Sizing Optimization of Structures by an Ant Colony Optimization Algorithm

Pruettha Nanakorn, Konlakarn Meesomklin, Chantima Somprasert and Nont Nimityongskul
Sirindhorn International Institute of Technology, Thammasat University
PO Box 22, Thammasat-Rangsit Post Office
Pathumthani 12121, Thailand
Phone (662) 986-9009 Ext. 1906
Fax (662) 986-9009 Ext. 1900
E-Mail: nanakorn@siit.tu.ac.th

Abstract

This paper presents an ant colony optimization algorithm for sizing optimization of structures. Recently, a heuristic optimization technique called ant colony optimization (ACO) has been developed for combinatorial optimization problems. The main concept of the technique is inspired by the way ant colonies function in the real world. An ant colony can collectively perform complicated tasks even with a low intelligence level of each individual ant. This complex colony-level behavior is obtained via interactions among individual ants which are achieved by using a chemical substance called pheromone. The ACO technique has been used in a few types of optimization problem with satisfactory results. In this study, the technique is applied to solve sizing optimization problems of structures. To this end, the structural optimization problems being considered have to be formulated in a suitable way that fits the ACO technique. Since the basic concept of the ACO is simple, the proposed ACO algorithm can be easily implemented. The validity of the algorithm in solving structural sizing optimization problems is investigated by solving sizing optimization problems of truss structures. The obtained results positively show the effectiveness of the proposed ACO algorithm in solving structural sizing optimization problems.

1. Introduction

Ant colonies can collectively perform complicated tasks even with a low intelligence level of each individual ant. One of the examples is the foraging behavior of ants. An ant colony is capable of finding the shortest path between its nest and a food source without using visual clues. This capability of the colony is achieved by indirect communication between ants via the use of pheromone. It is well-known that ants lay and follow pheromone trails. These simple trail-laying and trail-following mechanisms enable the colony to seek out the shortest paths.

Consider a colony of ants shown in Fig.1. In the figure, it is assumed that there are two available paths of different distances between the colony's nest and a food source from which ants may select. At the beginning, the ants will select the two paths with equal probability, meaning that there will be approximately half of the ants selecting each path. Since the shorter path requires less time to complete, for the same amount of time, the ants on the shorter path will be able to complete more rounds. As a result, the quantity of pheromone on the shorter path grows faster than on the longer one. Due to the shorter path's higher pheromone level, more ants will be probabilistically attracted to the shorter path and lay even more pheromone on this path. Finally, the levels of pheromone on the two paths will be so different that virtually all ants will select the shorter one.

It is important to note that pheromone trails established by ants do not last forever but rather they evaporate. This pheromone evaporation is also an important mechanism since it avoids too rapid a convergence towards a sub-optimal path. In other words, it allows a very good path that has not been discovered by ants until after a certain number of trips to overtake those moderately good paths that are discovered earlier. The three aforementioned mechanisms, i.e. pheromone-trail laying, pheromone-trail following and pheromone

RECEIVED 6 DECEMBER, 2002
ACCEPTED 6 FEBRUARY, 2003

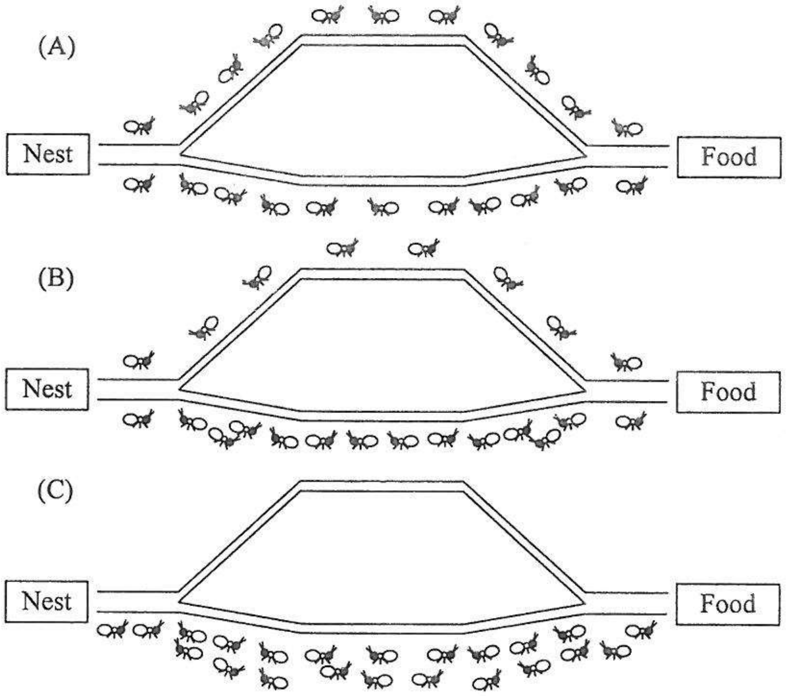


Fig. 1 (A) At the beginning, ants select the longer and shorter paths with equal probability.
(B) Pheromone is deposited faster on the shorter path. More ants select the shorter path.
(C) Finally, all ants select the shorter path.

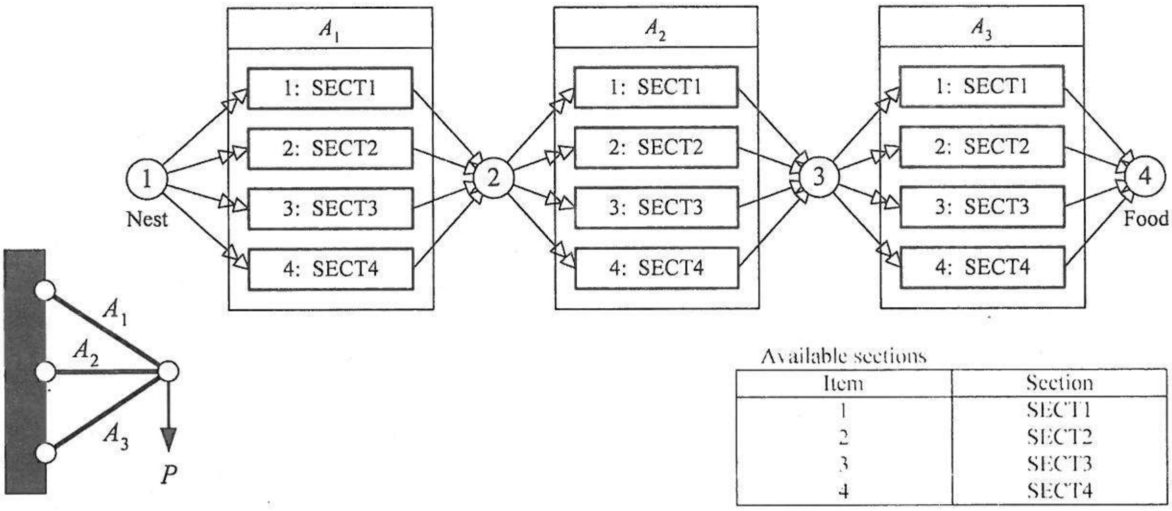
evaporation, can be artificially simulated by computers and constitute the Ant Colony Optimization (ACO) technique.

Recently, the ACO technique is becoming popular among researchers in the field of heuristic optimization (see, for example, a survey in [1]). The problem that seems to fit the technique naturally is the traveling salesman problem [2]. Nevertheless, the technique has been applied to various types of problem, such as the quadratic assignment problem [3], the just-in-time sequencing problem [4], optimization problems for designing and scheduling of batch plants [5], etc. The application of the technique in the field of civil engineering is still rare (see, for example, [6]).

The ACO technique has been developed for combinatorial optimization problems. Most of practical structural design optimization problems consider only sizing optimization, which is basically combinatorial optimization. In this study, the ACO technique is applied to solve structural sizing optimization problems. To this end, the structural sizing optimization problems under consideration have to be prepared in a suitable way that fits the ACO technique. After that, a simple ACO algorithm can be implemented. To show the validity and efficiency of the proposed algorithm, sizing optimization problems of two truss structures are solved. For comparison, one of the problems is also solved by a genetic algorithm. The results obtained by the proposed algorithm are also compared with those from the literature. Finally, the performance of the proposed algorithm is discussed.

2. Ant Colony Optimization for Sizing Optimization of Structures

To understand the concept of the ACO for sizing optimization of structures, consider Fig. 2 that shows a sizing optimization problem of a truss structure with three members. Since the truss has three members, there are three design variables, i.e. A_1, A_2 and A_3, which represent the areas of the three members. Assume that the section of each member is to be selected



Item	Section
1	SECT1
2	SECT2
3	SECT3
4	SECT4

Fig. 2 An ACO approach for sizing optimization of a three-bar truss.

from four available choices. As a result, the optimization becomes a problem of finding the combination of these sections that results in an optimal structure. The problem is a combinatorial optimization problem and can be thought of as a foraging problem of an ant colony. As shown in Fig. 2, an artificial nest and a food source can be established. In the figure, node 1 represents the nest and node 4 represents the food source. The ants will have to move from node 1 to node 4 by passing all other nodes in between. Between each pair of nodes, there are four available sub-paths, representing four different available sections for each design variable. The partial walk of the ants between nodes 1 and 2 represents the selection for the design variable A_1, and the partial walks between the subsequent nodes are for the subsequent design variables.

For the ACO to work, artificial ants will have to make many artificial tours and they must obey the following simple rules; i.e.

1) Ants will probabilistically select paths with higher levels of pheromone. In other words, paths with higher pheromone level will have higher chance to be selected by ants.
2) The amount of pheromone laid by an ant on the path which it has walked depends upon the quality of the path. If the path is of high quality, the ant that has walked the path will lay a large amount of pheromone on the path. For structural sizing optimization, a path is considered high quality if it represents an admissible structure with low weight.

These two rules, though simple, are enough for the colony to perform its task.

The first rule can be implemented by setting the probability of a sub-path being selected by an ant in the tour t as

$$p(A_i^a,t) = \frac{\tau(A_i^a,t)}{\sum_{k=1}^{S_i} \tau(A_i^k,t)}. \qquad (1)$$

Here, $p(A_i^a,t)$ denotes the aforementioned probability where A_i^a represents the a^{th} sub-path for the design variable i. In addition, S_i denotes the total number of available sub-paths for the design variable i. Finally, $\tau(A_i^a,t)$ denotes the amount of pheromone of the sub-path A_i^a in the tour t.

As an example, consider a partial walk between nodes 1 and 2 in the example in Fig. 2. The partial walk between nodes 1 and 2 is the selection for the design variable A_1. Between nodes 1 and 2, there are four available sub-paths; i.e. S_1=4. The probability of the sub-path

SECT3 (A_1^3) being selected by an ant, for example, can be written as

$$p(A_1^3,t) = \frac{\tau(A_1^3,t)}{\sum_{k=1}^{4}\tau(A_1^k,t)}. \qquad (2)$$

For the first tour where there is still no pheromone on any sub-paths, a random selection can be used.

To be able to implement the second rule of ants, it is necessary to define a parameter that represents the quality of the path. Since each path actually represents a design, the objective function value and the degree of constraint violation of each design must be considered in order to evaluate the quality of the design or path. For sizing optimization of structures, the following form of a quality function Q may be employed:

$$Q(\mathbf{A}) = Q_o(\mathbf{A}) - \lambda E(\mathbf{A}). \qquad (3)$$

Here, $\mathbf{A}$ denotes a design vector defined as

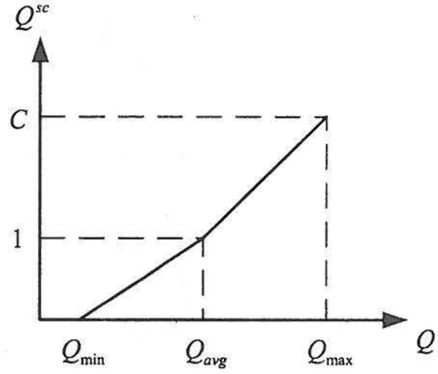
$$\mathbf{A} = [A_1, A_2, \ldots, A_n]^T \qquad (4)$$

where A_i represents the i^{th} design variable and n denotes the number of design variables. Moreover, $Q_o(\mathbf{A})$ is a function representing the basic quality value calculated from the objective function while $E(\mathbf{A})$ is a non-negative function representing the degree of constraint violation. In addition, λ denotes a user-defined positive constant.

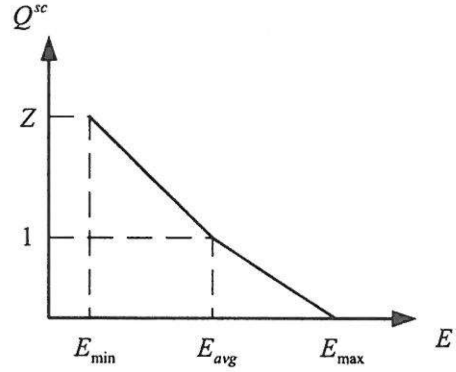
In order to have further control over the optimization, the quality function obtained in (3) will not be used directly. Rather, it will be scaled into a certain range. In this study, the bilinear scaling techniques shown in Fig. 3 will be used. In the figure, the scaling scheme A is used when, in the tour under consideration, there are some admissible designs selected by the ants and the scaling scheme B is used when all designs selected by the ants are inadmissible. The subscripts *min*, *max* and *avg* denote the minimum, maximum and average values, respectively. In addition, C and Z are user-defined constants. The scaled quality function Q^{sc} will be used in the subsequence calculation instead of Q.

By employing the general procedure for ACO algorithms (see, for example, [7]), the pheromone-trail-laying algorithm can be constructed. To this end, denote the design vector $\mathbf{A}$ selected by the ant Ant_j during the tour t as $\mathbf{A}(Ant_j,t)$ and define a function $\Delta\tau$ as

$$\Delta\tau(A_i^a, Ant_j, t) = \begin{cases} Q^{sc}[\mathbf{A}(Ant_j,t)]/n & \text{: if } Ant_j \text{ has traversed } A_i^a \text{ during the tour } t \\ 0 & \text{: otherwise.} \end{cases} \qquad (5)$$



(A) Some designs selected by ants are admissible.



(B) All designs selected by ants are inadmissible.

Fig. 3 Bilinear scaling for the quality function.

Note again that n denotes the number of the design variables.

Let

$$V(t)=\sum_{j=1}^{M}Q^{sc}[\mathbf{A}(Ant_j,t)] \qquad (6)$$

where M denotes the number of the ants. It can be seen that $V(t)$ is actually the summation of the value of $\Delta\tau$ in the tour t.

Next, let

$$\Delta\bar{\tau}(A_i^a,t)=\frac{\sum_{j=1}^{M}\Delta\tau(A_i^a,Ant_j,t)}{V(t)}. \qquad (7)$$

Finally, define the pheromone-updating scheme as

$$\tau(A_i^k,t+1)=\Delta\bar{\tau}(A_i^k,t) \qquad t=1$$
$$\tau(A_i^k,t+1)=(1-\rho)\tau(A_i^k,t)+\rho\Delta\bar{\tau}(A_i^k,t) \qquad t\geq 2 \qquad (8)$$

where ρ denotes the evaporation factor. This factor is used to control the evaporation rate. Actually, it can be seen from (8) that, in the first tour, there is no pheromone evaporation. From the second tour, the evaporation is implemented by the term $(1-\rho)\tau(A_i^k,t)$ while the pheromone laying is implemented by the term $\rho\Delta\bar{\tau}(A_i^k,t)$. It can be seen from (5)-(8) that the sum of pheromone values of all sub-paths remains equal to 1 in all tours.

In the calculation, identification of the best obtained design of all tours is naturally required. For a design to be acceptable, it must be at least admissible. Therefore, the quality function defined in (3) cannot directly be used for the purpose of finding the best admissible design and a new rule of comparison must be employed. In the algorithm, finding the best admissible design is actually equivalent to finding the ant which selects that design. To find the best ant from all available ants in the calculation, the following rule of comparison is defined; i.e. Ant_i is considered to be better than Ant_j when

1) Ant_i selects an admissible design while Ant_j selects an inadmissible design, or
2) Both Ant_i and Ant_j select admissible designs but Ant_i's design has a larger Q_o, or
3) Both Ant_i and Ant_j select inadmissible designs but Ant_i's design has a smaller E.

Finally, the complete algorithm can be summarized as

```
Tour=1;
All_Ants_Select_Paths(Random);
Calculate_Paths_Quality( );
Find_The_Best_Ant_of_The_Tour( );
Update_The_Best_Ant_of_All_Tours( );
All_Ants_Lay_Pheromone( );
For Tour=2 to N
{
   All_Ants_Select_Paths(Pheromone_Based);
   Calculate_Paths_Quality( );
   Find_The_Best_Ant_of_The_Tour( );
   Update_The_Best_Ant_of_All_Tours( );
   Pheromone_Evaporation( );
   All_Ants_Lay_Pheromone( );
}
```

3. Results

To investigate the validity and efficiency of the proposed ACO algorithm, two numerical examples are solved. They are sizing optimization problems of two truss structures, i.e. a six-bar truss and a ten-bar truss. To be able to clearly see the advantages of the proposed algorithm, the obtained results of the six-bar truss problem are compared with those obtained by a genetic algorithm (GA). A genetic algorithm is selected for comparison because it is well accepted that genetic algorithms are currently one of the best optimization techniques available. The comparison of the results from these two techniques includes not only the quality but also the uniformity of the results. In this way, the actual performance of the proposed

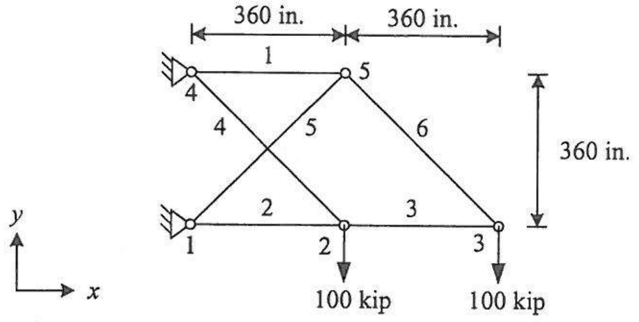


Fig. 4 Six-bar truss.

Table 1 Design parameters for the six-bar truss problem.

Item	Value
Modulus of elasticity	10^7 psi
Weight density	0.1 lb/in.^{3}
Allowable tensile stress	25,000 psi
Allowable compressive stress	25,000 psi
Maximum *y*-displacement	2 in.

algorithm can really be discussed. Finally, the results are also compared with those from the literature.

3.1 Six-Bar Truss

The first problem to be considered is the six-bar truss shown in Fig. 4. Since only sizing optimization is considered, design variables are six sectional areas of the six members of the truss. The cross-sectional area of each member is taken from the following 32 discrete values, i.e. 1.62, 1.80, 2.38, 2.62, 2.88, 3.09, 3.13, 3.38, 3.63, 3.84, 3.87, 4.18, 4.49, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 13.9, 14.2, 15.5, 16.0, 18.8, 19.9, 22.0, 22.9, 26.5, 30.0, and 33.5 in.^{2} There are two types of constraint in this problem, i.e. stress and displacement constraints. Design parameters used in the problem are shown in Table 1. Since the allowable stress values for tension and compression are the same, it implies that buckling behavior is not considered. Note that, in the calculation of stress and displacement responses, only the two point loads shown in Fig. 4 are considered as the applied forces whereas the weight of the structure is neglected.

Table 2 ACO parameters for the six-bar truss problem.

Item	Value
Number of ants	100 and 300
Number of tours	100
ρ	0.3
λ	0.0002
C	2
Z	5

Table 3 GA parameters for the six-bar truss problem.

Item	Value
Population size	100 and 300
Number of generations	100
Crossover probability	0.85
Mutation probability	0.05
λ	0.0002
C	2
Z	5

In this problem, the basic quality function and constraint-violation function are defined as

$$Q_o(\mathbf{A}) = \frac{1}{1+Weight}, \qquad (9a)$$

$$E(\mathbf{A}) = \sum_{i=1}^{NE} \max\left(\frac{|\sigma_i| - \sigma_a}{\sigma_a}, 0\right) + \sum_{i=1}^{NN} \max\left(\frac{|v_i| - v_a}{v_a}, 0\right). \qquad (9b)$$

Here, the unit of weight used is pound. In addition, σ and v represents the stress and *y*-displacement, respectively. The subscript *a* denotes the allowable values. Moreover, *NE* and *NN* represent the number of elements and the number of nodes, respectively.

Table 2 shows ACO parameters used in this problem. For comparison, two sets of calculations with different numbers of ants are performed. For each set, 200 runs are carried out. The reason why many runs are required is that the ACO technique includes probabilistic processes. As a result, even with the same problem and the same calculation parameters,

Table 4 Comparison of the results obtained by the proposed algorithm and the genetic algorithm for the six-bar truss problem.

Method		Minimum weight of the solutions of 200 runs (lb)	Maximum weight (lb)	Average weight (lb)	Standard deviation of weights (lb)
Proposed	Number of ants				
	100	4962.1	5199.1	4992.4	36.1
	300	4962.1	5003.9	4964.8	6.60
GA	Population size				
	100	4962.1	5298.3	5031.1	68.4
	300	4962.1	5028.1	4971.9	15.1

different results may be obtained from different runs. Many test runs will allow the efficiency of the technique, in terms of the result quality as well as the result uniformity, to be discussed.

As mentioned earlier, the problem is also solved by using a genetic algorithm. Here, a standard genetic algorithm will be used. The quality function for the ACO algorithm defined in (3), with its components defined in (9), is used as the fitness function in the genetic algorithm. In addition, the bilinear scaling techniques shown in Fig. 3 are also used for fitness scaling in the genetic algorithm (for more information related to genetic algorithms, see, for example, [8-10]). Similar to the ACO calculation, two sets of calculations with different population sizes are performed in the GA calculation and, for each set, 200 runs are carried out. Note that genetic algorithms also contain probabilistic processes, and to investigate the efficiency of genetic algorithms, many runs are required. Table 3 shows GA parameters used in this problem.

Define the best design as an admissible design with the minimum weight. Consequently, the solution of a run is defined as the best design ever found in that run even though it may not be the best design of the last tour (ACO) or the last generation (GA) of the run. For each set of calculations, after the 200 solutions of its 200 runs are obtained, the minimum, maximum, average and standard deviation values of the weights of the 200 solutions are found. Among the 200 solutions, the solution with the minimum weight is naturally the best solution for that particular set of calculations.

Table 4 shows the comparison of the results obtained by the proposed algorithm and the genetic algorithm. From the results of the ACO algorithm, it can be seen that the best solutions obtained with 100 and 300 ants are the same. Nevertheless, the average and worst solutions are improved rather significantly when the number of ants is increased from 100 to 300. In addition, the standard deviation of the weights of the results also decreases drastically when 300 ants are used. This indicates that the quality of the results obtained with more ants is more consistent. The genetic algorithm also gives the same best solution regardless of the population size. Moreover, increasing the population size also increases the quality of the average and worst solutions. Nonetheless, the average and worst solutions from the genetic algorithm are inferior to those from the ACO algorithm (see Table 4). Moreover, the standard deviations of the weights of the results of the genetic algorithm are larger than those of the ACO algorithm. This clearly shows that the ACO algorithm yields results that are more uniform than the GA results.

The best result obtained from the proposed ACO algorithm is compared with results reported in the literature in Table 5. The results from the literature are obtained by genetic algorithms. It can be seen that these results are exactly the same. Table 6 shows the constrained displacements and stresses of this best solution. It can be clearly seen that the

Table 5 Comparison of the results with the literature for the six-bar truss problem.

Member	Size of member (in.^{2})		
	Proposed	Genetic Algorithms: Rajan [8]	Genetic Algorithms: Nanakorn and Meesomklin [9]
1	30.0	30.0	30.0
2	19.9	19.9	19.9
3	15.5	15.5	15.5
4	7.22	7.22	7.22
5	22.0	22.0	22.0
6	22.0	22.0	22.0
Total weight (lb)	**4962.1**	**4962.1**	**4962.1**

Table 6 Constrained displacements and stresses of the best solution (with the total weight of 4962.1 lb)

Node/Element	y-displacement (in.)	Stress (ksi)
1	0	6.67
2	-1.77	-10.1
3	**-2.00**	-6.45
4	0	19.6
5	-0.701	-6.43
6	-	6.43

critical constraint is the y-displacement constraint.

3.2 Ten-Bar Truss

The next problem to be considered is the ten-bar truss shown in Fig. 5. This problem is one of the benchmark problems used to test structural optimization methods. Also in this problem, only sizing optimization is considered. Therefore, design variables are ten sectional areas. The cross-sectional areas of members 1, 3, 4, 7, 8 and 9 are taken from the following 32 discrete values, i.e. 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 13.9, 14.2, 15.5, 16.0, 16.9, 18.8, 19.9, 22.0, 22.9, 26.5, 30.0, and 33.5 in.^{2} For the rest of the members, the cross-sectional areas are taken from the following 32 discrete values, i.e. 1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 13.9, and 14.2 in.^{2} The constraints considered in this problem are also stress and displacement constraints. The design parameters and ACO parameters are shown in Tables 7 and 8. Similar to the previous problem, in the calculation of stress and displacement responses, only the two point loads shown in Fig. 5 are considered as the applied forces whereas the weight of the structure is neglected. The components of the quality function shown in (9) are also used for this problem. However, in this problem, the constraint on the horizontal displacement has to be added to (9b).

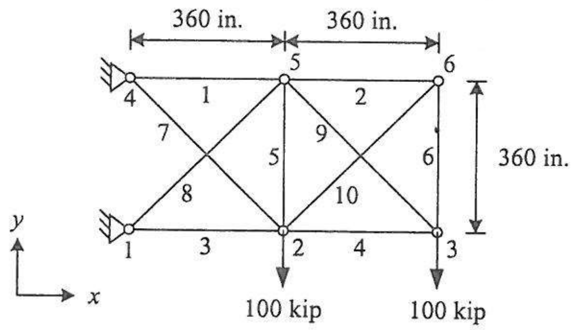


Fig. 5 Ten-bar truss.

Table 7 Design parameters for the ten-bar truss problem.

Item	Value
Modulus of elasticity	10^7 psi
Weight density	0.1 lb/in.^{3}
Allowable tensile stress	25,000 psi
Allowable compressive stress	25,000 psi
Maximum x- and y-displacements	2 in.

Table 8 ACO parameters for the ten-bar truss problem.

Item	Value
Number of ants	500
Number of tours	200
ρ	0.3
λ	0.0002
C	2
Z	5

Table 9 Comparison of the results with the literature for the ten-bar truss problem.

Member	Size of member (in.^{2})				
		Genetic Algorithms			
	Proposed	Nanakorn and Meesomklin [9]	Rajeev and Krishnamoorthy [10]	Camp et al. [11]	Galante [12]
1	30.0	33.5	33.5	30.0	33.5
2	1.80	1.62	1.62	1.62	1.62
3	26.5	22.9	22.0	26.5	22.0
4	14.2	15.5	15.5	13.5	14.2
5	1.62	1.62	1.62	1.62	1.62
6	1.62	1.62	1.62	1.62	1.62
7	11.5	7.22	14.2	7.22	7.97
8	19.9	22.9	19.9	22.9	22.9
9	22.0	22.0	19.9	22.0	22.0
10	1.80	1.62	2.62	1.62	1.62
Total weight (lb)	**5537.0**	5499.3	5613.8	5556.9	**5458.3**

Table 10 Constrained displacements of the results for the ten-bar truss problem.

Node	Displacement (in.) Proposed		Nanakorn and Meesomklin [9]		Rajeev and Krishnamoorthy [10]		Camp et al. [11]		Galante [12]	
	x-disp	y-disp	x-disp	y-disp	x-disp	y-disp	x-disp	y-disp	x-disp	y-disp
1	0	0	0	0	0	0	0	0	0	0
2	-0.261	-1.07	-0.277	-1.35	-0.319	-0.998	-0.242	-1.34	-0.292	-1.29
3	-0.504	**-2.00**	-0.506	**-2.00**	-0.538	**-2.00**	-0.504	**-2.03**	-0.541	**-2.01**
4	0	0	0	0	0	0	0	0	0	0
5	0.250	-0.803	0.241	-0.792	0.221	-0.759	0.267	-0.810	0.238	-0.778
6	0.334	-1.91	0.267	-1.97	0.344	-1.88	0.305	**-2.00**	0.277	-1.97

Table 11 Constrained stresses of the results for the ten-bar truss problem.

Element	Stress (ksi) Proposed	Nanakorn and Meesomklin [9]	Rajeev and Krishnamoorthy [10]	Camp et al. [11]	Galante [12]
1	6.94	6.68	6.13	7.40	6.61
2	2.33	0.740	3.43	1.07	1.08
3	-7.24	-7.69	-8.85	-6.71	-8.11
4	-6.75	-6.37	-6.09	-7.28	-6.92
5	7.56	15.5	6.65	14.7	14.4
6	2.59	0.740	3.43	1.07	1.08
7	11.3	14.9	9.44	15.3	13.9
8	-7.68	-7.65	-7.48	-7.54	-7.50
9	6.16	6.35	6.71	6.32	6.32
10	-3.29	-1.05	-3.00	-1.52	-1.52

Similar to the previous problem, 200 runs are carried out with the proposed ACO algorithm. The best result of the 200 runs is compared with results reported in the literature in Table 9. The results from the literature are obtained by genetic algorithms. It can be seen that the quality of the result obtained from the proposed algorithm is comparable with that of the results from the literature although some results from the literature are better. However, if the results from the literature are carefully investigated, it is found that some of these results actually violate the given displacement constraint. Tables 10 and 11 show the displacements and stresses of the results in Table 9. As in the previous problem, the critical constraint is the *y*-displacement constraint. In this study, the constraints are strictly enforced and only admissible results can be considered as solutions. If those results in Table 9 that are inadmissible are excluded from the comparison, the quality of the result from the proposed algorithm and the quality of the results from the literature become even more indistinguishable. In addition, it must be noted that since the ACO algorithm used in this study is a simple one, even better results may be expected when a more sophisticated ACO algorithm is employed.

4. Conclusions

In this study, an algorithm based on the ACO for sizing optimization of structures is proposed. The ACO mimics the way ant colonies function in the real world. An ant colony is capable of finding the shortest path between its nest and a food source by the use of pheromone. The task of finding the shortest path is achieved by using three basic mechanisms, i.e. pheromone-trail laying, pheromone-trail following and pheromone evaporation. These three mechanisms comprise the ACO and can be simulated by computers. In this study, sizing optimization problems of structures are first formulated in a suitable way that fits the ACO approach. To this end, the optimization problem under consideration is transformed into a foraging problem of an ant colony. Each design solution is interpreted as a route that ants can use to walk from the colony's nest to a food source. A better design is made to be equivalent to a shorter route. The proposed algorithm is used to solve sizing optimization problems of truss structures. The comparison with the results obtained by a standard genetic algorithm shows that, in terms of the result quality, the performance of the ACO and the performance of the genetic algorithm are comparable. Nevertheless, in terms of the result uniformity, the ACO evidently outperforms the genetic algorithm. In addition, it is also found that the results from the proposed ACO algorithm are as good as the best results found in the literature. It is expected that better performance of the algorithm can be achieved if schemes that are more sophisticated are included in the algorithm.

5. Acknowledgements

The authors are grateful to the Thailand Research Fund for providing financial support to this study.

References

[1] M. Dorigo, G.D. Caro and L.M. Gambardella, "Ant Algorithms for Discrete Optimization", Artificial Life, Vol. 5, No. 2, 1999, pp. 137-172.

[2] M. Dorigo and L.M. Gambardella, "Ant Colonies for the Travelling Salesman Problem", BioSystems, Vol. 43, 1997, pp. 73-81.

[3] E.-G. Talbi, O. Roux, C. Fonlupt and D. Robillard, "Parallel Ant Colonies for the Quadratic Assignment Problem", Future Generation Computer Systems, Vol. 17, 2001, pp. 441-449.

[4] P.R. McMullen, "An Ant Colony Optimization Approach to Addressing a JIT Sequencing Problem with Multiple Objectives", Artificial Intelligence in Engineering, Vol. 15, 2001, pp. 309-317.

[5] V.K. Jayaraman, B.D. Kulkarni, S. Karale and P. Shelokar, "Ant Colony Framework for Optimal Design and Scheduling of Batch Plants", Computers and Chemical Engineering, Vol. 24, 2000, pp. 1901-1912.

[6] K.C. Abbaspour, R. Schulin and M.Th. van Genuchten, "Estimating Unsaturated Soil Hydraulic Parameters Using Ant Colony Optimization", Advances in Water Resources, Vol. 24, 2001, pp. 827-841.

[7] W.J. Gutjahr, "A Graph-Based Ant System and Its Convergence", Future

Generation Computer Systems, Vol. 16, 2000, pp. 873-888.

[8] S.D. Rajan, "Sizing, Shape, and Topology Design Optimization of Trusses Using Genetic Algorithm", Journal of Structural Engineering, Vol. 121, No. 10, 1995, pp. 1480-1487.

[9] P. Nanakorn and K. Meesomklin, "An Adaptive Penalty Function in Genetic Algorithms for Structural Design Optimization", Computers and Structures, Vol. 79, 2001, pp. 2527-2539.

[10] S. Rajeev and C.S. Krishnamoorthy, "Discrete Optimization of Structures Using Genetic Algorithms", Journal of Structural Engineering, Vol. 118, No. 5, 1992, pp. 1233-1249.

[11] C. Camp, S. Pezeshk and G. Cao, "Optimized Design of Two-dimensional Structures Using a Genetic Algorithm", Journal of Structural Engineering, Vol. 124, No. 5, 1998, pp. 551-559.

[12] M. Galante, "Genetic Algorithms as an Approach to Optimize Real-World Trusses", International Journal for Numerical Methods in Engineering, Vol. 39, 1996, pp. 361-382.