Site Layout of Temparory Construction Facilities using Ant Colony Optimization.

Saurabh Arun Samdani,[1] Lalit Bhakal [2] and Arvind Kumar Singh [3]

ABSTRACT

The positioning of temporary facilities on a construction site is an area of research which has been recognised as important but which has received relatively little attention. In this paper, an Ant Colony Optimization (ACO) algorithm is proposed to solve the problem in which n facilities are to be positioned to n available sites such that the total cost of construction and interactive cost due to facility layout constraints is minimized. A formulation of the problem in terms of construction graph to be used in ACO is presented. Experiments on a sample problem suggest that the technique will prove useful when tackling real problems.

Keywords: Ant colony optimization, Site layout planning

INTRODUCTION

Site layout planning consists of identifying the facilities needed to support consruction operations, determining their size and shape, and positioning them within the boundaries of the available on-site or remote areas (Tommelein et al. 1992). Despite the importance of site space as a resource, site-layout planning is often neglected, and the attitude of engineers has been that it will be done as the project progresses. Good site layout, however, is important to promote safe and efficient operations, minimize travel time, decrease material handling, and avoid obstructing material and equipment movements, especially in the case of large projects (Hegazy and Elbeltagi 1999). In addition, such a problem becomes far from trivial if a construction site is confined due to the lack of available space, or if the site is very large, then traveling between facilities can be considerably time consuming (Li and Love 1998). Aim of construction site layout planning is to find convenient an feasible locations for different temporary facilities.

When temporary site level facilities are required to be located on a construction site, the locations of buildings to be constructed are assumed to be known. These locations are used to define available sites for temporary facilities. Then the problem can be defined as allocation of predetermined facilities like warehouses, job offices, workshops and batch plants so as to optimize an objective subject to layout constraints and requirements. Using such a definition of the problem, formulation in terms of a combinatorial optimization problem has been attempted. The vector of decision variables is the location of all the n facilities with the constraints being the uniqueness of location and site for a particular facility. The objective is typically to minimize sum of product of resource flow(considered in a generic sense) between two *facilites* and distance between the *locations* of the two facilites. The

[1]Undergraduate Student, Department of Civil Engineering , Indian Institute of Technology, Guwahati, Email: samdani@iitg.ernet.in

[2]Graduate Student, University of Illinois at Urbana-Champaign Email: lbhakal2@uiuc.edu

[3]Assistant Professor, Department of Civil Engineering , Indian Institute of Technology, Guwahati Email: arvind@iitg.ernet.in

problem is a special case of the Quadratic Assignment Problem (QAP), which is one of the most difficult $\mathcal{NP}-$ complete problems. Researchers have attempted various heuristic based techniques and approximation algorithms to solve the problem. In literature, some of the solution techniques used were Simulated Annealing, Neural Networks (Yeh 1995), Genetic Algorithms (Hegazy and Elbeltagi 1999; Cheung et al. 2002; Li and Love 1998) and other AI–based techiniques.

In this work, a solution to the problem is attempted using Ant Colony Optimization(ACO) algorithm (Dorigo et al. 1996; Dorigo and Stützle 2004), inspired by the foraging behavior of real ants. Formulation of site layout problem in terms of a solution construction graph to be used by (artificial) ants has been attempted. Artificial ants perform random and guided walks on this construction graph to find feasible and high quality solutions. The allowed walks on this graph produce only feasible solutions. Previous researchers have used penalty functions and other techniques to repair infeasible solutions. In the present model no such *exterior* constraint handling technique has been used. This leads to a considerable improvement in compuational efficiency as infeasible solutions are not generated at all. Efficacy of the model has been demonstrated with a case study of a 12 facility problem.

SITE LAYOUT PROBLEM FORMULATION

A set of facilities needs to be located on the site while satisfying the layout constraints and optimizing the layout objectives. There are n facilities given along with n locations on the site. So the vector of decision variables is the location of all the facilities, $\mathcal{L} = \{l_1, l_2, \ldots, l_n\}$. Let C_{xi} be the cost of constructing a facility x on site i and D_{xy} be the interactive cost of assigning facility x on the site neighboring facility y. A_{ij} is the site adjacency index matrix i.e.

$$\begin{aligned} A_{ij} &= 1 \quad \text{if site } \ i \ \text{ adjacent to site } \ j \\ A_{ij} &= 0 \qquad\qquad\quad \text{otherwise} \end{aligned}$$

The layout objective considered in this work is

$$f(\mathcal{L}) = \sum_x \sum_i \delta_{xi} C_{xi} + \sum_x \sum_y \sum_i \sum_j \delta_{xi} \delta_{yj} A_{ij} D_{xy} \tag{1}$$

where δ_{xi} is the permuatation matrix variable.

$$\delta_{xi} = \begin{cases} 1 & \text{if } l_i = x \\ 0 & \text{otherwise} \end{cases}$$

$\sum_x \sum_i \delta_{xi} C_{ij}$ represents the total cost of constructing all the facilities on their assigned locations, while $\sum_x \sum_y \sum_i \sum_j \delta_{xi} \delta_{yj} A_{xy}$ represents the total interactive cost between the facilities. Since one facility should be allocated to only one site,

$$\sum_i \delta_{xi} = 1, \forall x = 1, 2, \ldots, n. \tag{2}$$

and one site can accomodate only one facility

$$\sum_x \delta_{xi} = 1, \forall i = 1, 2, \ldots, n. \tag{3}$$

The problem now can be stated as

$$\text{minimize } f(\mathcal{L}) = \sum_{x}\sum_{i}\delta_{xi}C_{xi} + \sum_{x}\sum_{:}\sum_{i}\sum_{j}\delta_{xi}\delta_{yj}A_{ij}D_{xy} \tag{4}$$

subject to

$$\sum_{x}\delta_{xi} = 1, \forall i = 1, 2, \ldots, n. \tag{5}$$

$$\sum_{i}\delta_{xi} = 1, \forall x = 1, 2, \ldots, n. \tag{6}$$

$$1 \leq l_i \leq n, \forall i = 1, 2, \ldots, n. \tag{7}$$

We now present the basis of the ACO technique and the ACO algorithm.

REAL ANTS BEHAVIOUR

Ants are social insects, that is, insects that live in colonies and whose behavior is directed more to the survival of the colony as a whole than to that of a single individual component of the colony. Social insects have captured the attention of many scientists because of the high structuration level their colonies can achieve, especially when compared to the relative simplicity of the colony's individuals. An important and interesting behavior of ant colonies is their foraging behavior, and, in particular, how ants can find the shortest paths between food sources and their nest (Dorigo and Stützle 2004).

Although individual ants move in a quasi-random fashion, performing relatively simple tasks, the entire colony of ants can collectively accomplish sophisticated movement patterns and can find the shortest route between their nest and a food source. Ants accomplish this by depositing a substance called *pheromone* as they move. This chemical trail can be detected by other ants, which are probabilistically more likely to follow a path rich in pheromone. This chemical trail can be detected by other ants, which are probabilistically more likely to follow a path rich in pheromone.To show how trail information can be utilized to adapt to sudden unexpected changes in the terrain, a brief example is given next (Fig.1). In Fig.1a, a colony of ants is traveling in both directions between point A and point E. Each ant knows which direction to take because of the pheromone trail that is present from point A to point E (Fig.1a). Fig.1b shows what happens when an object is placed in the middle of the path. Since the object is not placed symmetrically on the trail, the path B-C-D is shorter than the path B-F-G-H-D. The ants moving from point B to point D and vice versa, will have to make a decision whether to turn left or right. Since there is no pheromone in either direction, the ant has an equal probability of turning right or left. Initially the first ants turn left or right equally, which means that the equal number of ants are taking path B-C-D and path B-F-G-H-D. The ant traveling along any path will leave pheromone along it. The ants traveling the B-C-D path will arrive at point D earlier than the ones traveling along path B-F-G-H-D. An ant traveling in the opposite direction and is at point D will detect more pheromone along the D-C path, since not only are ants going from D to C, but ants also have started to arrive from C to D. On the other hand the path B-F-G-H-D being longer, the ants have not yet started arriving on path D-H. The exact same thing is occurring at point B. Therefore due to more pheromone deposition, probabilistically more ants will begin taking path D-C-B.

Eventually the pheromone level on the D-C-B path will become so dominant that all of the ants will choose this path as in Fig.1c. This will also hold true for the ants traveling from point B to D. Hence the ants, using their highly effective pheromone based communication method, are able to find the shortest path between point B and D.

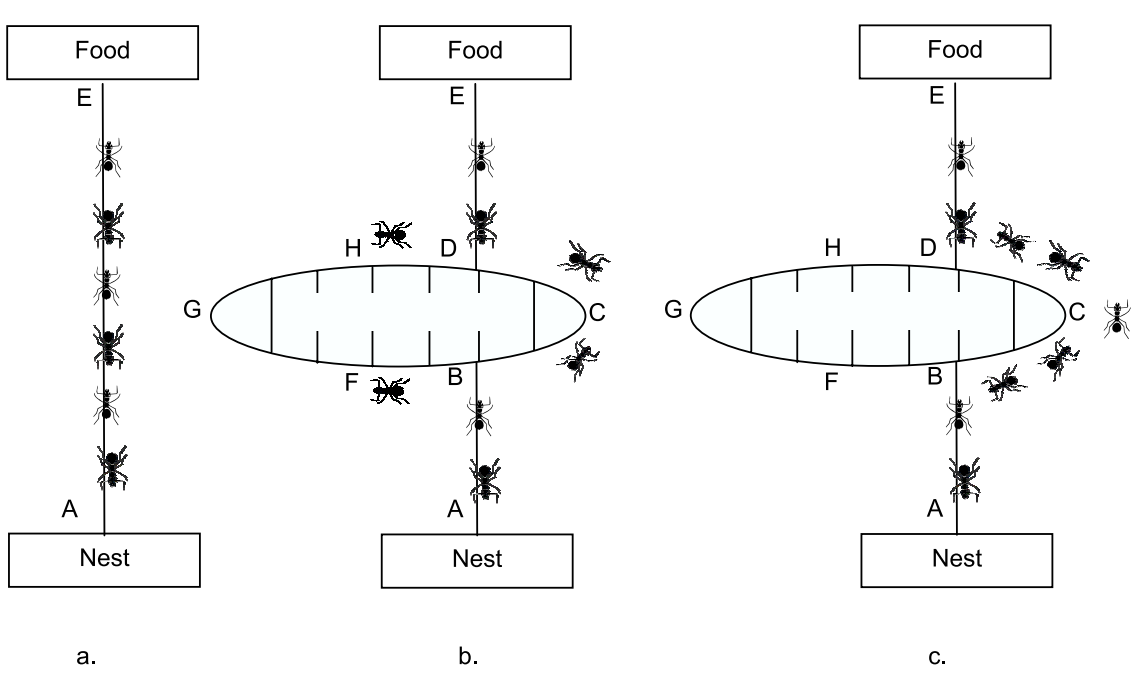

FIG. 1. Behaviour of biological ants.

This particular behaviour of ant colonies has inspired the Ant Colony Optimization (ACO) meta-heuristic algorithm, in which a set of artificial ants co-operate to find solutions to a given optimization problem by depositing pheromone trails throughout the search space.

ANT COLONY OPTIMIZATION

ACO algorithms are metaheuristic methods for tackling combinatorial optimization problems (Dorigo and Stützle 2004). The central component of an ACO algorithm is the pheromone model, which is used to probabilistically sample the search space. As outlined in Blum and Dorigo (2004), the pheromone model can be derived from a model of the Combinatorial Optimization (CO) problem under consideration. A model of a CO problem for the site layout problem can be stated as follows.

Definition 1 *A model $P = (S, \Omega, f)$ of a CO problem consists of:*

- *a **search (or solution) space** $\mathcal{S}$ defined over a finite set of discrete decision variables and a set Ω **constraints** among the variables;*
- *an **objective function** $f : \mathcal{S} \rightarrow \mathcal{R}^+$ to be minimized.*

*The search space $\mathcal{S}$ is defined as follows: Given is a set of n **discrete variables** $\mathcal{X}_i$ with domains $D_i = v_i^1, \cdots, v_i^{|Di|}, i = 1, \cdots, n$. A variable instantiation, that is, the assignment of*

a value v_i^j to a variable $\mathcal{X}_i$, is denoted by $\mathcal{X}_i = v_i^j$. A feasible solution $s \in \mathcal{S}$ is a complete assignment (i.e., an assignment in which each decision variable has a domain value assigned) that satisfies the constraints. If the set of constraints Ω is empty, then each decision variable can take any value from its domain independently of the values of the other decision variables. In this case we call $\mathcal{P}$ an ***unconstrained*** *problem model, otherwise a* ***constrained*** *problem model. A feasible solution $s^* \in \mathcal{S}$ is called a* ***globally optimal solution****, if $f(s^*) \leq f(s) \quad \forall s \in \mathcal{S}$ The set of globally optimal solutions is denoted by $\mathcal{S}^* \subseteq \mathcal{S}$ To solve a CO problem one has to find a solution $s \in \mathcal{S}^*$*

For the site layout problem decision variables $\mathcal{X}_i$ correspond to the location variables l_i, while the domain of each variable is the set $\{1, 2, \ldots, n\}$. The objective function is given by Eq. 4 and the set of constraints Ω is defined by Eqs. 5,6. A model of the CO problem under consideration implies the finite set of solution components and the pheromone model as follows (Blum and Dorigo 2004). First, we call the combination of a decision variable l_i and one of its domain values a *solution component* denoted by $\mathfrak{c}_i^j$. $\mathfrak{c}_i^j$ is nothing but the assignment of facility i to site j. Then, the pheromone model consists of a *pheromone* trail parameter $\mathcal{T}_i^j$ for every solution component $\mathfrak{c}_i^j$. The value of a *pheromone* trail parameter $\mathcal{T}_i^j$ called pheromone value – is denoted by τ_i^j. The set of all pheromone trail parameters is denoted by $\mathcal{T}$. As a CO problem can be modelled in different ways, different models of the CO problem can be used to define different pheromone models. Alg. 1 captures the framework of a basic ACO algorithm, as outlined in Blum and Dorigo (2004). It works as follows: At each iteration, n_a ants probabilistically construct solutions to the combinatorial optimization problem under consideration, exploiting a given pheromone model. Then, optionally, a local search procedure is applied to the constructed solutions. Finally, before the next iteration starts, some of the solutions are used for performing a pheromone update. The details of this framework (Blum and Dorigo 2004) are explained in more detail in the following.

InitializePheromoneValues($\mathcal{T}$). At the start of the algorithm the pheromone values are all initialized to a constant value $c > 0$.

ConstructSolution($\mathcal{T}$). The basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions. A constructive heuristic assembles solutions as sequences of elements from the finite set of solution components $\mathfrak{C}$. A solution construction starts with an empty partial solution $\mathfrak{s}_p = \langle\rangle$. Then, at each construction step ψ the current partial solution $\mathfrak{s}_p$ is extended by adding a feasible solution component $\mathfrak{c}^\psi$ from the set $\mathfrak{N}(s_p) = \mathfrak{C} \setminus \{\mathfrak{c}_k^\pi | m \leq \psi, l_m = \pi, k = 1, 2, \ldots, n\}$. Since $\mathfrak{c}_k^\pi$ is the solution component corresponding to the assignment of site π to to facility k, the set $\{\mathfrak{c}_k^\pi | m \leq \psi, l_m = \pi, k = 1, 2, \ldots, n\}$ represents the set of all solution components whose sites have been allocated to some facility at solution construction phase ψ. When we define the set of feasible solution components $\mathfrak{N}(s_p)$ in this way constraints Eqs. 5,6 are automatically satisfied, since the solution components corresopnding to already allocated sites are removed from the feasible set at every phase ψ. The process of constructing solutions can be regarded as a walk (or a path) on the so-called construction graph $\mathcal{G}_C = (\mathfrak{C}, \mathfrak{L})$, which is a fully connected graph whose vertices are the solution components $\mathfrak{C}$ and the set $\mathfrak{L}$ are the connections. The choice of a solution component from $\mathfrak{N}(s_p)$ is, at each construction step, done probabilistically. In most ACO algorithms the probabilities for choosing the next solution component – also called the *transition probabilities* – are defined as follows (Blum

Algorithm 1: Framework of a basic Ant Colony Optimization Algorithm

Algorithm:ACO procedure

Input: An instance of the problem $\mathcal{P}$ of a CO problem model $\mathcal{P} = (\mathcal{S}, f, \Omega)$

InitializePheromoneValues($\mathcal{T}$) ;

$\mathfrak{s}_{bs} \leftarrow NULL$;

while *Termination criteria not true* **do**

  $\mathfrak{S}_{iter} \leftarrow \phi$;

  for *each ant* k **do**

    $\mathfrak{s} \leftarrow$ ConstructSolution($\mathcal{T}$) ;

    $\mathfrak{s} \leftarrow$ LocalSearch($\mathfrak{s}$) ;

    if *(*$f(\mathfrak{s}) < f(\mathfrak{s}_{bs})$*) or (*$\mathfrak{s}_{bs} == NULL$ *)* **then**

      $\mathfrak{s}_{bs} = \mathfrak{s}_{bs}$

    end

    $\mathfrak{S}_{iter} \leftarrow \mathfrak{S}_{iter} \bigcup \{\mathfrak{s}\}$;

  end

  ApplyPheromoneUpdate($\mathcal{T}, \mathfrak{S}_{iter}, \mathfrak{s}_{bs}$) ;

end

Output: The best– so– far solution $\mathfrak{s}_{bs}$

and Dorigo 2004):

$$p(\mathfrak{c}_i^j|\mathfrak{s}_p) = \frac{{\tau_i^j}^\alpha}{\sum_{\mathfrak{c}_k^l \in \mathfrak{N}(s_p)} {\tau_i^l}^\alpha} \ , \quad \forall \mathfrak{c}_i^j \in \mathfrak{N}(s_p) \ , \tag{8}$$

where α is a positive parameter of the algorithm. ApplyPheromoneUpdate($\mathcal{T}, \mathfrak{S}_{iter}, \mathfrak{s}_{bs}$) Most ACO algorithms use the following pheromone value update rule:

$$\tau_i^j \leftarrow (1-\rho)\tau_i^j + \rho \sum_{\left\{\mathfrak{s} \in \mathfrak{S}_{upd} | \mathfrak{c}_i^j \in \mathfrak{s}\right\}} F(\mathfrak{s}), \tag{9}$$

for $i = 1, \cdots, n$, and $j \in \{1, \cdots, |\mathcal{D}_i|\}$. $\rho \in (0, 1]$ is a parameter called evaporation rate. $F : \mathfrak{S} \mapsto R^+$ is a function such that $f(\mathfrak{s}) < f(\mathfrak{s}\prime) \Rightarrow F(\mathfrak{s}) \geq F(\mathfrak{s}\prime), \quad \forall \mathfrak{s} \neq \mathfrak{s}\prime \in \mathfrak{S}$. $F(\mathfrak{s})$ is commonly called the quality function. Instantiations of this update rule are obtained by different specifications of $\mathfrak{S}_{upd}$, which is a subset of $\mathfrak{S}_{iter} \bigcup \{\mathfrak{s}_{bs}\}$, where $\mathfrak{S}_{iter}$ is the set of solutions that were constructed in the current iteration, and $\{\mathfrak{s}_{bs}\}$ is the best-so-far solution. A well-known example of update rule Eq.9 is the AS-update rule (Dorigo et al. 1996) (i.e., the update rule of Ant System (AS)) which is obtained from Eq.9 by setting $\mathfrak{S}_{upd} \leftarrow \mathfrak{S}_{iter}$ The goal of the pheromone value update rule is to increase the pheromone values on solution components that have been found in high quality solutions. In AS the quality function $F(\mathfrak{s})$ is defined as follows,

$$F(\mathfrak{s}) = \frac{Q}{f(\mathfrak{s})}, \quad \forall \mathfrak{s} \quad \in \quad \mathfrak{S}_{iter} \tag{10}$$

where Q is a constant. In Dorigo et al. (1996) elitist strategy for trail update was also suggested as per Eq.11.

$$F(\mathfrak{s}_{bs}) = n_e.\frac{Q}{f(\mathfrak{s}_{bs})} \tag{11}$$

where $\mathfrak{s}_{bs}$ is the the best–so–far solution and n_e is the number of elitist ants, a parameter of the algorithm.

Another method for trail update was introduced in (Bullnheimer et al. 1997) and called as Rank-based Ant system (AS_{rank}) In the AS_{rank}, the solutions created by the ants are ranked according to how well they solve the problem. Let μ_i denote the rank of the solution of ant i. Then $\mathfrak{s}_{iter}$ (the best solution in the current iteration) will have rank 1. $\mathfrak{S}_{upd}$ is defined as (Bullnheimer et al. 1997),

$$\mathfrak{S}_{upd} = \mathfrak{S}_{rank} \bigcup \{\mathfrak{s}_{bs}\}, \tag{12}$$

where $\mathfrak{S}_{rank}$ is defined by Eq.13 as

$$\mathfrak{S}_{rank} = \{\ \mathfrak{s}_i \mid \mu_i < \lambda\} \tag{13}$$

where λ denotes the number of top ranked ants used for trail update.

In AS_{rank} , the quality function $F(\mathfrak{s})$ is defined as follows,

$$F(\mathfrak{s}_i) = Q.\frac{\lambda - \mu_i}{f(\mathfrak{s}_i)}, \quad \forall \mathfrak{S}_i \in \mathfrak{S}_{rank} \tag{14}$$

where $\mathfrak{S}_i$ has rank λ_i.

ILLUSTRATIVE EXAMPLE

A C program was developed on a Personal Computer running Linux operating system as the implementation of the algorithm explained previously. The tests were run on a problem described in Yeh (1995) so as to make the comparison of the present technique easier. In this problem there are two eight story buildings on a campus with 12 available sites as shown in Fig. 6. The 12 facilities considered for placement are

1	Reinforcing steel shop, 1	R1	7	Concrete Batch plant, 1	B1
2	Reinforcing steel shop, 2	R2	8	Concrete Batch plant, 2	B2
3	Carpentry shop, 1	C1	9	Job office	JO
4	Carpentry shop, 2	C2	10	Labour residence	LR
5	Falsework shop, 1	F1	11	Electricity equipment and water-spply shop	E
6	Falsework shop, 2	F2	12	Warehouse	H

The data for construction cost matrix, site neighboring index matrix and interactive cost matrix is same as given in Case 1 by Yeh (1995). This data is given as input to the ACO program and the best solution obtained is shown in Fig 6. Rank based Ant System was used with $\lambda = 5$ ranks and one elitist ant. The number of ants in the colony n_a was 150 and the experiements were conducted for a maximum of 150 iterations. Initial value of trail was set to 1 while the evaporation constant, ρ was set to 0.1. The pheromone intensity parameter, α was set to 1. 5 independent experiments were performed with different random seeds each time.

TABLE 1. Comparison of Best solution obtained by ACO and Yeh (1995)

	Objective	R1	R2	C1	C2	F1	F2	B1	B2	JO	LR	E	WH
ACO	90.0	10	12	9	2	8	11	7	4	5	3	1	6
Yeh (1995)	93.0	12	9	8	1	7	3	4	11	6	2	10	4

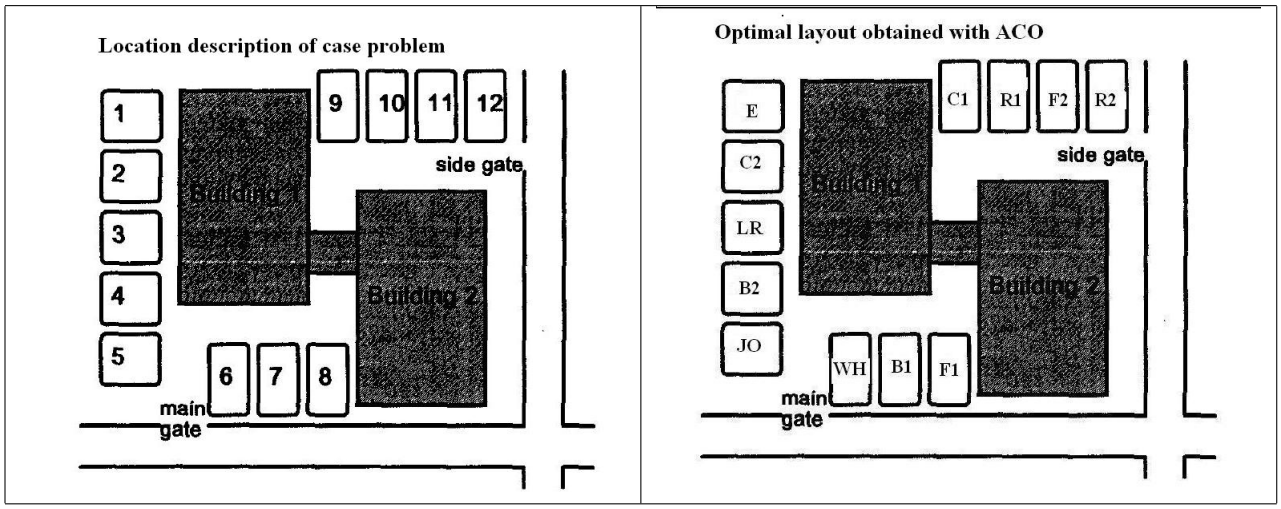

FIG. 2. Layout of site for the illustrative problem.

The best value of objective function was 90.0, while the best solution obtained by Yeh (1995) had objective function value of 93.0. Average objective value was 90.6 which is lower than obtained by Yeh. The average CPU time required for each experiment on a Pentium IV (1.8 GHz) processor was 0.22 s. The locations of the facililities obtained by ACO algorithm and by Yeh (1995) are compared in Table 1. Fig. 3 presents the convergence of the objective function value as the search with ACO proceeds. The best solution corresponding to objective function value 90.0 is obtained at 10th iteration. Fig. 4 shows the solution construction graph used by the ants. The solution construction mechanism allows the construction of only feasible solutions using the definition of the set $\mathfrak{N}(s_p) = \mathfrak{C} \setminus \{\mathfrak{c}_k^{\pi} | m \leq \psi, l_m = \pi, k = 1, 2, \ldots, n\}$. Such a definition of the set of feasible solution components does not allow any vertical or horizontal move on the solution construction graph shown in Fig. 4 and one site is allocated to only one location and vice-versa. Since no exterior penalty handling scheme is used the ACO approach is much more efficient than the penalty function method of handling infeasible solutions by Yeh (1995).

CONCLUSIONS

This paper presented construction site layout problem as problem of allocating n facilities to n sites. The problem is formulated as a combinatorial optimization problem with the objective function being minimization of the total cost of construction and interactive cost of assigning facilities on different locations. The ACO methodology has been presented to find a solution to the problem. The solution construction mechanism to be used by the

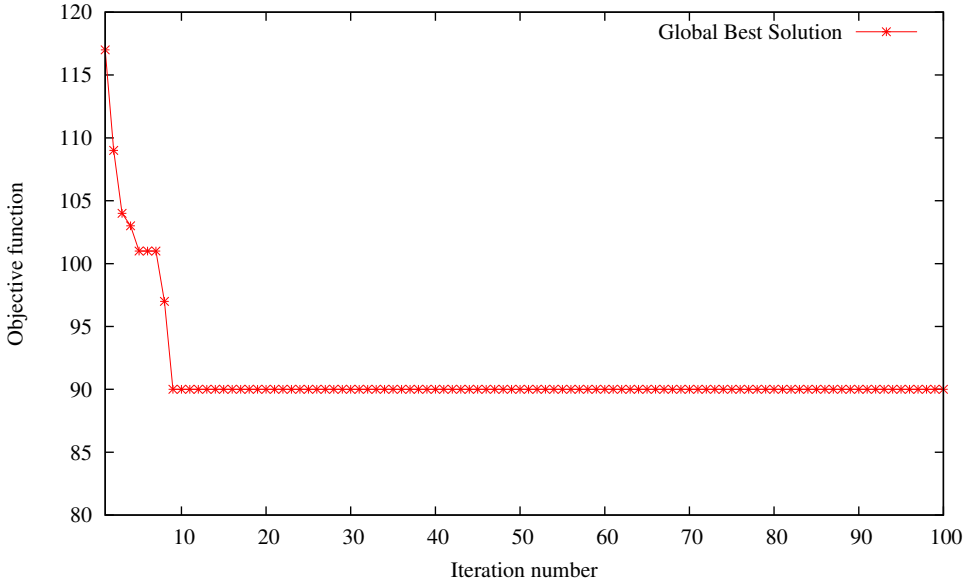

FIG. 3. Convergence of objective function for a typical run.

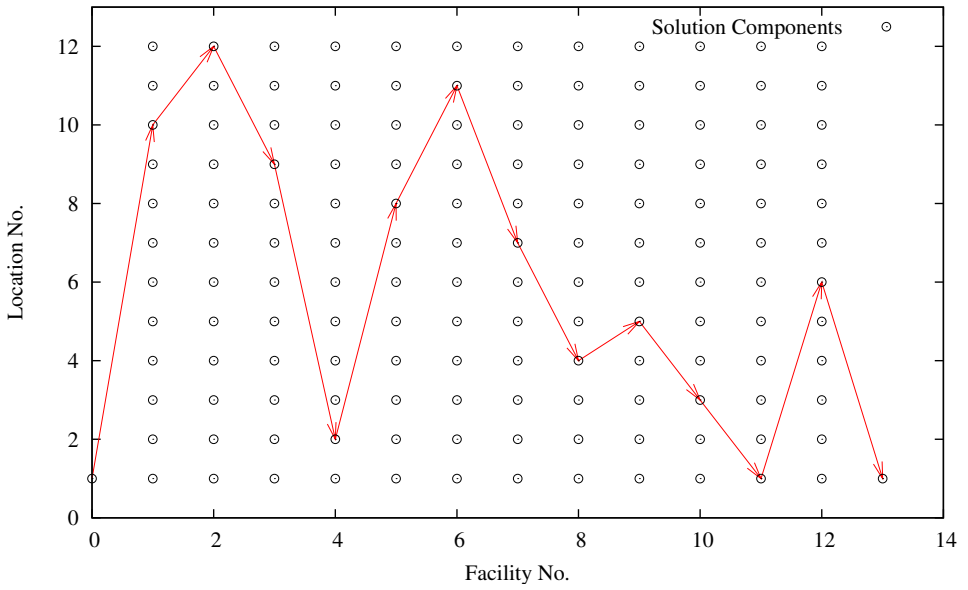

FIG. 4. Construction graph used by ants and the path corresponding to the best solution.

ants (artificial) has been explained. The proposed method has been implemented in the C programming language on a PC. The efficacy of the method has been demonstrated on a 12 facility problem. Since the solution construction mechanism takes care of the site layout constraints, the proposed method is quite efficient in producing feasible low-cost solutions than those proposed in literature.

As an extension to this work, authors are planning to extend the work where the size of facilities and available sites is not the same. Moreover there are multiple objectives to be satisfied in a site layout problem (Tommelein et al. 1992). Recentlly a multiple objective optimization approach to the site layout problem has been presented (Osman and Georgy 2005). Future research will concentrate on solving the multiple-objective site layout problem with ACO.

REFERENCES

Blum, C. and Dorigo, M. (2004). "The hyper-cube framework for ant colony optimization." *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 34(2), 1161–1172.

Bullnheimer, B., Hartl, R., and Strauss, C. (1997). "A new rank based version of the ant system — a computational study.

Cheung, S.-O., Tong, T. K.-L., and Tam, C.-M. (2002). "Site pre-cast yard layout arrangement through genetic algorithms." *Automation in Construction*, 11(1), 35–46.

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). "The ant system: Optimization by a colony of cooperating agents." *IEEE Transactions on Systems, Man, and Cybernetics*, 26, 29–41.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. MIT Press, Cambridge , MA.

Hegazy, T. and Elbeltagi, E. (1999). "Evosite: Evolution-based model for site layout planning." *Journal of Computing in Civil Engineering*, 13(3), 198–206.

Li, H. and Love, P. E. D. (1998). "Site-level facilities layout using genetic algorithms." *Journal of Computing in Civil Engineering*, 12(4), 227–231.

Osman, H. M. and Georgy, M. E. (2005). "Layout planning of construction sites considering multiple objectives: A goal-programming approach." Vol. 183. ASCE, 130–130.

Tommelein, I. D., Levitt, R. E., and Rayes-Roth, B. (1992). "Site layout modeling : how can artificial intelligence help?." *Journal of Construction Engineering and Management*, 118(3), 595–611.

Yeh, I.-C. (1995). "Construction-site layout using annealed neural network." *Journal of Computing in Civil Engineering*, 9(3), 201–208.