Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

PROPOSED METHODOLOGY FOR COMPARING SCHEDULE GENERATION SCHEMES IN CONSTRUCTION RESOURCE SCHEDULING

Jin-Lee Kim

Dept. of Civil Engineering & Construction Engineering Management
California State University, Long Beach
1250 Bellflower Boulevard
Long Beach, CA 90840, USA

ABSTRACT

This paper proposes a methodology to compare the serial scheme with the parallel scheme in a decoding procedure of permutation-based genetic algorithm for construction resource scheduling. Elitist genetic algorithm based on the serial scheme, which was previously developed by the author, is used as a platform to integrate the parallel scheme. Since two schedule generation schemes have different mechanisms, it is meaningful to demonstrate the effects of the scheme on the performance of an algorithm. The ultimate goal of this study is to address the issue with regard to which of the schedule generation schemes will perform better for an arbitrary instance of resource scheduling problems. Thirty problems are solved here to compare their project durations according to the scheme. Scheduling results indicate that Elitist genetic algorithm using the serial scheme provides better solutions than the one using the parallel scheme.

1 INTRODUCTION

An experimental evaluation study shows that commercial software packages for project management provide schedules with an average deviation of 4.3-9.8% from the optimal solution for a project network with up to 30 activities (Hartmann 1998). Most of the construction project scheduling software packages do not have the capability to set the limitation of resources for each activity over the duration of the project. Only a few give schedulers the ability to set resource limits by defining their resources in the resource dictionary. Thus, there is a need for efficient solution approaches to allow for the complexities of real-world problems as well as to search for a near-optimal and/or optimal solution to the resource-constrained project scheduling problem (RCPSP).

The computation of the fitness value for an individual schedule uses a schedule generation scheme. Decoding the schedule representation into a schedule is required in the activity-list representation when utilizing genetic algorithms (GA). The schedule generation scheme builds a feasible schedule by extending a partial schedule stepwise. A partial schedule is a schedule where only a subset of all activities considered has been scheduled. An issue has arisen with regard to which schedule generation scheme between the serial scheme and the parallel scheme will perform better for an algorithm for an arbitrary instance of the RCPSP. No general answer can be given to this issue because of the different mechanisms between them. Finding a better appropriate decoding procedure is necessary to obtain optimal solutions to the RCPSP. In an effort to address this issue, this paper compares two decoding mechanisms using Elitist GA developed by the author for solving multiple RCPSP in construction engineering and management (Kim and Ellis 2008). Elitist GA is designated here to incorporate the parallel scheme as a decoding procedure within the algorithm so that a better scheme can be predicted.

2 RESEARCH BACKGROUND

2.1 Previous Studies on Permutation-based GA

Most meta-heuristics that employed the activity list representation in the RCPSP literature make use of the serial scheme as a decoding procedure (Pinson et al. 1994; Boctor 1996; Hartmann 1998; Baar et al. 1999; Bouleimen and Lecocq 2003; Kim and Ellis 2008). A few meta-heuristics used the parallel scheme as a decoding procedure (Lee and Kim 1996; Kohlmorgen et

al. 1999). The selection of the serial scheme in most research has been recognized due to the fact that all optimal solutions from the search space may not be included when applying the parallel scheme, while the search space of the serial scheme always contains an optimal schedule. Since two schedule generation schemes have different mechanisms, it is meaningful to demonstrate the effects of the scheme on the performance of the algorithm. This paper attempts to find a better appropriate decoding procedure, which is important in searching for optimal solutions to the RCPSP, by comparing the mechanisms of two schedule generation schemes.

2.2 Schedule Generation Scheme

A schedule generation scheme is essential in most heuristic solution procedures for the RCPSP (Hartmann 1999). The computation of a fitness value, which is also known as the makespan or project duration, is a vital mechanism in the meta-heuristic approaches to the RCPSP. The schedule generation scheme can be classified into two different mechanisms, depending on either as the number of activity increases or as time increases. The former is called the serial schedule generation scheme and the latter is called the parallel schedule generation scheme in this paper. Schedules can be classified into one of three types of schedules, as defined by Sprecher et al. (1995): Semi-active schedule (SAS), active schedule (AS), and non-delay schedule (NDS). The set of semi-active schedules are the feasible schedules obtained by sequencing activities as early as possible. No activity can be started earlier without changing the order of the activities. The set of active schedules are the feasible schedules in which no activity could be started earlier without delaying some other activities or violating a precedence relationship. Note that an optimal schedule is always active so the search space can be safely limited to the set of all active schedules. The set of non-delay schedules are the feasible schedules in which no resource is kept idle when it could start scheduling an activity.

A diagram shown in Figure 1 is created by combining the Sprecher et al. (1995)'s theorem that defines semi-active, active, and non-delay schedules for the RCPSP with optimal schedules (Hartmann 1998). The serial scheme constructs active schedules while the parallel scheme constructs non-delay schedules. The set of the non-delay schedules (NDS) is a subset of the set of the active schedules (AS). Since the set of the active schedules (AS) always includes an optimal schedule (OS), this does not keep for the set of the non-delay schedules (NDS). That is, the parallel scheme may exclude an optimal schedule (OS). On the other hand, the parallel scheme may produce schedules of good average quality because it is likely to use the resources as early as possible, leading to compact schedules. Therefore, an issue has arisen in this research with regard to which of the scheme will perform better for an arbitrary instance of the RCPSP. No general answer can be given to this issue because of the difference between the serial scheme and parallel scheme.

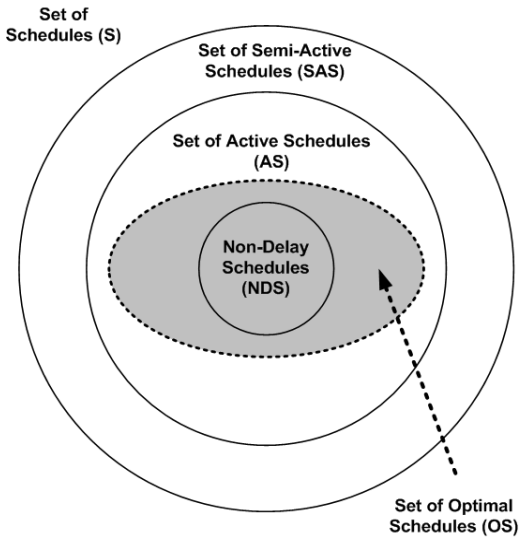


Figure 1: Relationship diagram of various schedules with regard to optimal schedules

The following section briefly introduces the Elitist GA and the incorporation of the parallel scheme into Elitist GA. Scheduling results using 30 standard problems are presented to examine which schedule generation scheme performs better. In doing so, the results obtained from the two schemes are compared to each other in terms of the project duration. Conclusions and recommendations are given.

3 PERMUTATION-BASED ELITIST GA

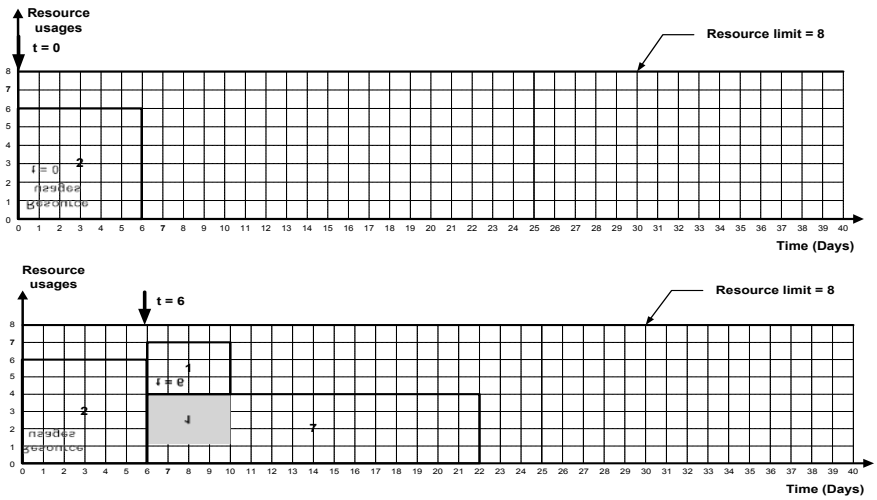
3.1 Elitist GA with Serial Scheme (Kim and Ellis 2008)

This section introduces Elitist GA to describe how the algorithm was developed and where the parallel scheme was incorporated for the purpose of the comparison. Elitist GA aims to search for the shortest duration of a project within the constraints of precedence relationships and limited resources. Since each activity needs to be performed only once, a real coded GA is suitable for solving the RCPSP, which is an order-based problem. Each individual solution for the RCPSP was represented in a chromosome, which shows an activity sequence such as {2, 6, 3, 8, 1, 4, 7, 9, 5, 11, 10} for 11 non-dummy activities. The chromosomes used in Elitist GA were real coded. Several operators implemented in Elitist GA include (1) the random number generator for producing an initial population, (2) the serial schedule generation scheme for calculating a fitness value of each individual, (3) the elitist roulette wheel selection operator for selecting a parent individual for the next generation, (4) the one-point crossover operator for exchanging parent string segments and recombining them to produce two resulting offspring individuals, and finally (5) the uniform mutation operator for playing a role of random local search, which searches regardless of the direction of learning to obtain a better solution.

3.2 Incorporating Parallel Scheme into Elitist GA

Elitist GA was designated to incorporate the parallel scheme in its second step along with the serial scheme. The parallel scheme is integrated into the Elitist GA to calculate the fitness value of an individual. The parallel scheme consists of at most J stages in each of which a set of activities is scheduled. A unique feature of the parallel scheme is that each stage n is associated with a schedule time t_n, where $t_m \leq t_n$ for $m \leq n$ holds. Due to this schedule time, the set of scheduled activities is divided into the following two subsets: Activities which were scheduled and are completed up to the schedule time are in the complete set (Cn), while activities which were scheduled, but which are at the schedule time still active, are in the active set (An). Also, the decision set (Dn) exists like the serial scheme. In contrast to the serial scheme, the parallel scheme contains all yet unscheduled activities which are available for scheduling with regard to precedence and resource constraints. The partial schedule of each stage is made by the activities in the complete set and the active set. The schedule time of a stage is equal to the earliest completion time of activities in the active set of the previous stage.

Each stage is made up of two steps: First, the new schedule time is determined and activities with a finish time equal to the (new) schedule time are removed from the active set and put into the complete set. This, in turn, may place a number of activities into the decision set. Second, one activity from the decision set is selected according to the order of the activity list representation and scheduled to start at the current schedule time. Afterwards, this activity is removed from the decision set and put into the active set. The second procedure is repeated until the decision set is empty, i.e., activities were scheduled or are no longer available for scheduling with regard to resource constraints. The parallel scheme terminates when all activities are in the complete set or active set.



At t=0,

Cn = { }
Dn = {2, 1, 3}
An = {2}

(a)

At t=6,

Cn = {2}
Dn = {7, 1, 6, 3}
An = {7, 1}

(b)

Figure 2: Parallel schedule generation scheme at (a) t=0 and (b) t=6 days

Figure 2 illustrates the parallel schedule generation scheme at t=0 and t = 6 days for the purpose of showing its process. The new schedule time is set to zero, as a starting point of scheduling the individual I= {2, 7, 1, 6, 4, 3, 8, 9, 5, 11, 10}. The first decision set {2, 1, 3} is created at time 0 stage because they are unscheduled and available for scheduling with regard to precedence relationships and resource constraints. All three activities, 2, 1, and 3, are considered to be scheduled at the same time. Activity 2 is selected first because it is placed in the first position of the decision set. Activity 2 then starts at time 0 for the duration of 6 because it does not have any predecessor and because its resource requirements (RR_2=6) do not exceed the resource availability (RA=8, which is constant throughout the project duration) through the period of its duration. Simultaneously, Activities 1 and 3 are considered to be scheduled at time 0, but they cannot start because of the limitation of available resources.

Next, the new schedule time is determined at time 6 based on the earliest completion time (Day 6) of activity in the active set {2} of the previous stage. At time 6 stage, the complete set {2} and the new decision set {7, 1, 6, 3} are generated. Activities 1 and 3 are contained in the previous decision set, while Activities 7 and 6 are newly included from the list of the individual considered because their predecessor, Activity 2, is completed. In other words, the new decision set should be ordered into {7, 1, 6, 3}, but not {1, 3, 7, 6} because the order of the activity list in the individual is {2, 7, 1, 6, 4, 3, 8, 9, 5, 11, 10}. Activities 7 and 1 are scheduled at time 7 for the duration of 16 and 4, respectively, because their resource requirements (RR_7=4 and RR_1+RR_7=3+4=7) do not exceed the resource availability through the period of their durations. However, the rest of the decision set cannot be scheduled due to the limitation of available resources. The parallel scheme continues up to the last activity in the individual, as the time increases.

4 COMPUTATIONAL RESULTS

Elitist GA is programmed using the JAVA™ programming language on the Windows® XP operation system. The parameters of Elitist GA include population size, crossover probability, and mutation probability for global search. Three different types of termination conditions can be chosen to stop the run of Elitist GA. They include the number of generations, timeout, and the number of unique schedules. In addition, each scheme can be set to examine which scheme performs well in Elitist GA.

4.1 Comparison of Resource Profile by Scheme

Elitist GA using two different schemes was tested on a case example associated with a single resource in order to demonstrate the robustness of the current approach. A construction project schedule was extracted from the work of Chan et al. (1996). Figure 3 shows the network information for the case example. The same parameter values as ones used in GA-scheduler (Chan et al. 1996) were employed for the equal comparison. Thus, the number of the population size was set to 50 and the total number of generations was set to 40 experimental runs so the total trial size of 2,000 was performed in Elitist GA. The crossover rate and mutation rate were set to 0.5 and 0.03, respectively. Elitist GA produced the project duration of 38 days, which is the same project duration as those obtained from GA-scheduler. Figure 4 illustrates the single resource profiles for the case example by scheme. Note that the chromosomes for the serial scheme and the parallel scheme are totally different, even though the resource profiles provide same information.

Activity	Precedence Relationships	Duration (Days)	ES	EF	TF	FF	RES.
A	-	4	0	4	10	0	3
B*	-	6	0	6	0	0	6
C	-	2	0	2	14	0	4
D	A	8	4	12	10	0	0
E	D	4	12	16	10	8	4
F	B	10	6	16	2	0	0
G*	B	16	6	22	0	0	4
H	F	8	16	24	2	0	2
I	C	6	2	8	14	14	4
J	E,H	6	24	30	2	0	5
K*	G,I	10	22	32	0	0	2
		Maximum resources					8

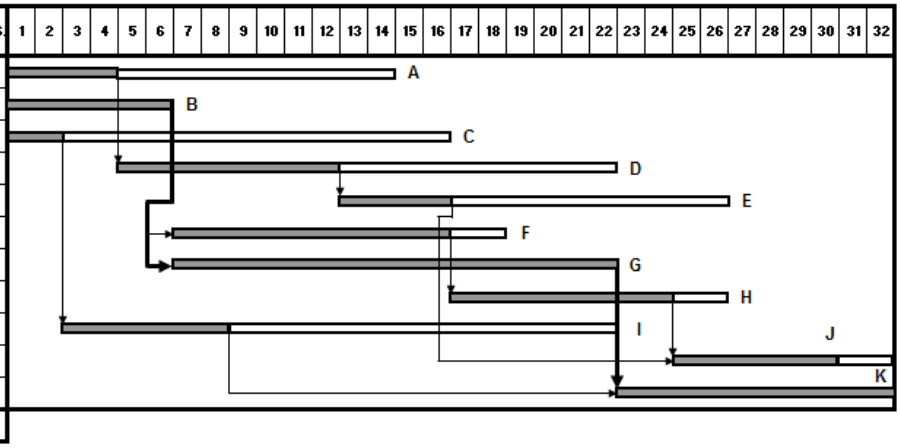


Figure 3: Network information for case example

ACT	DUR	Days																																							
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	**38**	39	40
A	4							3	3	3	3																														
B	6	6	6	6	6	6	6																																		
C	2																					4	4																		
D	8											2	2	2	2	2	2	2	2																						
E	4																													4	4	4	4								
F	10											2	2	2	2	2	2	2	2	2	2																				
G	16							4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4																		
H	8																							2	2	2	2	2	2	2	2										
I	6																							4	4	4	4	4	4												
J	6																																	4	4	4	4	4	4		
K	10																													2	2	2	2	2	2	2	2	2	2		
R1 (avail.)		8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
R1 (req.)		6	6	6	6	6	6	7	7	7	7	8	8	8	8	8	8	8	8	6	6	8	8	6	6	6	6	6	6	8	8	6	6	6	6	6	6	6	6	0	0

Figure 4: Results of single resource profiles for both serial and parallel scheme

4.2 Scheduling Results for Multiple Projects

This section describes a design experiment to test the performance of Elitist GA by scheme. Thirty problems were used to obtain the results of performance. The instance sets were obtained from the PSPLIB (Kolisch and Sprecher 1996). The instance sets used consist of 30 activities with up to four renewable resources. Each problem has been generated using a full factorial design of parameters, which determine the characteristics of the resource and precedence constraints. The instances were appropriate to be used because they were systematically generated by varying three parameters such as network complexity, resource factor, and resource strength.

The average fitness value, which means the project duration, was found according to the scheme through the run of Elitist GA. The input parameter values for Elitist GA were set as follows: the population size, crossover rate, and mutation rate were set to 50, 0.5, and 0.03, respectively. Elitist GA was terminated when it met 50 generations. Figure 5 compares the average fitness values (project duration) obtained from Elitist GA according to the scheme. Elitist GA using the serial scheme produced better project durations for thirteen projects out of thirty projects, which amount to 43.33%, than the ones using the parallel scheme. Only two pairs such as project No. 1 and No. 2, which amount to 6.67%, resulted in better project durations when using the parallel scheme. The rest of 30 projects, which amount to 50.00%, showed the same project duration regardless of the scheme employed in Elitist GA. The result indicates that, for most pairs of the average fitness values, Elitist GA using the serial scheme provides better solutions than the one using the parallel scheme. Their solutions need to be examined for optimality while the significance of the comparison needs to be made statistically.

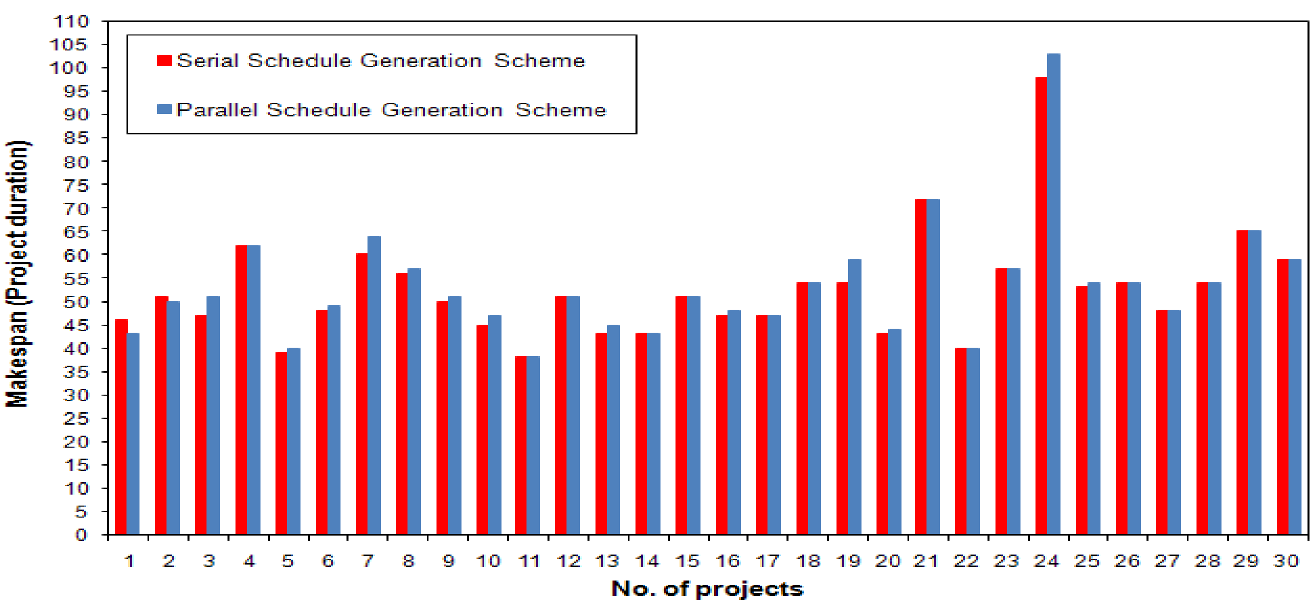


Figure 5: Comparison of average fitness values

5 CONCLUSIONS AND RECOMMENDATIONS

This paper proposed a methodology to compare the serial scheme with the parallel scheme in a decoding procedure of permutation-based genetic algorithm for construction resource scheduling. Elitist GA was designated to incorporate the parallel scheme in its second step along with the serial scheme for the computation of the fitness value of an individual. The scheduling results based on thirty standard problems indicate that, for most pairs of the average fitness values, Elitist GA using the serial scheme provides better solutions than the one using the parallel scheme. Their solutions need to be examined for optimality while the significance of the comparison needs to be made statistically. More project problems need to be solved to reinforce the results as well as to attempt to search for optimal solutions by exploring the solution space by the means of comparing the value obtained from the serial scheme and the one obtained from the parallel scheme simultaneously, then selecting the smaller one out of two values.

REFERENCES

Baar, T., P. Brucker, and S. Knust. 1999. Tabu-Search Algorithms and Lower Bounds for the Resource-Constrained Project Scheduling Problem. In S. Voss, S. Martello, I. Osman, and C. Roucairol, editors, *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Kluwer Academic Publishers, 1-8.

Boctor, F. F. 1996. Resource-Constrained Project Scheduling by Simulated Annealing. *International Journal of Production Research.* 34(8): 2335-2351.

Bouleimen, K., and H. Lecocq. 2003. A New Efficient Simulated Annealing Algorithm for the Resource-Constrained Project Scheduling Problem and Its Multiple Modes Version. *European Journal of Operational Research*. 149: 268-281.

Chan, W., D. K. H. Chua, and G. Kannan. 1996. Construction Resource Scheduling with Genetic Algorithms. *Journal of Construction Engineering and Management*. 122(2): 125-132.

Hartmann, S. 1998. A Competitive Genetic Algorithm for Resource-Constrained Project Scheduling. *Naval Research Logistics*. 45: 733-750.

Hartmann, S. 1999. *Project Scheduling Under Limited Resources: Models, Methods and Applications*. Lec. Notes in Eco. and Math. Sys. Springer-Verlag Berlin Heidelberg New York.

Kim, J.-L., and R. D. Ellis. 2008. Permutation Based Elitist Genetic Algorithm for Multiple Resource Constrained Project Scheduling Problem. *Journal of Construction Engineering and Management*. 134(11): 904-913.

Kohlmorgen, U., H. Schmeck, and K. Haase. 1999. Experiences with Fine-Grained Parallel Genetic Algorithms. *Annals Oper. Res.*, 90: 203-219.

Kolisch, R., and A. Sprecher. 1996. PSPLIB - A Project Scheduling Problem Library. *European Journal of Operational Research*. 96: 205-216.

Lee, J.-K., and Y.-D. Kim. 1996. Search Heuristics for Resource-Constrained Project Scheduling. *The Journal of the Operational Research Society*. 47(5): 678-689.

Pinson, E., C. Prins, and F. Rullier. 1994. Using Tabu Search for Solving the Resource-Constrained Project Scheduling Problem. Proceedings of 4th International Workshop on Project Management and Scheduling. Leuven, Belgium, 102-106.

Sprecher, A., R. Kolisch, and A. Drexl. 1995. Semi-Active, Active, and Non-Delay Schedules for the Resource-Constrained Project Scheduling Problem. *European Journal of Operational Research*, 80: 94-102.

AUTHOR BIOGRAPHIES

JIN-LEE KIM, Ph.D., P.E. is an assistant professor of Dept. of Civil Engineering & Construction Engineering Management at California State University, Long Beach. He was an assistant professor of Construction Engineering Technology at Missouri Western State University. He received his BE and first ME degrees in Architectural Engineering from Chungbuk National University, Korea and his second ME and PhD in Civil Engineering from University of Florida. He is a registered professional engineer in Florida. His research interests include information technology in construction, simulation-based resource scheduling, optimization techniques, and sustainable design and construction. He is a member of ASCE. His email address is <jin5176@gmail.com>.