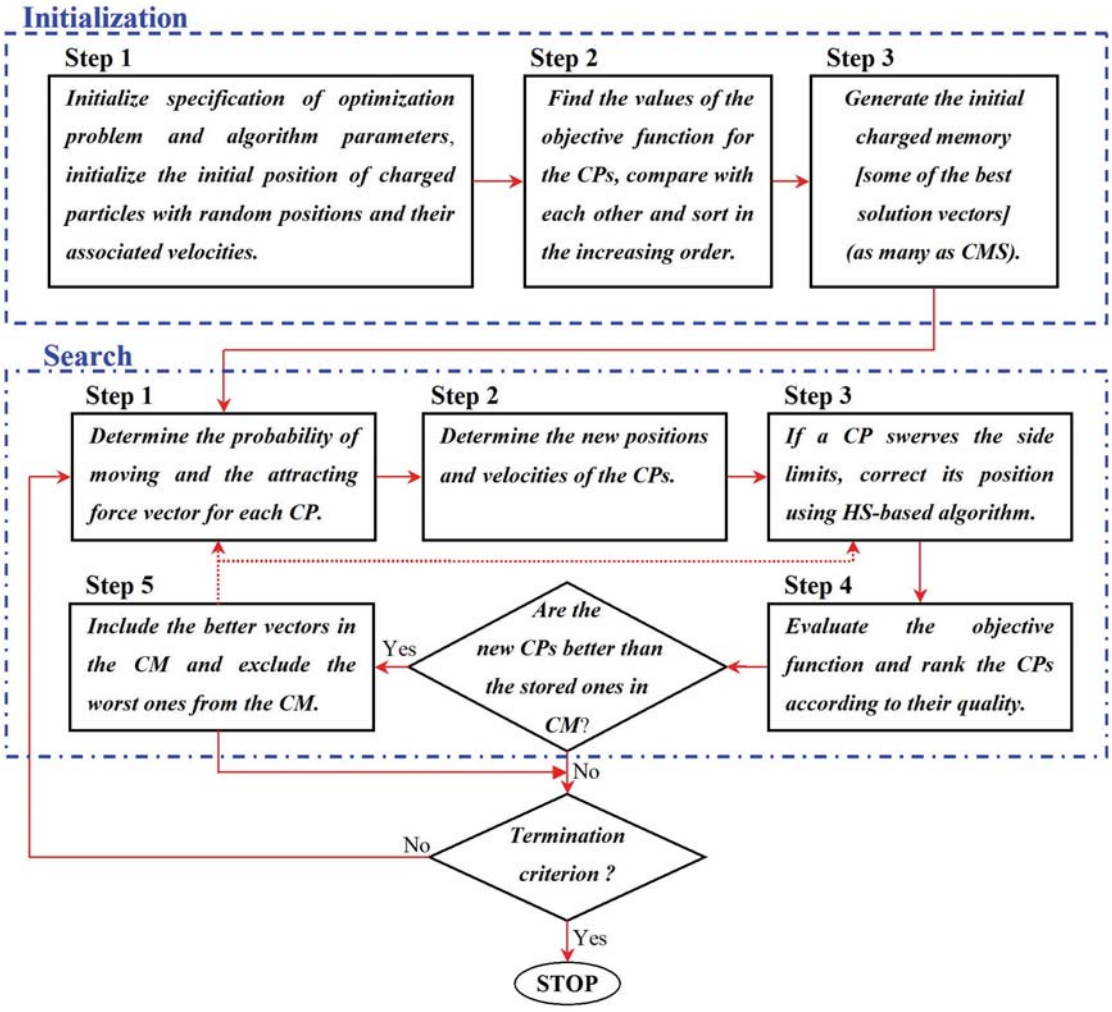

Fig. 3. The general flowchart of the CSS algorithm [14]

b) Nodal ordering for profile reduction

In the above mentioned graph, for the purpose of finding an optimal nodal ordering, we will try to label or assign the set of integers {*1, 2, 3,..., n*} to the nodes of *G*. The profile of the $N \times N$ matrix related to graph *G*, for the assignment *As(i)*, is defined as

$$P_{As} = \sum_{i=1}^{N} b_i \tag{22}$$

where the row bandwidth, b_i, for row *i* is defined as the number of inclusive entries from the first non-zero element in the row to the (*i+1*) th entry for this assignment. The efficiency of any given ordering for the profile solution scheme is related to the number of active equations during each step of the factorisation process. Formally, row *j* is defined to be active during elimination of column *i* if $j \geq i$ and there exists $a_{ik} = 0$ with $k \leq i$. Hence, at the *i*th stage of the factorization, the number of active equations is the number of rows of profile that intersect column *i,* ignoring those rows already eliminated. Let f_i

denote the number of equations that are active during the elimination of the variable x_i. It follows from the symmetric structures of the matrix that

$$P_{As} = \sum_{i=1}^{N} f_i = \sum_{i=1}^{N} b_i \tag{23}$$

where f_i is commonly known as the wavefront or frontwidth. Assuming that N and the average value of f_i are reasonably large, it can be shown that a complete profile or front factorisation requires approximately $O(NF_{rms}{}^2)$ operations, where F_{rmsi} is the root-mean square wavefront, which is defined as

$$F_{rms} = [\frac{1}{N}\sum_{i=1}^{N} f_i^2\ ]^{0.5} \tag{24}$$

5. THE CSS ALGORITHM FOR NODAL ORDERING

This algorithm attempts to find an optimal assignment for nodal ordering of a graph to reduce the bandwidth or profile of the associated matrix employing a charged system search algorithm. The basis of the algorithms for both bandwidth and profile reduction are identical and is based on reordering or assigning new labels to the graph nodes to achieve the optimal bandwidth or profile. The only difference is in defining the objective functions. That is, the main procedure of the CSS algorithm for reordering is the same but the objective function for bandwidth reduction comes from Eq. (21), while for profile reduction it comes from Eq. (23).

For an *n*n* sparse matrix associated to graph *G*, each permutation of rows and columns leads to a new reordering called the assigned set. If the initial numbering of the graph is $\{1,2,3,...,n\}$, each permutation of this set will be a new assigning list. The aim is to find the optimal assigning list in order to reach the best bandwidth or profile.

Each probable permutation of set $\{1,2,3,...,n\}$ is considered to be a potential solution which is called an agent. In CSS these agents are regarded as CPs. In fact, each solution candidate X_i containing a number of decision variables $x_{i,j}$, is considered to be a charged particle and each $x_{i,j}$ presents the number assigned to the node *j* in the original graph. Thus a solution candidate X_i which represents the position of CP_i, contains *n* arrays $x_{i,j}$ $(j=1,2,...,n)$ which stand for the assigned numbers.

The algorithm for nodal ordering follows the above mentioned nine general rules of CSS algorithms. As stated before, due to the nature of nodal ordering problem, the discrete version of CSS, consisting of nine steps is utilised.

Step 1: The number of CPs, i.e. candidate agents, is determined. For nodal ordering of sparse matrices this number is set to *N=[fix(n/100)+5]*, which means that for every *100* nodes one additional CP is added and at least *5* CPs are needed for any problem. Using a larger number of CPs may result in more accurate results, however, it significantly increases the computational time. On the other hand, using a smaller number of CPs may lead to undesirable results. The considered number of CPs is capable of keeping the balance at a moderate level.

Step 2: The CPs are defined and settled in their initial positions. For this purpose a random permutation of set $\{1,2,3,...,n\}$ is assigned to each agent as initial candidate solutions. That is, the initial candidate solutions X_i and, as a result, their positions $\{x^{(0)}{}_{i,j}\}$ are randomly nominated. In other words, in this phase, *N* candidate solution X_i $(i=1,2,...,N)$ which are located in their positions presented by $x^{(0)}{}_{i,j}$ are defined. $(j=1,2,...,n)$. The initial velocity for all CPs are considered to be zero. $(v^{(0)}{}_{i,j}=0 \quad \forall\ i,j)$

Step 3: The magnitude of charge for each CP is calculated using Rule 1. For this purpose the objective functions for each agent must be calculated. As mentioned before, this phase is the only distinction between bandwidth and profile reduction algorithm. The objective function for bandwidth reduction is obtained from Eq. (21), while for profile reduction it is calculated from Eq. (23). In this step, when objective functions are calculated, they should be put in order and the best and the worst ones and the best and the worst *N*/5 agents are saved. This will help the algorithm to judge better in the next steps. Then the magnitude of charge for each CP, i.e. q_i, is obtained through the Eq. (12).

Step 4: The separation distance between CPs is calculated. In the previous step, the position of each CP is defined by a coordinate of *n* arrays, having the X_i for all CPs. The separation distance between them is calculated using the Eq. (13). It should be mentioned that in such discrete problems in which X_i is an *n*-dimensional array, the intention of calculating distance between every two CPs is to find how far the two assumed nodes are in the *n*-dimensional space. In fact, the calculations of distance, velocity and acceleration are all made in a multidimensional space.

Step 5: The type and the degree of influence of each CP on the other agents are determined. For this reason, using the rank of the CPs obtained in step 3, a number between +1 and −1 is assigned to each agent proportional to its rank. That is, the number +1 is assigned to the best agent and −1 to the worst one and so on. Such an assignment leads to improvement of the abilities of exploration and exploitation simultaneously.

Step 6: The value of the resultant electrical force affecting a CP is determined using the Eq. (15). Each F_j is an *n*-dimensional array and shows the tendency of agent *j* toward other CPs.

Step 7: New position and velocity of each CP is determined considering Eq. (16) and (17), respectively. In Eq. (16) the function $Fix(\mathbf{X})$ shifts each $x_{i,j}$ to its nearest position. That is, the nearest permissible digit assigns to each $x_{i,j}$. As mentioned before, each new position determined by an *n*-dimensional array shows the new renumbering of CPs i.e., the new numbers assigned to nodes.

Step 8: The agents violating the limits of the variables are regenerated using the harmony search-based handling approach. Then the best so far solutions are saved.

Step 9: Maximum number of iterations is considered as the terminating criterion.

6. NUMERICAL EXAMPLES

In this section, three examples are presented and the results are compared to those of the other algorithm in Table 1. Then a comparison is made for the convergence rate of different algorithms for each example. For profile reduction, the results are compared to those of Sloan [5], King [17], and Kaveh and Sharafi [18, 19]; and for bandwidth minimization, the 4-step algorithm [7, 8], and an ACO algorithm [20] are used to perform the comparison.

The topological properties of the finite element models are transferred to the connectivity properties of graphs by the clique graphs [15]. This graph has the same nodes as those of the corresponding finite element model, and the nodes of each element are cliqued, avoiding the multiple edges for the entire graph.

All computations are performed on P9700 @2.40 GHz computer running MATLAB R2009b. In order to ensure that the obtained solution from ACO is global or near global optimum, many runs are performed

in parallel. Since each run is fully independent of the others, the program could be run in parallel so that the total execution time practically became the same as that required for a single run.

Example 1: Consider a finite element mesh (FEM) of a fan. The element clique graph of this model contains 1575 nodes as shown in Fig. 4. The performance of the CSS algorithm and some other algorithms are tested on this model, and the results are presented in Table 1.

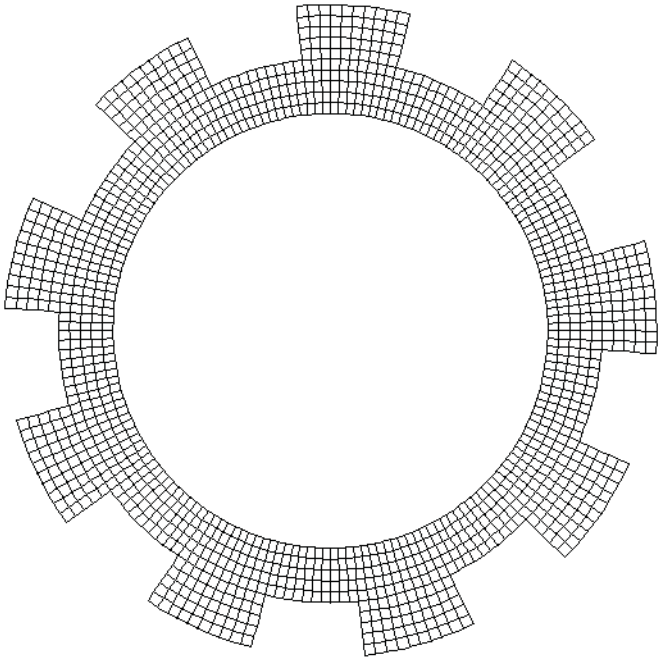

Fig. 4. The element clique graph of a fan

Example 2: The FEM of a shear wall with 760 nodes and four openings is shown in Fig. 5. Similar to the previous example, the performance of the CSS algorithm and some other algorithms are tested on this model and the results are presented in Table.1.

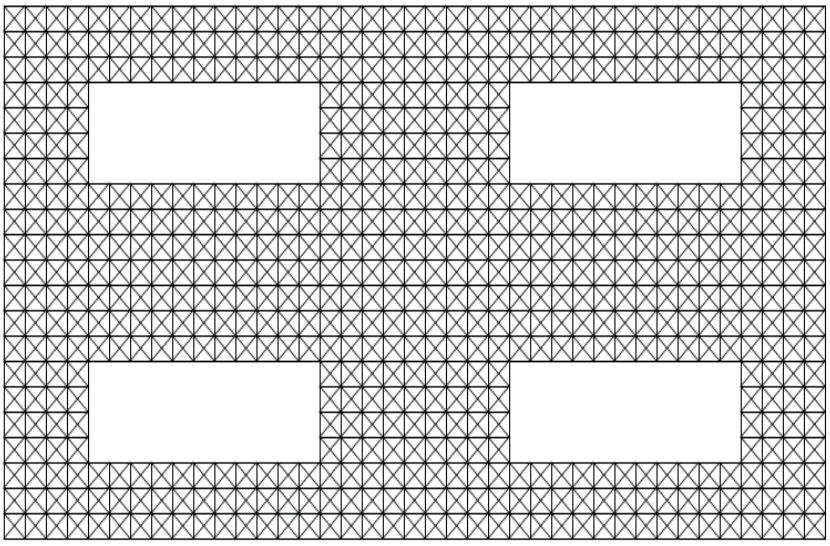

Fig. 5. The element clique graph of a rectangular FEM with four openings

Example 3: An H-shape finite element mesh (FEM) with 4949 nodes is considered, as shown in Fig. 6. The element clique graph of this model contains 4949 nodes and 9688 beam elements (edges). The performance of the CSS algorithm and some other algorithms are tested on this model, and the results are presented in Table 1.

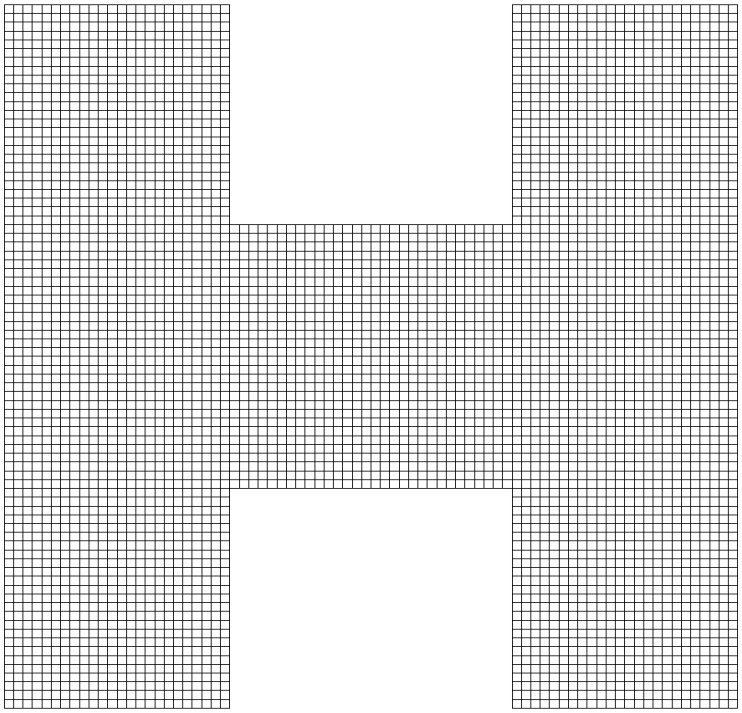

Fig. 6. An H-shaped finite element grid

Table 1. Comparison of the results

Instance	Type of ordering	Initial value	Algorithm	Optimized value	The CPU time (seconds)
Example 1 (Fig. 4)	Profile minimization	144351	Sloan [5]	31002	32.7
			King [17]	31982	24.1
			ACO [20]	29665	55.9
			CSS*	28770	17.3
	Bandwidth minimization	461	4-step [7]	23	4.9
			ACO [20]	23	10.7
			CSS*	21	4.4
Example 2 (Fig. 5)	Profile minimization	37584	Sloan [5]	19110	11.1
			King [17]	19613	9.8
			ACO [19]	19007	18.3
			CSS*	19232	8.2
	Bandwidth minimization	382	4-step [7]	46	1.8
			ACO [20]	42	4.4
			CSS*	41	2.0
Example 3 (Fig. 6)	Profile minimization	345437	Sloan [5]	210845	117.8
			King [17]	211731	98.2
			ACO [19]	208945	296.6
			CSS*	206649	98.7
	Bandwidth minimization	407	4-step [7]	66	17.7
			ACO [19]	60	29.5
			CSS*	58	13.3

* Present algorithm

7. CONCLUDING REMARKS

The main objective of this paper is to show the applicability and robustness of the CSS for nodal ordering of sparse matrices as a discrete optimization problem. From Table 1, it can be observed that the results are quite satisfactory, comparing other well-known graph theoretical methods and ACO.

As it can be seen from Figs. 7 to 9, the typical convergence histories for CSS and ACO show that these two methods act similarly to some extent, and move toward their optimum in relatively the same way. However, the outcomes suggest that CSS has achieved better results in a shorter time, compared to ACO and other metaheuristc algorithms. In fact, in many instances for which the factor of time is more important, one may achieve relatively better results by the CSS algorithm.

The algorithm of this paper can also be applied to nodal ordering of other systems such as hydraulic and electrical systems. This application can also be extended to nodal numbering of finite element models using other graphs and bandwidth reduction of equilibrium equations [21-23].

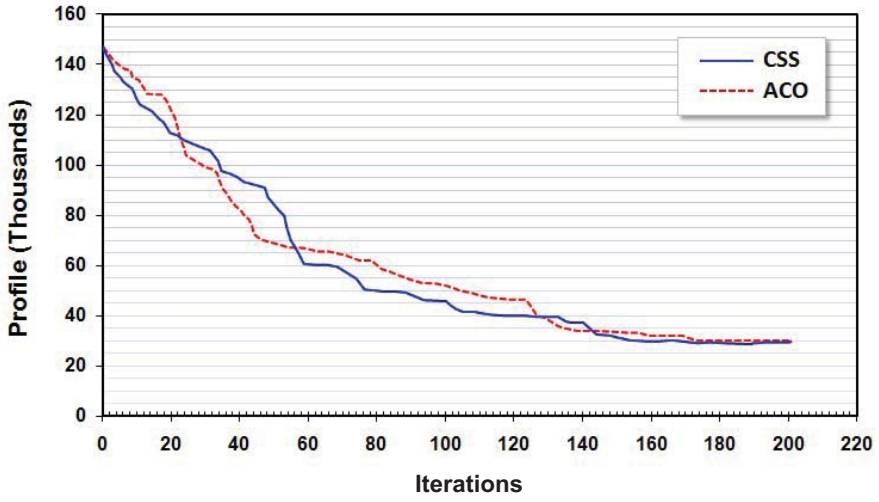

Fig. 7. The convergence history of example 1 for the CSS and ACO algorithms

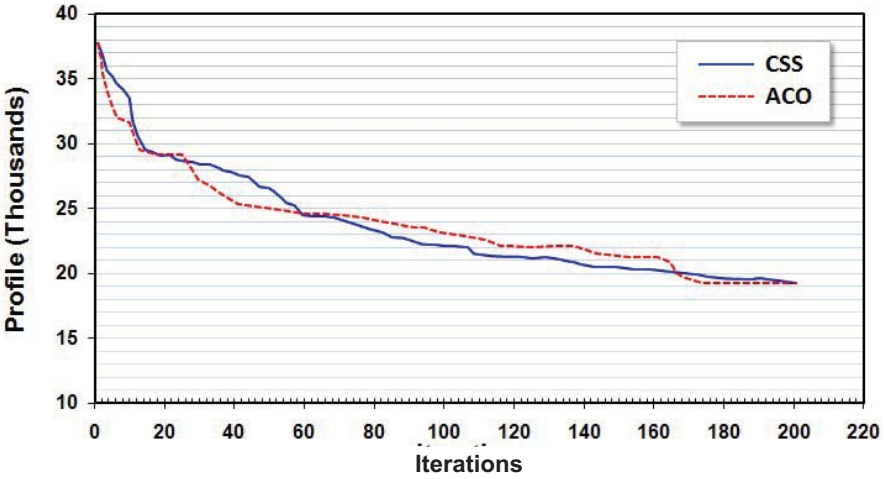

Fig. 8. The convergence history of example 2 for the CSS and ACO algorithms

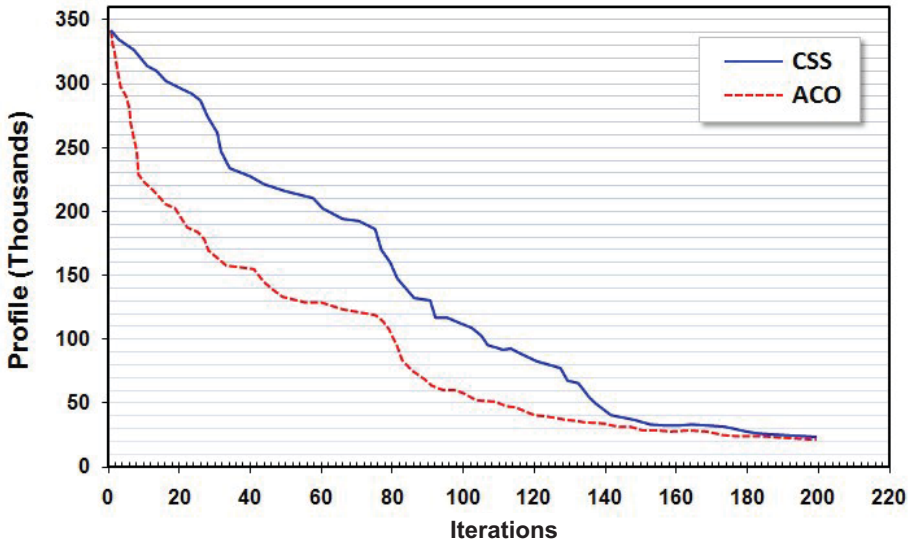

Fig. 9. The convergence history of example 3 for the CSS and ACO algorithms

***Acknowledgement*:** The first author is grateful to Iran National Science Foundation for the support.

REFERENCES

1. Kaveh, A. (2004). *Structural mechanics: Graph and matrix methods*. Research Studies Press, 3rd edition, Somerset, UK.
2. Kaveh, A. (2006). *Optimal structural analysis*. John Wiley, 2nd edition, Chichester, UK.
3. Cuthill, E. & McKee, J. (1969) Reducing the bandwidth of sparse symmetric matrices, Proc. *24th National Conference ACM, Bradon System Press, NJ*, pp. 157-172.
4. Souza, L. T. & Murray, D. W. (1993). An alternative pseudo-peripheral node finder for resequencing schemes. *International Journal for Numerical Methods in Engineering,* Vol. 36, pp. 3351-3379.
5. Sloan, S. W. (1986). An algorithm for profile and wavefront reduction of sparse matrices. *International Journal for Numerical Methods in Engineering*, Vol. 23, pp. 1693-1704.
6. Gibbs, N. E., Poole, W. G. & Stockmeyer, P. K. (1976). An algorithm for reducing the bandwidth and profile of a sparse matrix. *SIAM Journal of Numerical Analysis*, Vol. 12, pp. 236-250.
7. Kaveh, A. (1986). Ordering for bandwidth reduction, *Computers and Structures*. Vol. 24, pp. 413-420.
8. Kaveh, A. (1986). Multiple use of a shortest route tree for ordering. *Communcations in Numerical Methods in Engineering*, Vol. 2, pp. 213-215.
9. Kaveh, A. & Rahami, H. (2004). A new spectral method for nodal ordering of regular space structures. *Finite Elements in Analysis and Design*, Vol. 40, Nos. 13-14, pp. 1931-1945.
10. Kaveh, A. & Rahimi Bodarabady, H. (2002). A multi-level finite element nodal ordering using algebraic graph theory. *Finite Elements in Analysis and Design*, Vol. 38, No. 3, pp. 245-261.
11. Kaveh, A. & Shakouri, A. (2010). Harmony search algorithm for optimum design of slab formwork. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 34, No. B4, pp. 335-351.
12. Kaveh, A., & Malakouti Rad, S. (2010). Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 34, No. B1, pp. 15-34.
13. Jalali, M. R., Afshar, A. & Marino, M. A. (2006). Reservoir operation by ant colony optimization algorithms. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 30, No. B1, pp. 107-117.

14. Kaveh, A. & Talatahari, S. (2010). A novel heuristic optimization method: Charged system search, *Acta Mechanica*, Vol. 213, Nos. 3-4, pp. 267-286.
15. Kaveh, A. & Talatahari, S. (2010). Optimal design of truss structures via the charged system search algorithm. *Structural Multidisciplinary Optimization*, Vol. 37, No. 6, pp. 893-911.
16. Halliday, D., Resnick, R. & Walker, J. (2008). *Fundamentals of physics*. 8th Edition, John Wiley and Sons, USA.
17. King, I. P. (1970). An automatic reordering scheme for simultaneous equations derived from network systems. *International Journal for Numerical Methods in Engineering*, Vol. 2, pp. 523-533.
18. Kaveh, A. & Sharafi, P. (2007). A simple ant algorithm for profile optimization of sparse matrices. *Asian Journal of Civil Engineering*, Vol. 9, pp. 35-46.
19. Kaveh, A. & Sharafi, P. (2008). Optimal priority functions for profile reduction using ant colony optimization. *Finite Elements in Analysis and Design*, Vol. 44, No. 3, pp. 131-143.
20. Kaveh, A. & Sharafi, P. (2008). Nodal ordering for bandwidth reduction using ant system algorithm. *Engineering Computation,* Vol. 26, No. 3, pp. 313-337.
21. Kaveh, A. & Roosta, G. R. (1998). Comparative study of finite element nodal ordering methods. *Engineering Journal*, Vol. 20, Nos. 1&2, pp. 86-96.
22. Topçu, A. (1979). A contribution to the systematic analysis of finite element structures using the force method (in German). *Doctoral Dissertation*, Essen University, Germany.
23. Kaneko, I., Lawo, M. & Thierauf, G. (1982). On computational procedures for the force methods. *International Journal for Numerical Methods in Engineering*, Vol. 18, pp. 1469-1495.