Taylor & Francis
Taylor & Francis Group

JOURNAL OF CIVIL ENGINEERING AND MANAGEMENT
ISSN 1392-3730 print/ISSN 1822-3605 online
2012 Volume 18(4): 580–589
doi:10.3846/13923730.2012.704164

AN ANT COLONY SYSTEM BASED DECISION SUPPORT SYSTEM FOR CONSTRUCTION TIME-COST OPTIMIZATION

Yanshuai Zhang[1], **S. Thomas Ng**[2]

[1]*China Development Bank Hong Kong Branch, Suite 3307-3315 33/F One International Finance Center, No. 1 Harbor View Street, Central, Hong Kong*
[2]*Department of Civil Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong*
E-mails: [1]*yanshuai@graduate.hku.hk;* [2]*tstng@hkucc.hku.hk (corresponding author)*

Received 10 Dec. 2010; accepted 02 May 2011

Abstract. Time and cost are the two most important factors to be considered in every construction project. In order to maximize the profit, both the client and contractor would strive to minimize the project duration and cost concurrently. In the past, most of the research studies related to construction time and cost assumed time to be constant, leaving the analyses based purely on a single objective of cost. Acknowledging this limitation, an evolutionary-based optimization algorithm known as an ant colony system is applied in this study to solve the multi-objective time-cost optimization problems. In this paper, a model is developed using Visual Basic for Application™ which is integrated with Microsoft Project™. Through a test study, the performance of the proposed model is compared against other analytical methods previously used for time-cost modeling. The results show that the model based on the ant colony system techniques can generate better solutions without utilizing excessive computational resources. The model, therefore, provides an efficient means to support planners and managers in making better time-cost decisions efficiently.

Keywords: time-cost optimization, ant colony system, Pareto solution, construction project.

1. Introduction

In the construction field, time saving can be transformed into some kinds of opportunities *viz*. early occupancy, saving in overhead cost or bonus (Li, Love 1997; Ng *et al.* 2000). Despite that, to reduce the time would necessitate the mobilization of extra resources (Kapliński, Janusz 2006) and this could lead to an escalation in direct cost (Zheng *et al.* 2004). The relationship between construction time and cost has aroused the interest of researchers to identify the most effective approach to minimize the time or cost, and examples of these include the harmony search (Geem 2010), particle swarm optimization (Zhang, Li 2010; Zhang, Xing 2010), genetic algorithm (Ke *et al.* 2009) and memetic algorithm (Li, Wang 2009) based time-cost trade-off models. Yet, by minimizing the project time does not necessarily lead to the most satisfactory results (Zavadskas *et al.* 2010). Instead, clients and contractors should strive for optimizing both the time and cost are optimized concurrently if they were to maximize their profit under today's competitive environment. This has led to the development of time-cost optimization (TCO) concepts (Zheng, Ng 2005).

The tradeoff relationship of construction time and cost infers that these two parameters are conflicting in nature (Zhang *et al.* 2007), and hence the desired purpose of TCO is to arrive at an optimal compromise between these two opposing objectives so as to bring the overall duration and total cost of a project to a minimum. Over the last decade, various TCO models for the construction domain have been developed, and these include a genetic algorithms (GAs) model (Feng *et al.* 1997) which aimed at improving the hybrid linear/integer model put forwarded by Liu *et al.* (1995) earlier; a TCO model utilizing an adaptive weight approach (Gen, Cheng 2000); and a multi-objective time-cost optimization model based on the amalgamation of both the GAs concepts and a modified adaptive weight approach (MAWA) (Zheng *et al.* 2004).

While the above GA-based multi-objective TCO models serve to establish an optimal overall time and total cost concurrently, the problem of premature convergence exists when it comes to searching for the globally non-dominated solutions (Zheng *et al.* 2005). Besides, the time taken to processing a set of globally non-dominated solutions for a large scale project could be very long. These would affect the effectiveness and efficiency of the decisions being made. Hence, a novel approach for TCO modeling which can overcome the limitations of the GAs would be beneficial. In this paper, an ant colony system (ACS) approach is proposed for solving the TCO problems (Afshar *et al.* 2009).

ACS is a kind of ant colony optimization (ACO) algorithms which were originally proposed by Colorni *et al.* (1991) as a meta-heuristic scheme to locate near-optimal solutions. As reported by Dorigo and Stützle (2004), an ant system (Dorigo *et al.* 1992) being the most primitive ACO algorithms was applied in solving the travel salesmen problem. Then, several extensions of the

ant system including the ACS, elitist ant system, max-min ant system, and so on were inspired to improve its performance (Zhang *et al.* 2007). A comparison of various ACO algorithms have been conducted by Dorigo and Stützle (2004), which concludes that the performance of ACS is superior to the other ACO algorithms in terms of it convergence speed and processing time. By avoiding premature stagnation and improving the search speed, both the quality of solutions and the time required could be improved by using ACS.

In this paper, an automated ACS-based TCO model is introduced and its performance is compared with other modeling techniques. The paper first highlights the development environments being used for building the model. The components of the ACS-based TCO model along with their functions and operation are then introduced. Finally, the model is validated through a case study and the results of comparison are summarized.

2. Modelling environment

Before the ACS-based TCO model was developed, it is necessary to determine which programming language is the most suitable for building up the optimization model and user interface. The prime consideration was readiness of the programming language to interact with a dedicated scheduling tool and database package in a simple and direct manner. This is especially important if the model is to appeal to the planners and project managers who are not too familiarized with programming. In this research, Microsoft Project™ was chosen as the scheduling tool due to its widespread usage in the construction field. As for the package for data storage, Microsoft Excel™ was used because data can be exported to or imported from Microsoft Project™ for analysis easily.

Various development platforms such as FORTRAN™, Visual Basic™, Java™, Visual C™, and Visual Basic for Application™ (VBA) were compared. While all these programming languages are equally powerful and applicable, VBA offers the best integration and interaction with Microsoft Project™ and the database. Therefore, data in Microsoft Excel™ can be used by VBA for analysis while the results generated by VBA can be plugged to Microsoft Project™ so as to identify the critical path and compute the overall project duration without the needs for complicated coding. Consequently, VBA was selected as the language for developing the optimization model and the user interface.

3. The prototype model

The codes in VBA were compiled as a macro program and integrated with Microsoft Project™, and a new tab known as "T/C Analysis" has been added to the menu bar in Microsoft Project™. User can simply invoke the newly developed functions, i.e. "Data Input", "Optimization", and "Data Output" through the pull down menu in Microsoft Project™. To start the analysis, user is required to enter the project data and those data as required for ACS modeling through the "Data Input" module. Acknowledging that it is possible that the time or cost may be fixed, user may choose amongst different optimization strategies to suit the actual need. Should the "Time-Cost Optimization" option be chosen, a multi-objective time-cost optimization will be conducted. Alternatively, the "Total Cost Minimization" or "Time Minimization" would result in the minimization of either the total cost (i.e. the sum of direct cost and indirect cost) or the shortest overall duration respectively. Finally, the results are made available to the user through "Data Output" for checking and confirmation.

3.1. Data input module

The input variables for the ACO-based TCO model include the following:

1. The logical sequence of the activities in the project, or in other words the network of the project.
2. An estimation of the values for the time and cost of each activity including the best-guess, minimum, and maximum time and cost. For the units for the time and cost, "days" and "$" are used. Other units can also be considered, but attention should be paid to the unit of indirect cost being based on the time and direct cost. For example, if "day" and "$" are the units for the time and direct cost, the indirect cost should be determined by "$/day".
3. An estimation of the indirect cost which should be a fix value per day.

To illustrate how data is inputted into the proposed model, a simple discrete TCO problem as derived from Liu *et al.* (1995) is adopted. The project data consisting of seven activities (Table 1) is entered into Microsoft Project™ by the user. Along with the project data is up to five alternative time-cost options for each activity, and this information is inputted into Microsoft Excel™ for subsequent analysis. Apart from that, user is required to enter other data into model through the user interface (Fig. 1), and this information includes the indirect cost for the project, the number of iterations and the reward factor for the ACS algorithm. In this case example, the indirect cost is set as $1,500/day while the number of iterations and reward factor are given as 40 and 20 respectively.

Table 1. Details for the case example (Liu *et al.* 1995)

Activity description	Activity number	Precedent activity	Option	Duration (days)	Direct cost ($)
Site preparation	1		1	14	23,000
			2	20	18,000
			3	24	12,000
Forms and rebar	2	1	1	15	3,000
			2	18	2,400
			3	20	1,800
			4	23	1,500
			5	25	1,000

End of Table 1

Activity description	Activity number	Precedent activity	Option	Duration (days)	Direct cost ($)
Excavation	3	1	1	15	4,500
			2	22	4,000
			3	33	3,200
Precise concrete girder	4	1	1	12	45,000
			2	16	35,000
			3	20	30,000
Pour foundation and piers	5	2,3	1	22	20,000
			2	24	17,500
			3	28	15,000
			4	30	10,000
Deliver PC girders	6	4	1	14	40,000
			2	18	32,000
			3	24	18,000
Erect girders	7	5,6	1	9	30,000
			2	15	24,000
			3	18	22,000

3.2. Optimization module

The ACS-based consists of four main phases, which could be described briefly as following.

Phase 1 – parameters initializing

During this phase, the parameters for ACS including the number of ants and the pheromone (τ_0), evaporation rate, and other parameters shall be initialized.

Phase 2 – solution construction

The probability for ant k to select option j is established according to the selection rule as shown in Eq. 1:

$$p_{ij}(k,t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{j \in (1,ni)} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}, \quad (1)$$

where: $p_{ij}(k,t)$ is the probability that option (i,j) is chosen by ant k for activity i at iteration t; $\tau_{ij}(t)$ is the pheromone value of the option (i,j) being left by the ants in iteration t, indicating the attractiveness of the option (i,j) for ant k; η_{ij} represents a heuristic function which evaluates the utility of choosing option (i,j); and α and β are the weights of τ_{ij} and η_{ij}, which were determined in Phase 1.

Phase 3 – pheromone updating

After one solution is generated, local updating is conducted on the options being visited as follows:

$$\tau_{ij}(t) = (1-\rho) \cdot \tau_{ij}(t-1) + \rho \cdot \tau_0, \quad (2)$$

where τ_0 is the initial pheromone value for each option and ρ is the evaporation rate.

When an entire iteration is finished, global updating would be performed to the iteration-best options (i.e. the options visited by the best ant with the least fitness value in the current iteration) following the following global updating rules:

$$\tau_{ij}(t) = (1-z) \cdot \tau_{ij}(t-1) + z \cdot \Delta\tau; \quad (3)$$

$$\Delta\tau = R \cdot f_{\text{best-iter}}, \quad (4)$$

where: z is the evaporation rate in the global-updating process; R is a constant representing the pheromone reward factor; and $f_{\text{best-iter}}$ is the fitness value of the best ant in this iteration.

Phase 4 – algorithm stopping

Certain number of iterations would be adopted as the stopping criteria for the proposed model which can be determined by the user.

Readers are referred to Ng and Zhang (2008) for further details of the four phases mentioned above in particular the pheromone updating process.

Furthermore, as demonstrated by Ng and Zhang (2008), MAWA is an effective way to solve the multi-objective problems, such approach would be applied in the current model. Hence, the fitness function for the k^{th} solution (in an ACS model, the fitness for the k^{th} ant in the current iteration) can be represented as follows:

$$f(k) = w_{\text{t}} \frac{z_{\text{t}}^{\max} - z_{\text{t}}(k) + \gamma}{z_{\text{t}}^{\max} - z_{\text{t}}^{\min} + \gamma} + w_{\text{c}} \frac{z_{\text{c}}^{\max} - z_{\text{c}}(k) + \gamma}{z_{\text{c}}^{\max} - z_{\text{c}}^{\min} + \gamma}, \quad (5)$$

where $z_{\text{t}}(k)$ and $z_{\text{c}}(k)$ are the time and cost value of ant k respectively; and γ is a positive random number between 0 and 1.

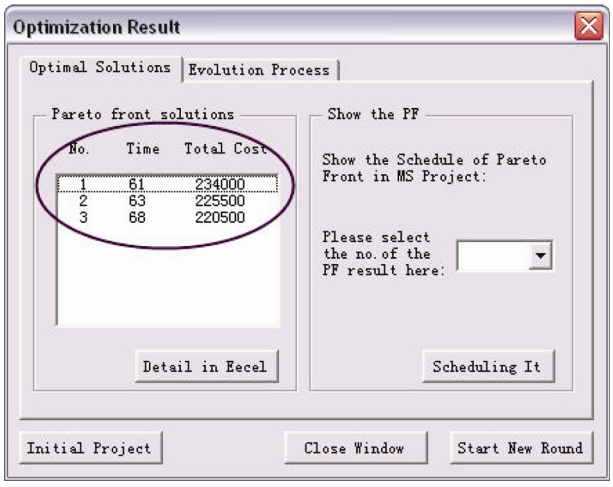
3.3. Output module

After the evaluation, the following information will be provided to the user through the data output module:

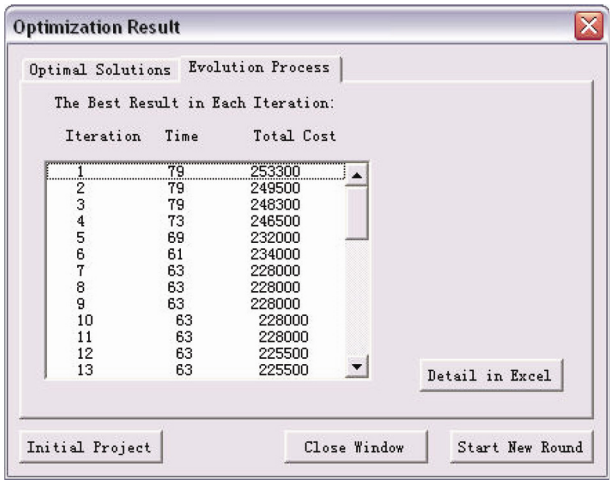
1. The optimum solution(s) of the optimization, which is/are indeed the most optimum value(s) for the total time and cost of a project. For the time-cost optimization, they would prefer to have a series of globally non-dominated solutions. However, when it comes to time and cost optimization, the shortest time and the minimal total construction cost respectively are the most important findings they would like to know.
2. The time and cost options being used for all the activities to generate each of the globally non-dominated solution would be extremely useful, as planners can make use of the solutions to derive programs in the Microsoft Project™ platform.

A summary of all the parameters being entered will be shown to the users for checking when the "Parameter Summary" option in menu is selected. Should the user satisfy with the input data, they can view the results of optimization which include the Pareto solutions for the optimization as well as the evolution process of the time

and cost according to the iteration when they select the "Result Box" option. Taking the results shown in Fig. 1 as an example, there are three globally non-dominated solutions as listed in the left-hand side of the text box (Fig. 1a). Besides, the iter-best solutions (i.e. the best solution in each iteration) are also listed (Fig. 1b) to reflect the convergence of solutions.



a) Solutions at Pareto front



b) The evolution process

Fig. 1. The results of optimization generated by the ACS-based TCO model

3.4. Revised program in project

Should the user wish to schedule the project according to any one of the globally non-dominated solutions as identified, he/she can simply select the number of the solutions in the right box and click the "Schedule It" button. For instance, when Pareto solution number 2 is selected (see Fig. 2), the duration and cost for the activities of this project will be changed accordingly (with total cost and duration being \$225,500 and 63 days respectively) as shown in Table 2. The options as highlighted by the rectangle box are the optimum options for solution number 2.

3.5. Data storage in Excel

More details about the results such as the exact options selected can be seen in Microsoft Excel™ format once the user click the "Details in Excel" button in the results form (Fig. 3). Figs 3a and 3b show the value of the Pareto solutions and the iter-best solutions in Microsoft Excel™ respectively. Fig. 3a shows at which iteration the Pareto front solutions are found and the option for all the seven activities. In the case project, the first optimum solution (i.e. 61 days and \$234,000) is obtained at the 6th iteration and the last one is found in the 17th iteration. After the 17th iteration, ACS can no longer derive a better solution.

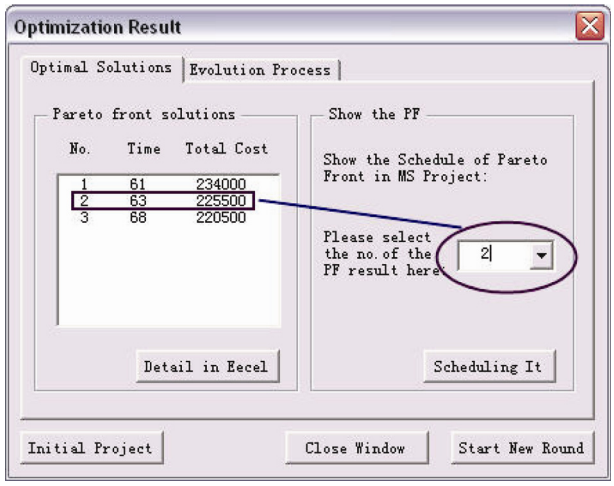


Fig. 2. Selection of Pareto solution

Table 2. Project scheduling according to Pareto front solution number 2

	Task Name	Predecessors	Duration	Cost	D1	C1	D2	C2	D3	C3	D4	C4	D5	C5
1	Site preparation		14 d	\$23,000.00	14	23000	20	18000	24	12000				
2	Forms and rebar	1	15 d	\$3,000.00	15	3000	18	2400	20	1800	23	1500	25	1000
3	Excavation	1	15 d	\$4,500.00	15	4500	22	4000	33	3200				
4	Precise concrete girder	1	16 d	\$35,000.00	12	45000	16	35000	20	30000				
5	Pour foundation and piers	2,3	24 d	\$17,500.00	22	20000	24	17500	28	15000	30	10000		
6	Deliver PC girders	4	24 d	\$18,000.00	14	40000	18	32000	24	18000				
7	Erect girders	5,6	9 d	\$30,000.00	9	30000	15	24000	18	22000				

	A	B	C	D	E	F
1	Paretofront No.	1	2	3		
2	@ iteration No.	6	12	17		
3	Optimum time(Days)	61	63	68		
4	Optimum cost($)	234000	225500	220500		
5	Var. No.1	1	1	1		
6	Var. No.2	1	1	1		
7	Var. No.3	1	1	1		
8	Var. No.4	3	2	3		
9	Var. No.5	1	2	4		
10	Var. No.6	2	3	3		
11	Var. No.7	1	1	1		
12						

a) For Pareto front solutions

A	B	C	D	E	F	G	H	I	J	K
Iteration No.	1	2	3	4	5	6	7	8	9	10
iter-best time(D	79	79	79	73	69	61	63	63	63	63
iter-best cost($	253300	249500	248300	246500	232000	234000	228000	228000	228000	228000
If PF solutions	No	No	No	No	No	Yes	No	No	No	No
Var. No.1	2	2	2	2	2	1	1	1	1	1
Var. No.2	3	1	3	1	1	1	1	1	1	1
Var. No.3	2	2	2	2	1	1	1	1	1	1
Var. No.4	2	3	3	3	2	3	2	2	2	2
Var. No.5	1	1	1	1	1	1	1	1	1	1
Var. No.6	2	2	2	2	3	2	3	3	3	3
Var. No.7	2	2	2	1	1	1	1	1	1	1

b) For *iter-best* solutions

Fig. 3. Data exported to Excel file

Fig. 3b shows part of the Microsoft Excel™ file with the first 10 iterations shown. In the first iteration, the iter-best duration in the critical path is 79 days, and the total cost is $253,300 with an indirect cost of $1,500/day. Then, the time and cost are both reduced to 69 days and $232,000 in the 5th iteration representing a very quick improvement. In the 6th iteration, the shortest duration of 61 days with a total cost of $234,000 – a Pareto front solution is derived.

4. Model validation

In this section, a test project consisting of 18 activities is applied to test the performance of the developed ACS-based model. Since this construction project (Elbeltagi *et al.* 2005) was adopted in several other comparable research studies (e.g. Feng *et al.* 1997; Zheng *et al.* 2005), direct comparisons can be conducted between the current model and the previous ones. The data of the test project including the project activities, their predecessors, estimation values for cost and duration being is highlighted in Table 3. Again, associated with each activity is up to five different possible combinations of time and cost. To evaluate the performance of the ACS-based model, the tests were not restricted to the TCO analysis but also to time and cost optimization analyses respectively.

4.1. Time-Cost optimization

In solving this 18-activity project, the maximum number of iterations and reward factors are set as 200 and 20 respectively. The results confirm that these parameters are proper for this project. Based on the selected criteria, the best solution in each iteration (i.e. the iter-best solution) is first listed, and the best (i.e. Pareto front) solution is then selected from the iter-best solutions of all the iterations (Fig. 4). The evolution trend of the ACS algorithm is shown in Fig. 4 with the speed of evolution being very high at the beginning (i.e. time and cost reduce quickly) but the speed gradually decreases until the algorithm converges at the Pareto front solutions. During the first iteration when the options were randomly selected by the model, the time and cost are 121 days and $317,605 respectively. However, after 200 iterations, the time and cost have significantly reduced to 104 days and $276,320 respectively indicating that this is one of the Pareto front solutions for this test project.

From Fig. 4, it is easy to notice that the convergence of the ACS algorithm is extremely quick, and the evolution is very greedy which is extremely desirable for the best solutions. It follows that it may not be very effective in satisfying the need for diversity of solutions. To obtain more solutions, an ACS-SGPU algorithm was developed whereby a two-step global updating strategy is adopted. The first step is to derive the best solutions, while the subsequent step is to generate the second-best solutions. Since the ants may be attracted to those solutions other than the "best" ones, the diversity of solutions can be guaranteed. Hence, after the global pheromone updating with respect to the best ant, a similar update is conducted with respect to the second best ant according to the following equations:

$$\tau_{ij}(t) = (1-z)\cdot\tau_{ij}(t-1) + z\cdot\Delta\tau\,; \qquad (6)$$

Table 3. Details of the test project (Elbeltagi *et al.* 2005)

Task	Pred	Duration (day)	Cost ($)	Alternatives									
				D1	C1	D2	C2	D3	C3	D4	C4	D5	C5
1		14	2,400	14	2,400	15	2,150	16	1,900	21	1,500	24	1,200
2		20	1,800	15	3,000	18	2,400	20	1,800	23	1,500	25	1,000
3		33	3.200	15	4,500	22	4,000	33	3,200				
4		20	30,000	12	45,000	16	35,000	20	30,000				
5	1	30	10,000	22	20,000	24	17,500	28	15,000	30	10,000		
6	1	18	32,000	14	40,000	18	32,000	24	18,000				
7	5	18	22,000	9	30,000	15	24,000	18	22,000				
8	6	16	200	14	220	15	215	16	200	21	208	24	120
9	6	15	300	15	300	18	240	20	180	23	150	25	100
10	2,6	15	450	15	450	22	400	33	320				
11	7,8	20	300	12	450	16	450	20	300				
12	5,9,10	22	2,000	22	2,000	24	1,750	28	1,500	30	1,000		
13	3	24	1,800	14	4,000	18	3,200	24	1,800				
14	4,10	15	2,400	9	3,000	15	2,400	18	2,200				
15	12	12	4,500	12	4,500	16	3,500						
16	13,14	22	2,000	20	3,000	22	2,000	24	1,750	28	1,500	30	1,000
17	11,14,15	14	4,000	14	4,000	18	3,200	24	1,800				
18	16,17	9	3,000	9	3,000	15	2,400	18	2,200				

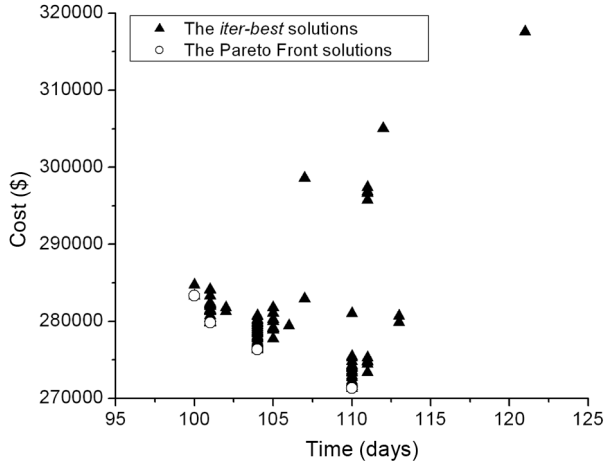


Fig. 4. Solutions after 200 iterations using the ACS

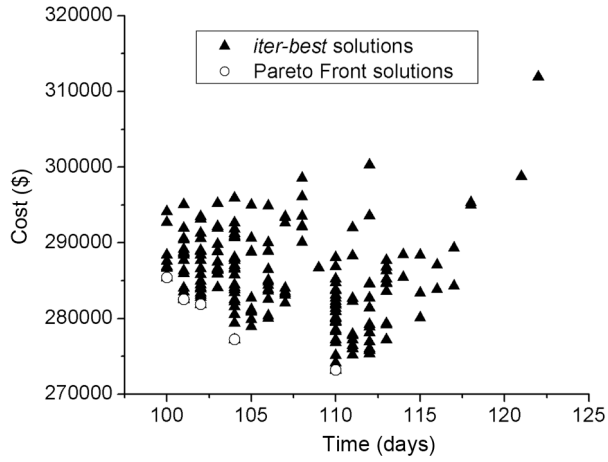


Fig. 5. Solutions after 200 iterations using the ACS-SGPU algorithm

$$\Delta\tau = f_{\text{best-iter}} \cdot R / 2 , \quad (7)$$

where $f_{\text{best-iter}}$ is the fitness value of the best ant in this iteration; and $\tau_{ij}(t)$ represents the pheromone value of the options (i,j) belonging to the second-best tour but not belonging to the best tour, so that the same option being updated two times could be avoided.

With the parameters remain unchanged, the behavior of the ACS-SGPU algorithm can be identified from Fig. 5. From this figure, one can witness that the solution points are more dispersed even with same number of iterations involved. More specifically, the Pareto front solutions obtained by ACS and ACS-SGPU after 200 iterations are listed in Table 4. The results confirm that even though the solutions obtained by ACS-SGPU after 200 iterations are not significantly better than that generated by ACS, the diversity can be guaranteed. In some situation, this would offer managers with more options to choose from when planning the project.

Table 4. Pareto front solutions of the ACS and ACS-SGPU algorithms

Description	ACS algorithm		ACS-SGPU algorithm	
Best results got from the models (with indirect cost=$1500)	*Time (day)*	*Cost ($)*	*Time (day)*	*Cost ($)*
	100	283,320	100	285,400
	101	279,820	101	282,508
	104	276,320	102	281,850
	110	271,320	104	277,200
			110	273,165

Figs 6 and 7 show the convergence of time and cost for ACS and ACS-SGPU algorithms respectively. From these figures, one could notice that the time and cost are reduced to a range which actually including the time and cost values of the Pareto front solution as found in Fig. 6. In the situation of ACS, the time ranges from 100 to 110 days just after around 20 iterations and the cost ranges from $27,500 to $28,500 after 55 iterations. However, as in the case of the ACS-SGPU, the time and cost fluctuate in a slightly broader range mainly because of its diversity.

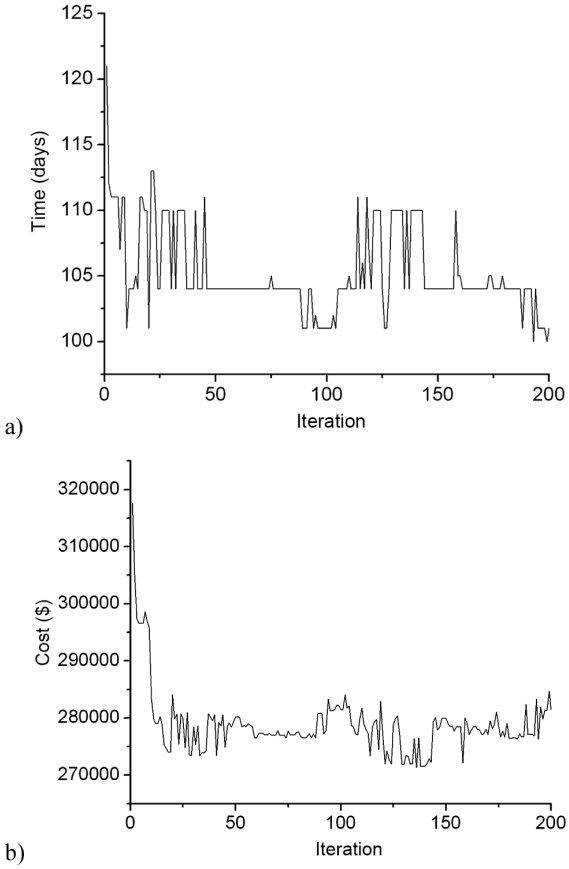


Fig. 6. Convergences of time and cost using the ACS algorithm

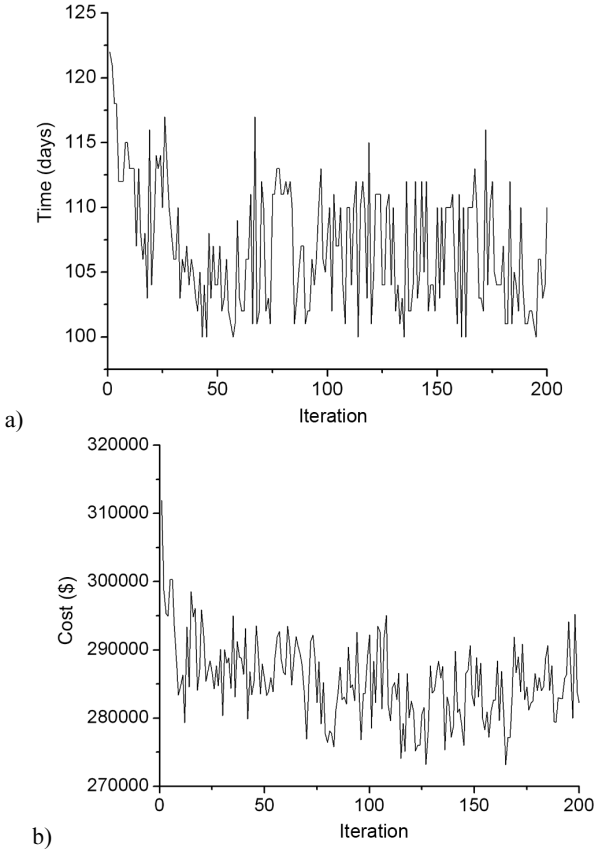


Fig. 7. Convergences of time and cost using the ACS-SGPU algorithm

Figs 6 and 7 can be combined into a 3-D diagram to illustrate the evolutionary trend of the ACS algorithm (Fig. 8) and ACS-SGPU algorithm (Fig. 9). The dots shown in these two figures represent the solutions and they converge to an area quickly with slight fluctuations. The projections of the data points (solutions) can be seen on the relevant 3-D plane. Therefore, these two diagrams can show the relationship between the time and cost; as well as the time-iteration and cost-iteration on the *y-z*, *x-y* and *x-z* projection planes respectively.

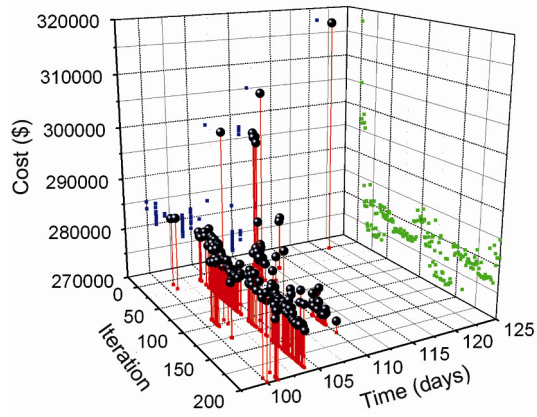


Fig. 8. Evolution of the ACS algorithm

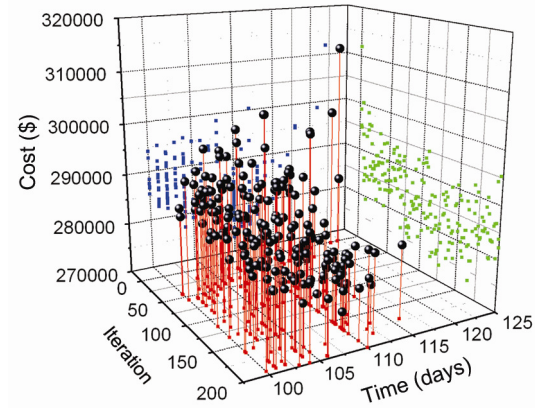


Fig. 9. Evolution of the ACS-SGPU algorithm

4.2. Time optimization

When performing TCO analysis using the ACS techniques, the weight values control the balance of the two objectives – i.e. time and cost. These values can be modified so that the model only optimizes the total cost or total duration. When the manager is concerning the cost only, the weight value could be set as ($w_t = 0$, $w_c = 1$). On the other hand, when the duration is the prime objective the weight value could be changed to ($w_t = 1$, $w_c = 0$).

The performance of ACS in time optimization is shown in Fig. 10. The results show that the "best-so-far" total time of the project as derived by ACS reduces sharply with the iteration, and from the 10th iteration to the end of algorithm, there is no change in terms of the time. The original total time (in the 1st iteration) of the project is 124 days, while the optimum duration is 100 days (after 10th iteration) with 24 days which is equivalent to around

20% reduction. This indicates that this model can minimize the project time very quickly.

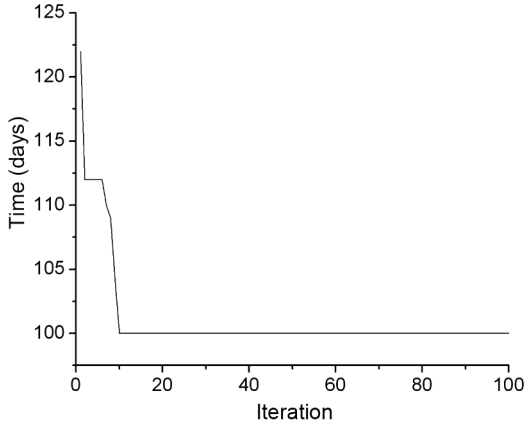


Fig. 10. Convergence of time optimization

4.3. Cost optimization

When the cost is the prime concern (i.e. w_t=0, w_c=1), the model results could vary. To compare the results of Elbeltagi *et al.* (2005), the indirect cost is set as \$500 per day while keeping other parameters unaltered, and the convergence of the cost is shown in Fig. 11. This figure only shows the relationship between the "best-so-far" solutions (the lowest cost found from the very beginning of the algorithm) with the iteration instead of the iter-best solutions. From Fig. 11, an optimal cost of \$161,270 at the duration of 110 days can be reached after 28 iterations, and the result of any subsequent iteration remains unchanged beyond that point.

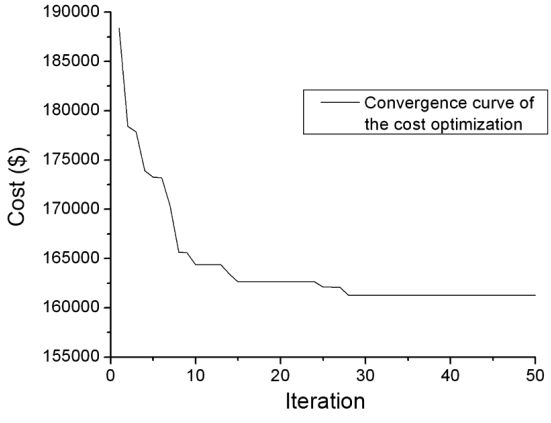


Fig. 11. Convergence of cost optimization

The efficiency of the ACS-based model can also be confirmed by the iter-best solutions for cost optimization as depicted in Fig. 12. From this figure, it is evident that the cost points are almost reducing along with the iteration with the exception of two or three points which have a slight upward deviation. This means the algorithm can almost reach the best solution in every cycle of iteration after the 28th iteration. The ACS is an evolutionary-based algorithm and has the nature to explore new solutions, if the ants always travel in the same way, there will be no evolution of the solution. According to the ACS algorithm, the ants can remember the best solution established in previous iteration. That is to say, the options belonging to this solution will have a greater probability of being selected, but this does not mean that these options can definitely be chosen. In such cases, other options could be selected by the ants and the solution of that iteration may or may not be as good as the previous one, and this can explain the deviation of two or three points in Fig. 12.

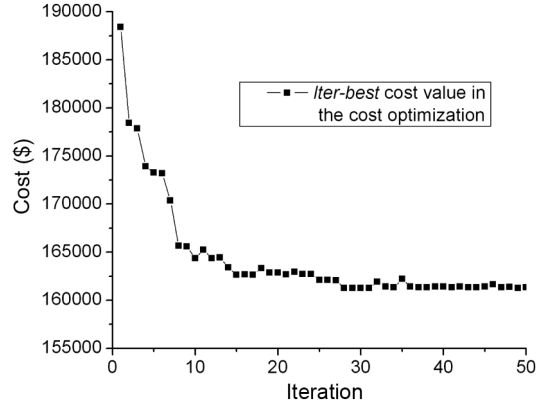


Fig. 12. Cost values based on the cost optimization algorithm

4.4. Comparisons

A comparison amongst the ACS algorithm, ACS-SGPU algorithm and GA-based TCO model (Zheng *et al.*, 2005) using the same project is shown in Table 5. From Table 5, it is apparent that both the populations and number of iterations of the ACS-based model are less than those of the GA-based model. As for the time and cost results for the case project, the ACS-based model can obtain a more optimal cost value under the same duration. For example, when the duration is fixed at 100 days, the GA-based model can only come up with a solution which costs \$287,720 while the solution derived by the ACS algorithm would cost \$283,320 – a saving of \$4,400 which is equivalent to 1.53% of the total cost. In the situation of ACS-SGPU, the total cost is \$285,400, which is in between the GA and ACS models. However, the iterations of ACS-SGPU and ACS are less than that of GA. Even though the quality of solutions generated by ACS-SGPU is not as good as ACS, it is superior to the GA model and can generate more Pareto front solutions (i.e. five solutions from ACS-SGPU as opposed to four in the case of ACS and GA).

To provide a much clearer comparison, the Pareto front solutions of the three models are illustrated in Fig. 13. The comparisons among these three algorithms indicate that ACS has the best ability to search for globally non-dominated solutions; ACS-SGPU has an excellent power in searching for more solutions. The behavior of GA-based model, however, is not as good as other algorithms under investigation.

The discussions above focus on the TCO analysis. Besides, comparison between the ACS-based model and Elbeltagi's ACO-based time-cost tradeoff (TCT) model was also conducted. Since the TCT analysis involves an

Table 5. Comparison between different algorithms

Description	GA model		ACS algorithm		ACS-SGPU algorithm	
Populations in an iteration	50		10		10	
Number of iterations to get the solutions	500		200		200	
Best results got from the models (with indirect cost=$1500)	*Time (day)*	*Cost ($)*	*Time (day)*	*Cost ($)*	*Time (day)*	*Cost ($)*
	100	287,720	100	283,320	100	285,400
	101	284,020	101	279,820	101	282,508
	104	280,020	104	276,320	102	281,850
	110	273,720	110	271,320	104	277,200
					110	273,165

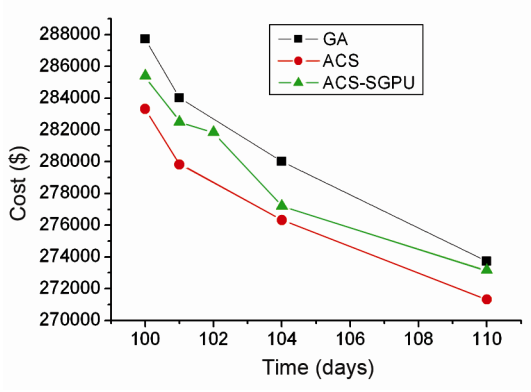


Fig. 13. Comparison of Pareto front between different algorithms

optimization of the total cost, the result would only be the best solution. Hence, the ACS-SGPU algorithm which strives to guarantee the diversity of the solutions is irrelevant to this analysis. Using the ACS algorithm, the best solution of 110 days and $161,270 is reached with lesser number of iterations (Table 6). These results illustrate that the ACS-based model can reduce the computation time especially when the network gets more sophisticated as compared with the previous ACO model, and this is an important advantage of the ACS algorithm.

Table 6. Comparison between the ACO-based and the ACS-based model

Description	ACO based model	ACS-based model
Number of ants	30	10
Number of iterations to get the best solution	100	28
Minimum time	110 days	110 days
Minimum cost	$161,270	$161,270

5. Conclusions

Although many research studies have been conducted using the ant colony optimization algorithms, there has been little attention to the application of such algorithms the ant colony system (ACS) for time-cost optimization (TCO) problems. In this study, a TCO model based on the ACS techniques is developed to optimize the time and cost simultaneously for construction projects. To deal with this multi-objective problem, a modified adaptive weight approach (MAWA) is applied to combine the time and cost into a single objective function through the relevant weight values on the time and cost objectives. In the ACS algorithm for TCO, the weight values can be calculated after the maximal and minimal time and cost of the current iteration. On the other hand, the model can be used to optimize the time or total cost when the relevant weight value is adjusted.

The ACO-based TCO model has been developed using VBA on the platform of Microsoft Project™, and three modules have been built in the proposed model which includes the data input, optimization, and data output. The data input interface allows the user to select the exact algorithm (ACS or ACS-SGPU) and the optimization type (TCO, time optimization or cost optimization). In the optimization module, the ACS algorithm has been coded in accordance with the four development phases. Finally, the results are generated through the data output module. The solutions can be plugged to Microsoft Project™ so as to facilitate interaction with the user. The model, therefore, provides a user-friendly and efficient platform to support time-cost optimization decisions.

The model has been compared with other systems developed using by other approaches, and the results of the validation show that the ACS technique can generate a group of near-global solutions within a shorter processing time indicating that ACS is superior for multi-objective TCO modeling. The results of the validation confirm that MAWA can efficiently work with ACS. Therefore, apart from broadening our understanding on the application domain of ACO algorithms, the model also provides a new direction for modeling construction time-cost analysis. Using the ACS-SGPU, planners or project managers are offered a series of possible solutions for TCO analysis so that the chance of identifying a suitable solution is increased.

References

Afshar, A.; Kasaeian Ziaraty A.; Kaveh, A.; Sharifi, F. 2009. Nondominated archiving multicolony ant algorithm in time-cost trade-off optimization, *Journal of Construction Engineering and Management* ASCE 135(7): 668–674. http://dx.doi.org/10.1061/(ASCE)0733-9364(2009)135:7(668)

Colorni, A.; Dorigo, M.; Maniezzo, V. 1991. Distributed optimization by ant colonies, in *Proc. of the First European Conference on Artificial Life*, 11–13 December, 1991, Paris, France, 134–142.

Dorigo, M.; Di Caro, G.; Gambardella, L. M. 1992. Ant algorithms for discrete optimization, *Artificial Life* 5(2): 137–172. http://dx.doi.org/10.1162/106454699568728

Dorigo, M.; Stützle, T. 2004. *Ant Colony Optimization*. Cambridge: MIT Press. 320 p. http://dx.doi.org/10.1007/b99492

Elbeltagi, E.; Hegazy, T.; Grierson, D. 2005. Comparison among five evolutionary-based optimization algorithms, *Advanced Engineering Informatics* 19(1): 43–53. http://dx.doi.org/10.1016/j.aei.2005.01.004

Feng, C.-W.; Liu, L.; Burn, S. A. 1997. Using genetic algorithms to solve construction time-cost trade-off problems, *Journal of Computing in Civil Engineering* ASCE 11(3): 184–189. http://dx.doi.org/10.1061/(ASCE)0887-3801(1997)11:3(184)

Geem, Z. W. 2010. Multiobjective optimization of time-cost trade-off using harmony search, *Journal of Construction Engineering and Management* ASCE 136(6): 711–716. http://dx.doi.org/10.1061/(ASCE)CO.1943-7862.0000167

Gen, M.; Cheng, R. 2000. *Genetic Algorithms and Engineering Optimization*. New York: Wiley-Interscience Publication. 512 p.

Kapliński, O.; Janusz, L. 2006. Three phases of multifactor modelling of construction processes, *Journal of Civil Engineering and Management* 12(2): 127–134.

Ke, H.; Ma, W.; Ni, Y. 2009. Optimization models and a GA-based algorithm for stochastic time-cost trade-off problem, *Applied Mathematics and Computation* 215(1): 308–313. http://dx.doi.org/10.1016/j.amc.2009.05.004

Li, H.; Love, P. 1997. Using improved genetic algorithms to facilitate time-cost optimization, *Journal of Construction Engineering and Management* ASCE 123(3): 233–237. http://dx.doi.org/10.1061/(ASCE)0733-9364(1997)123:3(233)

Li, H.; Wang, Z. 2009. Memetic algorithm for solving construction time-cost optimization, in *Proc. of the International Conference on Management and Service Science (MASS 2009)*, 20–22 September, 2009, Wuhan, China, IEEE Press, 1–4. http://dx.doi.org/10.1109/ICMSS.2009.5305099

Liu, L.; Burns, S. A.; Feng, C.-W. 1995. Construction time-cost trade-off analysis using LP/IP, *Journal of Construction Engineering and Management* ASCE 121(4): 446–454. http://dx.doi.org/10.1061/(ASCE)0733-9364(1995)121:4(446)

Ng, S. T.; Deng, M. Z. M.; Skitmore, R. M.; Lam, K. C. 2000. A conceptual case-based decision model for mitigating construction delays, *International Journal of Construction Information Technology* 8(2): 1–20.

Ng, S. T.; Zhang, Y. S. 2008. Optimizing construction time and cost using ant colony optimization approach, *Journal of Construction Engineering and Management* ASCE 134(9): 721–728. http://dx.doi.org/10.1061/(ASCE)0733-9364(2008)134:9(721)

Zavadskas, E. K.; Turskis, Z.; Tamošaitienė, J. 2010. Risk assessment of construction projects, *Journal of Civil Engineering and Management* 16(1): 33–46. http://dx.doi.org/10.3846/jcem.2010.03

Zhang, H.; Li, H. 2010. Multi-objective particle swarm optimization for construction time-cost tradeoff problems, *Construction Management and Economics* 28(1): 75–88. http://dx.doi.org/10.1080/01446190903406170

Zhang, H.; Xing, F. 2010. Fuzzy-multi-objective particle swarm optimization for time-cost-quality tradeoff in construction, *Automation in Construction* 19(8): 1067–1075. http://dx.doi.org/10.1016/j.autcon.2010.07.014

Zhang, Y. S.; Ng, S. T.; Kumaraswamy, M. M. 2007. Applying ant colony system to solve construction time-cost trade-off problem, in *Proc. of the 25th Anniversary Conference "Construction Management and Economics"*, 16–18 July, 2007, University of Reading, Reading, United Kingdom. 7 p.

Zheng, D. X. M.; Ng, S. T. 2005. Stochastic time-cost optimization model incorporating fuzzy sets theory and nonreplaceable front, *Journal of Construction Engineering and Management* ASCE 131(2): 176–186. http://dx.doi.org/10.1061/(ASCE)0733-9364(2005)131:2(176)

Zheng, D. X. M.; Ng, S. T.; Kumaraswamy, M. M. 2004. Applying a genetic algorithm-based multiobjective approach for time-cost optimization, *Journal of Construction Engineering and Management* ASCE 130(2): 168–176. http://dx.doi.org/10.1061/(ASCE)0733-9364(2004)130:2(168)

Zheng, D. X. M.; Ng, S. T.; Kumaraswamy, M. M. 2005. Applying Pareto ranking and niche formation to genetic algorithm-based multiobjective time-cost optimization, *Journal of Construction Engineering and Management* ASCE 131(1): 81–91. http://dx.doi.org/10.1061/(ASCE)0733-9364(2005)131:1(81)

Yanshuai ZHANG. Manager of Corporate Banking Division in China Development Bank. He obtained his degree in Masters of Philosophy from The University of Hong Kong. His research interests include financial arrangement of mega infrastructure and construction projects, optimization of construction time and cost, and the application of evolutionary approaches in construction management domains.

S. Thomas NG. Associate Professor in the Department of Civil Engineering, The University of Hong Kong. Thomas is a Chartered Surveyor and Chartered Builder, and he worked in the construction industry for more than ten years. His research interests include low carbon construction, construction industry development, contractor and consultant selection, delays mitigation, time/cost performance, and construction information technology.