An Improved Artificial Immune Network For Solving Construction Site Layout Optimization

Duc Quang Vu
Thai Nguyen University
Email: vdquang1991@gmail.com

Van Truong Nguyen
Thai Nguyen University
Email: nguyenvantruong@dhsptn.edu.vn

Xuan Huan Hoang
VNU University of Engineering and Technology
Email: huanhx@vnu.edu.vn

***Abstract*—Nature-inspired algorithms are often used to find optimal solutions for many combinatorial problems. An immune inspired algorithm, opt-aiNet algorithm, is well known for function optimization. In this paper, we develop a combination of local search with opt-aiNet, called lopt-aiNet, to solve construction site layout (CSL) problem. The effectiveness of the proposed algorithm is investigated through experiments on some datasets taken from the state-of-art and a randomly created dataset. Experimental results show that the lopt-aiNet can produce optimal transportation cost with lower run time compared to the site layouts generated by metaheuristics: Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and aiNet.**

***Keywords*—Construction site layout; Artificial immune system; opt-aiNet.**

I. Introduction

Site layout is an important task which should be considered early in construction planning activity. The objective of this is to arrange the locations and areas reserved for the temporary support facilities such as Site office, Storeroom, Warehouses, etc. depending on the size and location of the project. As usual, this task is performed by construction managers. However, decisions are often made based on intuition, experiments and experience. The impact of good layout practices on money and timesaving becomes more obvious on the larger projects [1]. A good site layout is important to promote safe and efficient operations, minimize travel time, decrease material handling, and avoid obstructing material and equipment movements, particularly for large case projects [2].

There are a lot of models proposed for this problem. Most of them are based on computational intelligence approaches like GA, PSO, ACO, etc. A subfield of computational intelligence, Artificial Immune System (AIS) is consider as one of the most promising approach for optimisation. AIS is computational systems inspired by theoretical immunology, observed immune functions, principles and mechanisms in order to solve problems [3]. They have already attracted much attention from 1990s. Due to their appealing characteristics and corresponding functions as well as success stories of AIS to the field of optimization [4], we would like to apply opt-aiNet, a popular algorithm of AIS, for CSL problem in this research.

The rest of the paper is organized as follows. In the next section, we present the background of CSL problem. Section 3 briefly reviews related works including GA, PSO, ACO and opt-aiNet. Section 4 presents our new algorithm lopt-aiNet in detail. Section 5 describes experiments on 6 case studies. Section 6 concludes the paper and discuss some possible future works.

978-1-5090-4134-3/16/$31.00

II. Site Layout Problem Formulation

In this paper, we assume that the number of both predetermined facilities and places equal n. In case the number of locations is greater than the number of facilities then some "dummy" facilities with zero distance and frequency may be added to ensure that both numbers are equal.

In cases 1 - 3, the facilities have a connection together, such as a salesman usually moves between Site office and Concrete batch workshop, but he rarely moves from Site office to Storeroom. Therefore, the locations for facilities will be carefully chosen in order to minimize the transportation cost, which is also the target of the problem. The total distance is defined as Equation (1) and Equation (2).

$$Min\, F = \sum_{i=1}^{n}\sum_{x=1}^{n}\sum_{j=1}^{n}\delta_{xi} f_{xi} d_{ij} \tag{1}$$

Subject to

$$\sum_{x=1}^{n}\delta_{xi} = 1 \quad \{i = 1, 2, ..., n\} \tag{2}$$

Where n is the number of facilities; δ_{xi} is the permutation matrix variable. Coefficient f_{xi} is the frequencies of trips made by construction personnel between facilities x and i. From the definition, it can be observed that f_{ij} equals to f_{ji}. The frequency is expressed as the number of trips per time period, it is defined in this paper as the number of trips per day. Coefficient d_{ij} is the distances between locations i and j. Therefore, objective function, F, reflects the total traveling distance made by construction personnel. Fig. 1 is an example with n = 11. There are five case studies were selected from the literature. These case studies were selected as they used the same CSL problem, and, thus, they can act as good benchmarks. Moreover, a larger dataset is created randomly as a further benchmark.

In case 1, it is assumed that each of the predetermined location is available for accommodating every facility [6]. The facilities to be located within the site boundaries are shown in Table I. The frequencies of trips (in one day) between facilities are as listed in Table II. As seen from the table, the frequency of the trips from one facility to another and the other way around are the same and that the matrix is symmetric. For

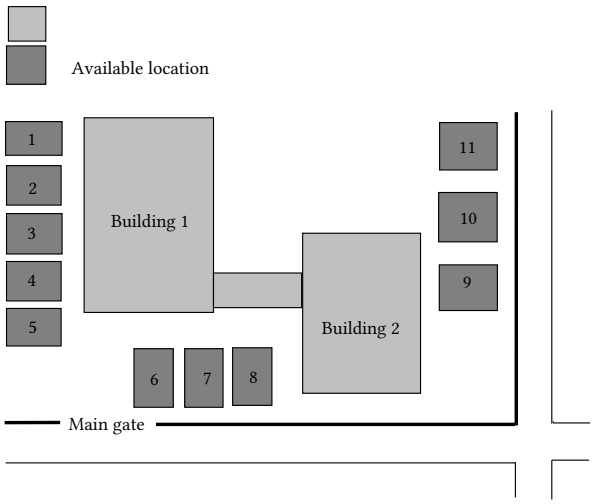

Fig. 1. Location representation of the construction site [5].

TABLE I. THE FACILITIES TO BE LOCATED

Site facilities	Abbreviations
Site office	SO
Falsework shop	FS
Labor residence	LR
Storeroom 1	S1
Storeroom 2	S2
Carpentry workshop	CW
Reinforcement steel workshop	RW
Side gate	SG
Electrical, water and other utilities control room	UR
Concrete batch workshop	BW
Warehouse	W

example, the daily trips from site office to concrete batch workshop and vice versa are 9.

The distances of the available locations are listed in Table III. It should be noted that the site does not offer alternative roads from one location to another. The distances are measured in meters.

In case 2, it is assumed that side gate and main gate are assigned to locations 1 and 10, respectively [7]. This case was defined to represent a real life solution approach in a construction site, which is usually determining the side and main gates before the construction starts as the locations of gates are important for access and, thus, transportation. Therefore, these gates have to be positioned on predetermined locations.

In case 3, it is assumed that site office, labor residence concrete batch shop cannot be allocated to the relatively smaller locations 7 and 8 [8]. Case 3 was used to illustrate the constraint in which facilities that are relatively larger than other facilities cannot be accommodated to every possible location. Unequal area constraint has to be stated to ensure that no larger facilities are positioned to smaller locations.

In cases 4 and 5, the layout objective considered is the cost that calculated from the adjacency and distance of objects, also the consideration of space availability for object location, and the position and view of object in relation to other objects. The problems are adapted from Yeh [2] and Mawdesley et al. [9].

TABLE II. THE FREQUENCY OF TRIPS BETWEEN FACILITIES IN ONE DAY

	SO	FS	LR	S1	S2	CW	RW	SG	UR	BW	W
SO	0	5	2	2	1	1	4	1	2	9	1
FS	5	0	2	5	1	2	7	8	2	3	8
LR	2	2	0	7	4	4	9	4	5	6	5
S1	2	5	7	0	8	7	8	1	8	5	1
S2	1	1	4	8	0	3	4	1	3	3	6
CW	1	2	4	7	3	0	5	8	4	7	5
RW	4	7	9	8	4	5	0	7	6	3	2
SG	1	8	4	1	1	8	7	0	9	4	8
UR	2	2	5	8	3	4	6	9	0	5	3
BW	9	3	6	5	3	7	3	4	5	0	5
W	1	8	5	1	6	5	2	8	3	5	0

TABLE III. DISTANCES BETWEEN AVAILABLE LOCATIONS (M)

	1	2	3	4	5	6	7	8	9	10	11
1	0	15	25	33	40	42	47	55	35	30	20
2	15	0	10	18	25	27	32	42	50	45	35
3	25	10	0	8	15	17	22	32	52	55	45
4	33	18	8	0	7	9	14	24	44	49	53
5	40	25	15	7	0	2	7	17	37	42	52
6	42	27	17	9	2	0	5	15	35	40	50
7	47	32	22	14	7	5	0	10	30	35	40
8	55	42	32	24	17	15	10	0	20	25	35
9	35	50	52	44	37	35	30	20	0	5	15
10	30	45	55	49	42	40	35	25	5	0	10
11	20	35	45	53	52	50	40	35	15	10	0

The CSL is formulated as:

$$Min\, F = \sum_{x=1}^{n}\sum_{i=1}^{n}\delta_{xi}C_{xi} + \sum_{x=1}^{n}\sum_{i=1}^{n}\sum_{y=1}^{n}\sum_{j=1}^{n}\delta_{xi}\delta_{yj}A_{ij}D_{xy} \tag{3}$$

Subject to

$$\delta_{yj} = 0\, if\, \delta_{xi} = 1\, and\, y \neq x \tag{4}$$

$$\delta_{xj} = 1\, if\, \delta_{xi} = 1\, and\, i \neq j \tag{5}$$

where, the objective function, F, is a integration of total distance and cost; δ_{xi} is the permutation matrix variable (=1 if facility x is assigned to location i); C_{xi} is the construction cost of assigning facility x to location i; $A_{ij} = 1$ if location i is neighboring to location j; D_{xy} is the interactive cost of assigning facility x on the location neighboring facility y. Fig. 2 is an example with n = 12.

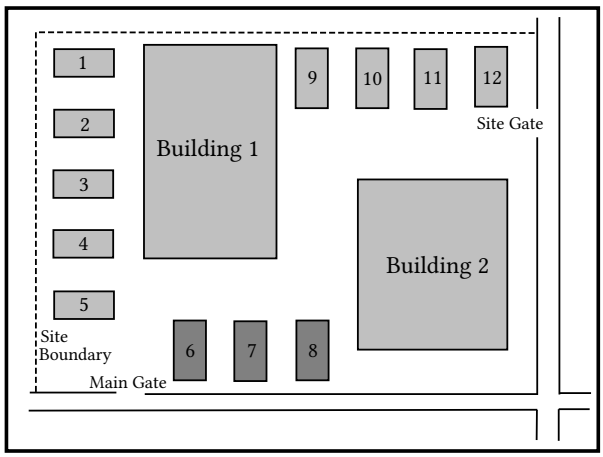

Fig. 2. Example Site Layout [2].

TABLE IV. CONSTRUCTION COST MATRIX (C)

	1	2	3	4	5	6	7	8	9	10	11	12
R1	35	35	30	30	35	15	10	15	6	6	7	10
R2	35	30	9	9	13	30	30	35	15	18	12	7
C1	18	15	15	15	15	8	14	10	8	10	15	15
C2	13	7	12	18	18	15	15	15	15	8	8	12
F1	18	15	15	20	15	8	10	8	8	7	15	15
F2	14	8	10	17	12	15	15	15	15	8	7	9
B1	32	35	15	15	15	10	9	13	7	10	15	15
B2	31	30	9	8	15	18	15	16	15	15	15	15
JO	39	35	13	8	8	15	18	15	8	18	9	18
LR	18	8	8	8	15	10	15	15	13	15	15	15
E	7	10	8	19	15	10	10	8	15	10	6	15
W	9	10	6	7	7	7	15	15	18	15	15	12

TABLE V. CASE 4 SITE NEIGHBORING INDEX MATRIX (A)

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	0	0	0	0	0	0	0	0	0	0
2	1	0	1	0	0	0	0	0	0	0	0	0
3	0	1	0	1	0	0	0	0	0	0	0	0
4	0	0	1	0	1	0	0	0	0	0	0	0
5	0	0	0	1	0	1	0	0	0	0	0	0
6	0	0	0	0	1	0	1	0	0	0	0	0
7	0	0	0	0	0	1	0	1	0	0	0	0
8	0	0	0	0	0	0	1	0	1	0	0	0
9	0	0	0	0	0	0	0	1	0	1	0	0
10	0	0	0	0	0	0	0	0	1	0	1	0
11	0	0	0	0	0	0	0	0	0	1	0	1
12	0	0	0	0	0	0	0	0	0	0	1	0

In case 4, there are two permanent buildings on a campus to be constructed. There are 12 available locations where the following facilities may be placed: Reinforcing steel shop 1 (R1), Reinforcing steel shop 2 (R2), Carpentry shop 1 (C1), Carpentry shop (C2), Falsework shop 1 (F1), Falsework shop 2 (F2), Concrete batch plant 1 (B1), Concrete batch plant 2 (B2), Job office (JO), Labor residence (LR), Electricity equipment and water-supply shop (E), Warehouse (W). The construction cost matrix (C), site neighboring index matrix (A) and interactive cost matrix (D) (the unit of all costs in the test case is 1,000) are shown in Table IV, V and VI, respectively.

Case 5 is similar to case 4, except the site neighboring index matrix and interactive cost matrix that are shows in Table VII and Table VIII, respectively.

Case 6 is similar to case 1, except $n = 50$ and frequencies as well as distances are randomly generated[1].

III. LITERATURE REVIEW

CSL problem is classified as quadratic assignment problem, which is an NP hard Problem [10, 11]. There are various approaches to solve the problem and most of them are approximate algorithms such as PSO, GA, ACO, etc.

The GA methods have been widely applied for solving CSL problems [7, 8]. Mawdesley et al. [9] proposed a solution technique where the cost of movement is modeled using an augmented GA. They formulated the CSL problem as sequence-based GA. Four kinds of crossover operators, i.e., order 1, order 2, partially mapped and cycle, and another two mutation operators i.e., scramble sub-list and reversal were used to explore the global optimum solution in the new

[1]The dataset could be downloaded from https://goo.gl/fkbZzr

TABLE VI. CASE 4 INTERACTIVE COST MATRIX (D)

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	0	0	0	0	100	100	0	0
2	0	0	0	0	0	0	0	0	100	100	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0
9	100	100	0	0	0	0	0	0	0	0	0	0
10	100	100	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0

TABLE VII. CASE 5 SITE NEIGHBORING INDEX MATRIX (A)

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	0.5	0	0	0	0	0	0	0	0.5	0
2	1	0	1	0.5	0	0	0	0	0	0	0	0.5
3	0.5	1	0	1	0.5	0	0	0	0	0	0	0
4	0	0.5	1	0	1	0.5	0	0	0	0	0	0
5	0	0	0.5	1	0	1	0.5	0	0	0	0	0
6	0	0	0	0.5	1	0	1	0.5	0	0	0	0
7	0	0	0	0	0.5	1	0	1	0.5	0	0	0
8	0	0	0	0	0	0.5	1	0	1	0.5	0	0
9	0	0	0	0	0	0	0.5	1	0	1	0.5	0
10	0	0	0	0	0	0	0	0.5	1	0	1	0.5
11	0.5	0	0	0	0	0	0	0	0.5	1	0	1
12	0	0.5	0	0	0	0	0	0	0	0.5	1	0

TABLE VIII. CASE 5 INTERACTIVE COST MATRIX (D)

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	0	0	0	0	100	100	100	0
2	0	0	0	0	0	0	0	0	100	100	100	0
3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	100	100	100	0
8	0	0	0	0	0	0	0	0	100	100	100	0
9	100	100	0	0	0	0	100	100	0	-50	-50	-50
10	100	100	0	0	0	0	100	100	-50	0	0	0
11	100	100	0	0	0	0	100	100	-50	0	0	0
12	0	0	0	0	0	0	0	0	-50	0	100	0

population. Ka Chi Lam et al. [12] proposed conjoining from Max-Min Ant System (MMAS) to GA.

The PSO is a population based stochastic optimization technique inspired by social behavior of bird flocking or fish schooling to a promising position for certain objectives. The position of a particle can be used to represent a candidate solution for the problem at hand. A swarm of particles with randomly initialized positions would fly toward the optimal position along a path that is iteratively updated based on the current best position of each particle, i.e., local best and the best position of the whole swarm, i.e., global best [13]. PSO is well known for the potential in solving the facility layout problem, particularly in the construction field. In [13], Zhang and Wang proposed a PSO-based method to solve the construction site unequal-area facility layout problem. Lien and Cheng [14] proposed a hybrid method of PSO and bee algorithm namely Particle-Bee Algorithm to solve CSL problem.

The ACO is a biological inspired metaheuristics that mimics the behavior of ants searching for food. This algorithm has been founded on the observation of real ant colonies by

Dorigo et al. [15]. In the natural world, ants start to look randomly for food. Each ant would select a path randomly, and the ant on the shortest path will tend to deposit pheromone with higher concentration than the rest of the colony. As such, the ants nearby will smell the highly-concentrated pheromone and join the shortest path, which will increase the pheromone concentration. More ants will keep joining until the majority of the colony converges to the shortest path. The idea of ACO is to mimic this behavior with simulated ants walking around the graph representing the problem to solve. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. For CSL, ACO algorithm is utilized to solve the problem in a hypothetical medium-sized construction project [16]. Ning and Liu [11] used MMAS, which is one variation of ACO algorithm to solve CSL planning. Gharaie et al. [6] utilized ACO to solve a static CSL for a construction project.

Gulben Calis and Orhan Yuksel [17] proposed a hybrid approach of ACO and Local Search (2-opt). In [5], Gulben Calis and Orhan Yuksel proposed a hybrid of ACO, Parametric Analysis (PA) and Local Search (2-opt) which led to better results than those in previous literature. Adrian [18] present method of selecting optimal arguments for 3 algorithms GA, PSO and ACO. ACO is considered the fastest among the three algorithms to find the optimum result.

IV. Application of lopt-aiNet to Construction Site Layout Problem

A. Local Search

Local search is a heuristic method for solving computationally hard optimization problems. Local search can be used on problems that can be formulated as finding a solution maximizing a criterion among a number of candidate solutions. Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until a solution deemed optimal is found or a time bound is elapsed.

A local search is used to improve the quality of the solution that is found by the algorithm. In this paper, we use 2-exchange neighborhood structure for our implementation. The algorithm is showed in the Algorithm 1. Integer value vectors of length *n*, *S* and *S'*, are presentations of candidates where $S[i] = j$ (i,$j = 1, .., n$) that means location of facility *i* is *j*, and *S'* is as a temporary variable.

Algorithm 1 Local Search

```
1: procedure LocalSearch
2:     for i = 1 to n do
3:         S' = S;
4:         for j = 1 to n do
5:             swap S'[i] for S'[j];
6:             if F(S') < F(S) then
7:                 S = S';
8:     return S;
```

B. Opt-aiNet

AiNet algorithm was first proposed by de Castro and Von Zuben in [19] to perform data analysis and clustering tasks. Opt-aiNet, evolves a population, which consists of a network of antibodies (considered as candidate solutions to the function being optimised). These antibodies undergo a process of evaluation against the objective function, clonal expansion, mutation, selection and interaction between themselves. Opt-aiNet creates a memory set of antibodies that represent (over time) the best candidate solutions to the objective function [20].

Opt-aiNet consists of a network of antibodies that similar to population in GA. It has selection and mutation methods like that of GA. All opt-aiNet, PSO and ACO have memory set. But among four algorithms GA, opt-aiNet, PSO and ACO, only opt-aiNet has some following features:

- The population size is dynamically adjustable.
- Has the capability of maintaining many optima solutions.
- Has interaction.
- Demonstrates exploitation and exploration of the search space.

These interesting properties inspire us to take opt-aiNet into account for the CSL problem.

There are some modified versions of opt-aiNet. De Castro and Timmis developed a version of aiNet for multimodal optimization problems, called opt-aiNet (Artificial Immune Network for Optimization) [3]. Copt-aiNet was further proposed by Gomes et al. in [21] as an extension of opt-aiNet for combinatorial optimization tasks. Dopt-aiNet (Artificial Immune Network for Dynamic Optimization) [22], is an improved and extended version of opt-aiNet for time-varying fitness functions. In all works, the authors demonstrated empirically the suitability of the cited algorithms for each kind of optimization problem, with competitive results when compared to the literature. Another algorithm, Omni-aiNet, is mainly based on opt-aiNet, but incorporates some mechanisms introduced by dopt-aiNet [23].

To the best of our knowledge, there has not been any published attempt in applying opt-aiNet or its modified version for the CSL problem.

A modified version of the opt-aiNet in [20] is showed in the Algorithm 2. In this version, cardinality of the population is always stable.

Algorithm 2 Optimization Artificial Immune Network (opt-aiNet)

```
1: procedure OPT-AINET
2:     Randomly initialise population including N cells
3:     repeat
4:         Determine fitness of each network cell against
       objective function;
5:         Generate Nc clones for each network cell;
6:         Apply hypermutation for each clone;
7:         Determine the fitness of mutated clones;
8:         Apply clone selection to the Nc most fit and remove
       the others;
9:         Update the population by adding N - Nc cells to it;
10:    until A stopping criteria has been met
11:    Choose a cell with the highest affinity as a solution;
```

The initial population consists of *N* network cells randomly created (line 2). Each cell is a real value vector, which

TABLE IX. RESULTS COMPARISON FOR CASES 1-3

Case	GA [7]	GA [8]	PSO [13]	ACO [6]	ACO [17]	ACO-PA [5]	opt-aiNet	lopt-aiNet
1				12546		12150	12436	12150
2	15090					12578	12582	12546
3		15160	16060		12628	12606	12616	12606

represents a candidate solution. In lines 4-9, each network cell undergoes a process of clonal expansion and affinity maturation. Clones of cells are mutated according to the affinity of the parent cell. The fitness represents the value of the function for the specific candidate solution.

C. Lopt-aiNet

For improving performance of aiNet, local search is used for fast locating local optima. The proposed lopt-aiNet algorithm is described in Algorithm 3. In comparison with original opt-aiNet, this algorithm is extended by adding local search (line 9) and resizing population by removing some worst cells (line 10). Our aim is to fast locate optima and finish the algorithm.

Algorithm 3 Local search with Artificial Immune Network (lopt-aiNet)

1: **procedure** LOPT-AINET
2: Randomly initialise population including *N* cells.
3: **repeat**
4: Determine fitness of each network cell against objective function;
5: Generate *Nc* clones for each network cell;
6: Apply hypermutation for each clone;
7: Determine the fitness of mutated clones;
8: Apply clone selection to the *Nc* most fit cells and remove the others;
9: Apply local search for one of the best network cells;
10: Reduce population by removing Nr worst network cells;
11: Update the population by adding *N* - *Nc* cells to it;
12: **until** A *stopping criteria* has been met
13: Choose a cell with the highest affinity as a solution;

V. EXPERIMENTS

Lopt-aiNet algorithm is tested with three datasets (cases 1-6) as shown in Section 2. We use computer with CPU Pentium P6200 2.13GHz, RAM 2GB for testing. The objective of the experiments is to compare our algorithm's performance with that of some approaches presented in [5–8, 13, 17, 18].

Arguments used in experiments for cases 1-5 are $N = 200$, $Nr = 1$, *Nc* = 10 and *stopping criteria* is 100 iterations. In implementation for case 6, arguments are configured as $N = 200$, $Nr = 0$, *Nc* = 10 and *stopping criteria* is 1000 iterations. As can be shown in Table IX, lopt-aiNet algorithm yielded better results in comparison with those of GA, PSO, ACO and opt-aiNet algorithms. For case 1 and case 3, the results of lopt-aiNet and ACO-PA algorithms are the same (namely 12150 in case 1 and 12606 in case 3). In case 2, however, lopt-aiNet was able to generate a site layout where total travel distance was reduced by 0.25% compared to the most recent work in [5]. In term of running time, ACO-PA in [5] found the optimal solution in 1.15 seconds on Intel Core 2 Duo processor at 2.66 GHz and 4 GB of RAM, meanwhile our lopt-aiNet can produce the optima only in 0.15 seconds on a less powerful computer.

TABLE X. RESULTS COMPARISON FOR CASES 4 AND 5

Run	GA [18]		PSO [18]		ACO [18]		opt-aiNet		lopt-aiNet	
	result	*time*	*result*	*time*	*result*	*time*	*result*	*time*	*result*	*time*
Case 4										
1st	91	0.53	90	1.93	90	0.37	100	0.19	90	0.22
2nd	90	0.52	91	1.97	90	0.34	101	0.18	90	0.18
3rd	93	0.56	90	1.97	93	0.32	100	0.18	90	0.20
4th	90	0.58	90	1.93	91	0.33	102	0.21	90	0.21
5th	91	0.52	92	1.96	90	0.32	103	0.20	90	0.20
Ave.	91.0	0.54	90.6	1.96	90.8	0.33	101.2	0.19	90.0	0.20
Case 5										
1st	90	0.52	93	1.82	90	0.37	103	0.19	90	0.21
2nd	92	0.55	90	1.87	91	0.35	104	0.20	90	0.20
3rd	90	0.54	90	1.89	93	0.35	105	0.18	90	0.22
4th	96	0.54	91	1.88	91	0.33	100	0.19	90	0.24
5th	90	0.52	90	1.89	90	0.35	104	0.21	90	0.20
Ave.	91.6	0.54	90.8	1.87	91.0	0.35	103.2	0.19	90.0	0.21

TABLE XI. RESULTS COMPARISON FOR CASE 6

Run	ACO	ACO + Local Search	opt-aiNet	lopt-aiNet
1st	144922826	132011036	143964692	129880182
2nd	144376886	131444656	143235094	130465842
3rd	145390728	131243164	143849330	130949714
4th	145261510	131398924	142939814	130821608
5th	144595272	131922126	143995612	130717240
Average	144909444.4	131603981.2	143596908.4	130566917.2

Table X shows the performance with 5 runs of all algorithms GA, PSO, ACO, opt-aiNet and lopt-aiNet. Experiment results show that lopt-aiNet can find the same optimum, 90, in all runs while the others produce greater values in average (in the bottom row). This implies that lopt-aiNet is consistent in finding minimum distance for both cases.

Objective function values drawn from 5 runs of four algorithms ACO, opt-aiNet, ACO combined with local search, and lopt-aiNet (Table XI), show that our algorithm lopt-aiNet can find the shortest total distance for this large project demo. In the implementation of ACO and ACO combined with local search, we used 10 ants with 1000 iterations. Besides, local search based on 2-opt was integrated into ACO similar to that of lopt-aiNet for a fair comparison.

VI. CONCLUSIONS

In this paper, we have proposed a novel approach to combine local search and artificial immune network. The new algorithm, lopt-aiNet, allows for fast localization of the optima. Experiments show that lopt-aiNet outperforms four other algorithms opt-aiNet, GA, ACO and PSO in terms of both objective function value and running time, except opt-aiNet with lower running time in case 5 and case 6. Moreover, the optimal results found by all iterative runs (cases 4 and 5) strongly support the consistency of proposed method for CSL problem.

In near future, we are planning to apply our algorithm on other construction tasks such as equipment routing planning and material storage layout for real projects. Besides, how to dynamically calculate arguments like N, Nr for early ending

of the algorithms is currently being pursued by the authors. This will help our approach more suitable for larger case studies.

REFERENCES

[1] A. Hamiani and C. Popescu, "Consite: A Knowledge-Based Expert System for Site Layout," in *Proceedings of the Computing in Civil Engineering: Microcomputers to Supercomputers*, 1988, pp. 248–256.

[2] I. C. Yeh, "Construction-Site Layout Using Annealed Neural Network," *Computing in Civil Engineering*, vol. 9, pp. 201–208, 1995.

[3] L. N. de Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, 2002, pp. 699–704.

[4] E. Y. C. Wong and H. Y. K. Lau, "Advancement in the twentieth century in artificial immune systems for optimization: Review and future outlook," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 2009, pp. 5195–4202.

[5] G. Calis and O. Yuksel, "An Improved Ant Colony Optimization Algorithm for Construction Site Layout Problems," *Building Construction and Planning Research*, vol. 3, pp. 221–232, 2015.

[6] E. Gharaie, A. Afshar, and M. R. Jalali, "Site Layout Optimization with ACO Algorithm," in *Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, 2006, pp. 90–94.

[7] H. Li and P. E. Love, "Comparing Genetic Algorithms and Non-Linear Optimisation for Labor and Equipment Assignment," *Computing in Civil Engineering*, vol. 12, pp. 227–231, 1998.

[8] ——, "Genetic search for solving construction site level unequal area facility layout problems," *Automation in Construction*, vol. 9, pp. 217–226, 2000.

[9] M. J. Mawdesley, S. H. Al-jibouri, and H. Yang, "Genetic algorithms for construction site layout in project planning," *Construction Engineering And Management*, vol. 128, pp. 418–426, 2002.

[10] D. M. Tate and A. E. Smith, "Unequal-area facility layout by genetic search," *IIE Transactions*, vol. 27, pp. 465–472, 1996.

[11] X. Ning and W. H. Liu, "Max-Min Ant System Approach for Solving Construction Site Layout," *Advanced Materials Research*, vol. 328-330, pp. 128–131, 2011.

[12] K. C. Lam, X. Ning, and M. C.-K. Lam, "Conjoining MMAS to GA to Solve Construction Site Layout Planning Problem," *Construction Engineering and Management*, vol. 35, pp. 1049–1057, 2009.

[13] H. Zhang and J. Y. Wang, "Particle Swarm Optimization for Construction Site Unequal-Area Layout," *Construction Engineering and Management*, vol. 9, pp. 739–748, 2008.

[14] L. C. Lien and M. Y. Cheng, "A hybrid swarm intelligence based particle-bee algorithm for construction site layout optimization," *Expert Systems with Applications*, vol. 39, pp. 9642–9650, 2012.

[15] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, pp. 29–41, 1996.

[16] K. C. Lam, X. Ning, and T. Ng, "The application of the ant colony optimization algorithm to the construction site layout planning problem," *Construction Management and Economics*, vol. 25, pp. 359–374, 2007.

[17] G. Calis and O. Yuksel, "A comparative study for layout planning of temporary construction facilities: optimization by using ant colony algorithms," in *Proceedings of the International Conference on Computing in Civil and Building Engineering*, 2010, pp. 267–274.

[18] A. M. Adrian, A. Utamima, and K. J. Wang, "A comparative study of GA, PSO and ACO for solving Construction Site Layout Optimization," *KSCE Journal of Civil Engineering*, vol. 19, pp. 520–527, 2014.

[19] L. N. de Castro and F. J. V. Zuben, "aiNet: An Artificial Immune Network for Data Analysis," *Data Mining: A Heuristic Approach*, vol. 1, pp. 231–259, 2001.

[20] J. Timmis and C. Edmonds, "A Comment on Opt-AiNet: An Immune Network Algorithm for Optimisation," in *Proceedings of the Genetic and Evolutionary Computation (GECCO)*, 2004, pp. 308–317.

[21] L. de C. T. Gomes, J. S. de Sousa, G. B. Bezerra, L. N. de Castro, and F. J. V. Zuben, "Copt-aiNet and the Gene Ordering Problem," in *Proceedings of the Brazilian Workshop on Bioinformatics*, 2003, pp. 27–33.

[22] F. O. de Franca, F. J. V. Zuben, and L. N. de Castro, "An Artificial Immune Network for Multimodal Function Optimization on Dynamic Environments," in *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, 2005, pp. 289–296.

[23] G. P. Coelho and F. J. Von Zuben, "Omni-aiNet: An Immune-inspired Approach for Omni Optimization," in *Proceedings of the 5th International Conference on Artificial Immune Systems*, 2006, pp. 294–308.