Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

SIMULATION AND OPTIMIZATION FOR CONSTRUCTION REPETITIVE PROJECTS USING PROMODEL AND SIMRUNNER

Chachrist Srisuwanrat
Photios G. Ioannou

Civil & Environmental Engineering Department
2350 G.G. Brown
University of Michigan
Ann Arbor, MI 48109-2125, U.S.A.

Omer Tsimhoni

Industrial and Operations Engineering Department
1205 Beal Avenue
University of Michigan
Ann Arbor, MI 48109-2117, U.S.A.

ABSTRACT

We introduce a new approach, called Relative Start and Idle Time (RSIT), to solve probabilistic scheduling problems of construction repetitive projects. RSIT is a process of determining a range of input variables and employing optimization through simulation to solve scheduling problems. RSIT reduces the modeler's effort because it does not heavily rely on manual trial-and-error. The two primary advantages of this approach are: (1) it does not require additional solving algorithm code and (2) it does not impose unnecessary limitations on the simulation model in order to solve the scheduling problem. The new approach is presented in detail and applied to a real past repetitive project of four four-story buildings. Results from RSIT are evaluated and compared to the results from a deterministic approach. The example is modeled in ProModel and optimized in SimRunner.

1 INTRODUCTION

Repetitive projects are projects consisting of identical or similar units that require resources to work repetitively from the first unit to the last one. Examples of these projects are housing projects, highways, and tunneling projects. In these projects, some or all of the same set of activities are repeatedly performed from unit to unit by the same crews. To derive a practical and effective schedule for repetitive projects, three main constraints are applied: precedence, resource availability, and resource continuity constraints. Precedence constraints ensure that activities will be performed in technological construction orders, whereas resource availability constraints ensure the practical use of available resource. Resource continuity constraints are applied to maximize resource utilization by keeping resources working continuously without interruption. Accordingly, resource continuity constraints of these particular activities significantly affect the resource utilization of the overall project.

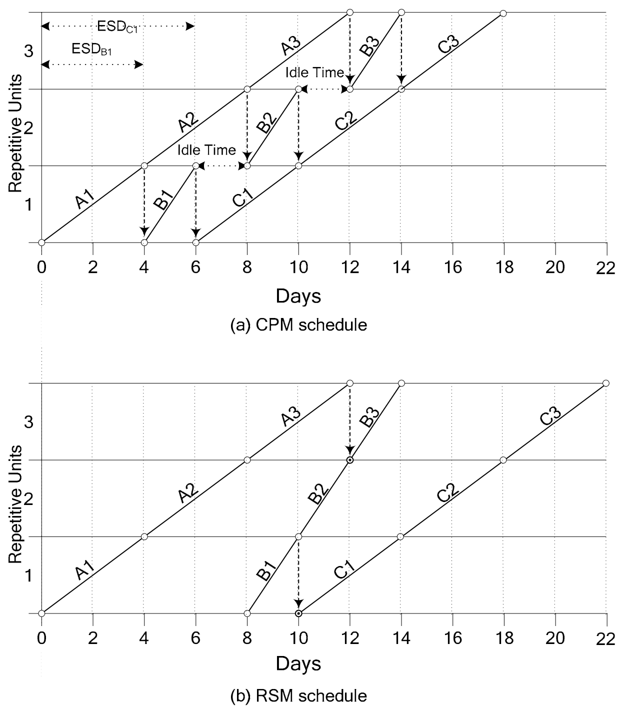


Figure 1: Increased project duration due to resource continuity constraints

Figure 1 is an example of a repetitive project consisting of 3 units. Figure 1a is schedule derived from critical path method (CPM). As shown in Figure 1a, Resource B for activity B has a total idle time of 4 days (Lag$_{B1,B2}$ and Lag$_{B2,B3}$). To eliminate idle time, two concepts of scheduling can be applied 1) balancing production rates and 2) postponing start date. Focusing on the latter, assuming production rates are fixed, activity B in Figure 1 must be postponed by 4 days to achieve continuous workflow of resource B and eliminate interruption, which exists in CPM schedule. This 4 days of postponement is called *time buffer*.

The idea of postponing activity start date to satisfy continuity constraints in repetitive projects can be found in Linear Scheduling Method (LSM) by Johnston (1981) and Repetitive Scheduling Method (RSM) by Harris and Ioannou (1998).

Satisfying resource continuity constraints usually lengthens project duration. As shown in Figure 1b, postponing activities from their early start date to eliminate idle time increases project duration from 18 days to 22 days. Evidently, tradeoff between resource continuity and project duration must be considered in order to achieve an optimum solution. The tradeoff can be objectively measured in several ways such as in terms of duration and cost.

In the past, many researchers have proposed simulation in order to study and solve repetitive scheduling problems. Examples of these studies are Harris and Evans (1977), Ashley (1980), Dabbas and Halpin (1982), Lutz (1990), AbouRizk and Shi (1994), Tommelein, Riley, and Howell (1999), Alves and Tommelien (2004), Ioannou and Srisuwanrat (2006), Sacks, Esquenazi, and Goldin (2007), and Srisuwanrat and Ioannou (2007). Nevertheless, only a few of these studies have focused on the tradeoff between maintaining and relaxing resource continuity constraints by using time buffer. The combination of resource allocation and probabilistic scheduling problems aggravates the problems. To be specific, trying to achieve resource continuity under variability of resource production rates and work quantities adds considerable difficulties. Moreover, it is difficult to quantify size and location of time buffer needed to optimize the overall project. Thus, simplification and limitation are imposed, or different type of buffers are used so that the problems can be solved.

In this paper we present a new method, the *relative start and idle time* (RSIT) that helps determining domain variables of resource arrival dates (size and location of time buffers) in order to optimize probabilistic scheduling problems of repetitive projects. Imposing minimal limitation on the problems and simulation model, the relative start and idle time method is a generalized concept that can be applied to simulation systems with optimization capability. An example of four four-story buildings which are built concurrently and sharing the same resources is presented here. The example has been developed and tested in ProModel and optimized using SimRunner. Results and discussions show the potential use of this new method in optimizing probabilistic scheduling problems of repetitive projects with complex and dynamic resource allocation.

2 SIMULATION OPTIMIZATION FOR REPETITVE PROJECTS

In simulation optimization for construction repetitive projects, there are two main distinct decision variables used: 1) number of resources and 2) buffers. Number of resources can be number of crews or equipment. When number of resources are used as decision variables, projects are optimized by balancing production rates among activities to achieve an optimum solution for the overall project performance. This is the same concept as Line-Of-Balance (LOB). The benefit of using number of resources to optimize repetitive projects is that the domain of input variables is finite and easy to determine. Capacity of available resources is limited and known prior to construction phase. Examples of studies using number of resources are Ashley (1980) using number of masonry crews and forklifts, and Dabbas and Halpin (1982) using number of hoists and buggies.

Despite the benefit and the popularity of using number of resources in optimizing repetitive projects there are circumstances in which balancing production rates cannot provide an optimal solution. These circumstances occur when there are 1) limited number of certain resources and/or 2) limited fastest or slowest production rates. Either one of these could cause a great difference in production rates between direct preceding and succeeding activities resulting in resource idle time and interruption in work flows. For example in Figure 1, if there are not enough resources to speed up activity A and activity B is already at the slowest production rate, balancing production rates between activity A and B is not possible. To improve the schedule, activity B must be postponed from its early start date (ESD); in other words, time buffer should be used to improve the work flow of activity B and also optimize the overall project schedule. Evidently, using number of resources solely leaves out the feasible benefit of postponing activities in optimization for construction repetitive projects.

Another type of decision variables in optimization for construction repetitive projects is *buffer*. Buffer can be in the form of inventory, finished work, work-in-process, distance, or time (Alves and Tommelein 2004). Buffer is used to prevent waste or interruption in activities stemming from the difference and variability in production rates. Certain type of buffer such as time and distance buffers provides such flexibility in adjusting schedule that may not be found in line-of-balance (adjusting production rates by altering number of resources). For examples, Harris and Evans (1977) used distance buffers in road construction in order to minimize cost and project duration; Lutz (1990) used time buffer to improve continuous work flow in repetitive activities; Alarcon and Ashley (1999) studied the effect of finished-work buffer size on production liability and project cost.

It is important to note that certain types of buffers are considered discrete numbers, which is similar to number of resources. Their domain variables can be determined easily. For example, the largest buffer size (upper bound) of finished-work buffer cannot exceed the number of total units such as floors in a building. On the other hand, time buffer does not have an upper bound and is therefore not easy to determine.

With respect to the simulation and optimization for repetitive construction projects, most studies can be grouped into 3 categories, shown in Figure 2. Figure 2a depicts the traditional method of optimizing repetitive projects. Users manually alter input parameters in their simulation to find an optimal solution. Number of resources and the concept of LOB are preferably used to optimize repetitive projects by this traditional approach. The main drawback of this approach is that it is time-consuming for the user and relies heavily on the user's intuition and trial-and-error. Moreover, if the trial-and-error process is not exhaustive, the solution may be far from optimum. Interestingly, this traditional approach is still in use by recent studies, regardless of the mentioned drawback and recent development in simulation software.

Figure 2b illustrates the second approach that integrates optimization into the system. Due to the advance in simulation and optimization software, increase in computational speed, integration between simulation and optimization engines becomes available (such as ProModel and SimRunner, Arena and OptQuest, or Stroboscope and ChaStrobeGA), and so does the improvement in automated programmability in simulation engines (such as ProModel ActiveX using Visual Basic for Applications.) The ProModel ActiveX connection enables the automation of creating complicated and large-scale simulation models for repetitive projects, without manually creating and coding simulation models. (See ProModel ActiveX User Guide, 2003, for more information).

The approach in Figure 2b eliminates the trial-and-error process performed by users in the traditional approach. The convenience and faster processing time provided by the integration and the automation between simulation and optimization allows users to explore a wider and deeper range of domain variables. Moreover, instead of employing exhaustive enumeration, the exploration (or optimization) is conducted intelligently by various optimization methodologies.

Examples of optimization methodologies used in simulation software are the evolutionary algorithms in SimRunner, the neural network in OptQuest, the genetic algorithm in ChaStrobeGA, or customized optimization algorithm such as by AbouRizk and Shi (1994).

The effectiveness of the second approach relies mainly on 1) type of decision variables, 2) the user-predefined domain of variables, and 3) the capability of the optimization engine. While studies using number of resources or similar as decision variables benefit from this approach, studies using time buffer cannot, because of the difficulty in defining domain of variables of time buffer, explained in the next section.

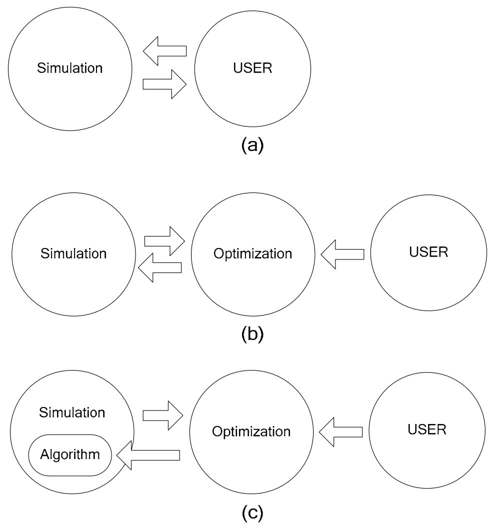


Figure 2: Three approaches in Simulation and Optimization for construction repetitive projects

Figure 2c illustrates a relatively new approach for optimizing repetitive construction projects. Different from the second approach, the third approach implements auxiliary algorithm in the simulation model. Examples of this approach are Sequence Step Algorithm (SQS-AL) (Ioannou and Srisuwanrat 2006) and Completed Unit Algorithm (CU-AL) (Srisuwanrat and Ioannou 2007a). In their studies, auxiliary algorithms, coded as part of the simulation code, determine resource arrival dates (time buffers) that provides better continuity of resource utilization. The algorithm proceeds with the process of solving the problem according to its concept and the user's predefined range of decision variables; confidence level in SQS-AL and number of completed units in CU-AL. The optimization part attunes decision variables within their given domains to find an optimum solution.

The reasons that the third approach and also the auxiliary algorithm are proposed are:

- Discrete-event simulation itself cannot guarantee the continuity in resource utilization.
- Difficulty in defining effective domains of input variables of resource arrival dates which are indefinite.
- The desire to speed up the optimization process by directionally reducing the domains of decision variables.

The effectiveness of the third approach relies on several factors such as the effectiveness of the auxiliary algorithm in solving the problem, extra processing time required by the algorithm, capability of optimization engine, and the interaction between the algorithm and the selected optimization methodology. If the algorithm is well constructed, this approach provides a better solution and reduces users effort tremendously. Moreover, it could scope down domain size of input variables, yet returning a satisfying result (see Srisuwanrat and Ioannou 2007b for examples about the effect of different auxiliary algorithms and decision variables on simulation optimization for repetitive project scheduling).

The drawback of the third approach is that the auxiliary algorithm may impose *additional* limitations on the problems and simulation model. In other words, the limitation is not incurred by the problems nor by the simulation model, but by the introduced algorithm. The algorithm may require the problems and simulation model to be constructed and solved in a certain way. For example, SQS-AL solves repetitive scheduling problems by calculating resource arrival dates in the order of sequence steps in which activities occur. The requirement of solving problems in sequence step orders is an additional limitation.

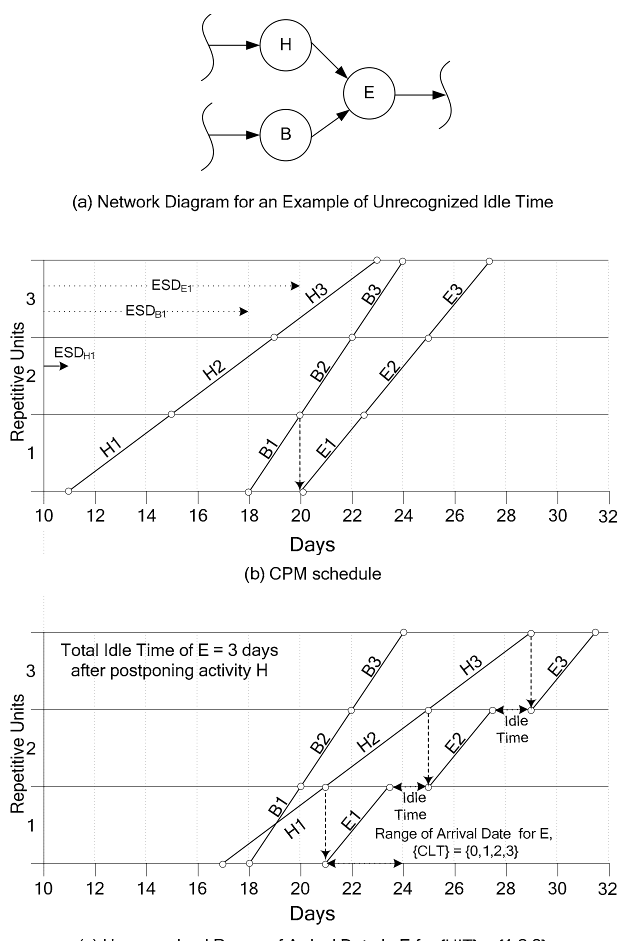
Due to the potential drawbacks of auxiliary algorithms, the authors believe that it is worth trying to solve the problems of probabilistic repetitive scheduling using optimization, without auxiliary algorithms. Thus, the authors have revisited the second approach, Figure 2b, and proposed a generalized method that tackles the difficulty in defining effective domain of input variables, which prohibits the second approach from using time buffer to optimize the repetitive project scheduling problems.

3 DIFFICULTIES IN DETERMINING DOMAINS OF RESOURCE ARRIVAL DATE VARIABLES

The main difficulty in optimizing repetitive projects is selecting input variables of resource arrival dates. The following questions must be answered before optimization:

- What is the minimum value of resource arrival date?
- What is the range of resource arrival date, measured from the minimum value?

Given that a simulation model of a repetitive project is created, running the model provides a schedule that is equivalent to an early start date as in the critical path method (CPM), which means that idle time between units is common. An early start date (ESD) of activities in CPM as *a constant* should not be used to determine the minimum value of resource arrival date, especially in optimization. Figure 1a shows that the start date of activity C should be adjusted automatically according to its predecessor' start date. Its start date changes from day 6 to day 10, not because of its idle time but because of the delay in activity B, due to the postponement in B in order to eliminate idle time of B. In other words, the start date, which is decision variables in optimization, should be assigned as a *relative value* to its predecessors' start date.



(a) Network Diagram for an Example of Unrecognized Idle Time

(b) CPM schedule

(c) Unrecognized Range of Arrival Date in E for {UIT} = {1,2,3}

Figure 3: Unrecognized Range of Resource Arrival Date

An example that a constant start date in CPM cannot be used as the minimum value of a resource arrival date is shown in Figure 3. Figure 3b is CPM schedule showing zero idle time in activities H, B, and E. However, after predecessors of H are postponed from their ESD to achieve resource continuity, activity H is, in effect, postponed as well, shown in Figure 3c. Accordingly, the minimum value of resource arrival date for activity E should be 21 days, instead of 20 days. Keep in mind that, the minimum value should be assigned as relative value, not a constant.

In Figure 3, postponing activity H incurs idle time of 3 days in activity E shown in Figure 3c, which previously does not exist in CPM schedule, Figure 3b. This situation is called "*unrecognized idle time*" in activity E. This situation confirms that the postponement period cannot be determined from a CPM schedule. One may assume from a CPM schedule in this case that there is no need to delay the start date of activity E. However, as shown in Figure 3b, activity E has an idle time of three days, and thus postponing activity E from zero to three days (from its minimum value which is relative to start dates of activity

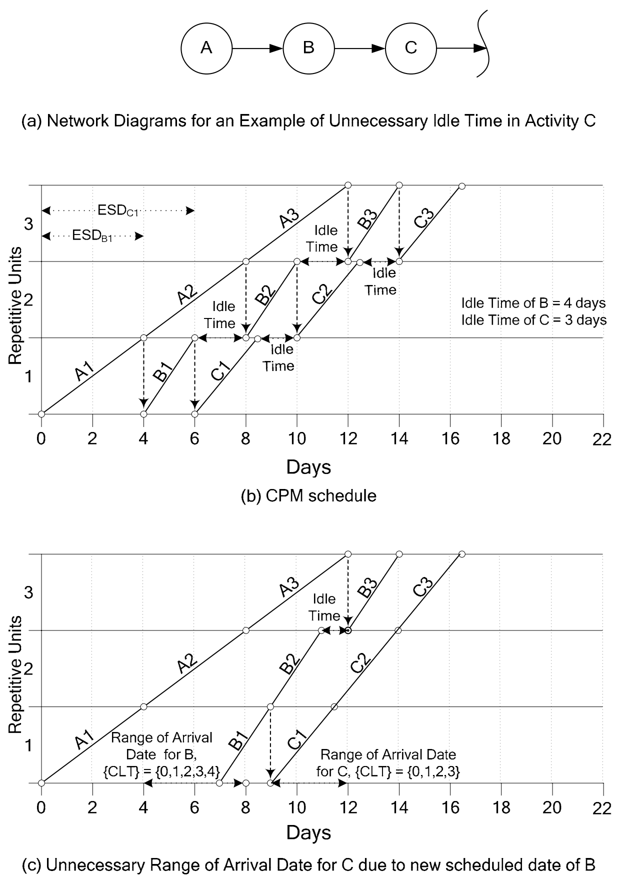


Figure 4: Unnecessary Range of Resource Arrival Date

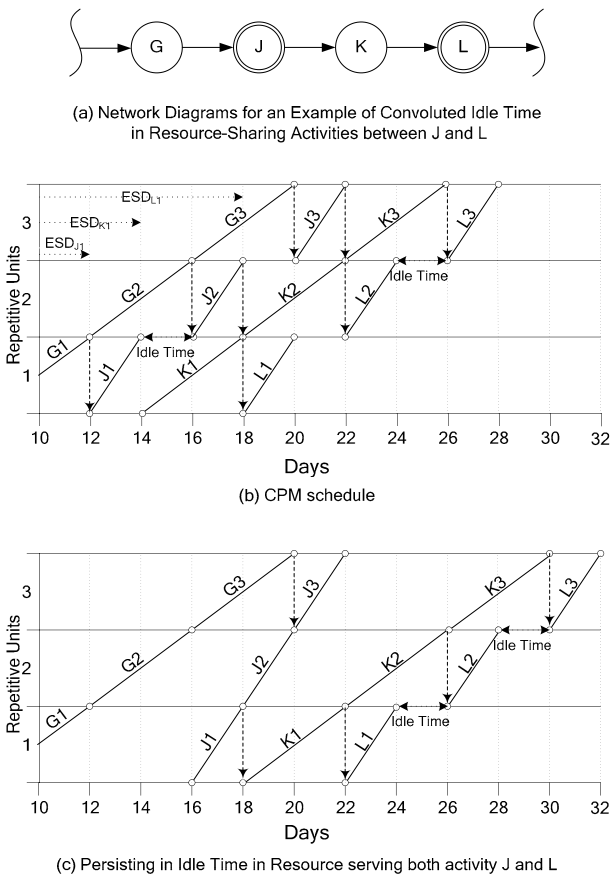


Figure 5: Convoluted Idle Time in Resource-Sharing Activities

H and B) should be tested in order to optimize this repetitive project. This range of zero to three days is the possible range of resource arrival date measured from its *relative* minimum value.

Note that the relative minimum value for activity E in Figure 3c is at the completion of H1, not a constant value of day 21.

Another difficulty in determining an effective range of resource arrival date is shown in Figure 4, called "*unnecessary range of arrival date*." The adverse effect of this situation is the elongation of the optimization process. This situation occurs when a delay in a predecessor eliminates idle time in its successor. In Figure 4, delaying activity B beyond three days will result in zero idle time in activity C. Thus, it is not necessary to consider the delay in activity C from its minimum relative value, if activity B is delayed for three or four days. However, it is recommended to consider the unnecessary range due to simplicity and also possible effect from variability, while realizing the existence of this situation.

Note that in Figure 4 the relative minimum values of activity B and C are at the completion of A1 and B1 respectively.

Figure 5 shows another situation that information from CPM schedule should not be used to establish domain of decision variables. In Figure 5, two activities J and L (resource-sharing activities) share the same resource, and postponing J incurs idle time in L. This situation is called "*convoluted idle time in resource-sharing activities*." This situation occurs when a delay in a preceding resource-sharing activity always incurs idle time in succeeding resource-sharing activity. In Figure 5, activity J and L have indirect dependence relationship and also share the same resource. By postponing activity J, its direct successor, activity K, will be postponed and will incur idle time in activity L, as shown in Figure 5b. From the example, it is imperative to detect the problem so that proper modification in the simulation model can be made prior to optimization.

Note that an easy solution for the problem shown in Figure 5 is assigning a dedicated resource for each repetitive activity.

4 RELATIVE START AND IDLE TIME (RSIT)

Relative Start and Idle Time is a means of determining the range of resource arrival dates, which serve as input variables for optimizing repetitive projects. Two main ideas of RSIT in selecting the parameters are that (1) arrival of resources of successors are relative to the arrival date of their predecessors and (2) the ranges of possible delays measured from a relative minimum value of arrival date are derived from idle time. This means if the arrival date of a resource is postponed, its successor's resource arrival date will be postponed as well. Using the relative minimum value of resource arrival date as a milestone, resources might be delayed *further* within the range of possible delay in order to minimize idle time in that resource. To apply RSIT, the following steps are required:

Step 1: Create a simulation model for the repetitive project.

Step 2: Set the simulation system to record the first date that a resource is utilized (SD) and its total crew idle time (CIT).

Step 3: Set the resource arrival date in the simulation using the equation 1.

$$R_2\text{Arrive} = R_1\text{Arrive} + \text{Offset} + R_2\text{CLTi}$$

R_2 is a resource of a successor, and R_1 is a resource of its predecessors. Offset is a user-selected constant used to offset the arrival of R_2 from its predecessors. For most cases, offset is equal to the duration of the preceding activity in the first unit. This means that the soonest that resource of successor can arrive is after a resource of its predecessor completes the first unit. Crew lead time (CLT) is a variable which will be determined in step 4, and is used to delay the arrival of resources in order to eliminate their idle time. At this point, CLT are set to zero.

Step 4: Run the simulation and determine the minimum value of start date, Min(SD), and the maximum value of idle time, Max(CIT), for each resource. Min(SD) is the earliest start date that a resource is utilized, whereas Max(CIT) is the maximum value of total crew idle time. The difference between Max(CIT) and Min(SD) is the possible range of delay measured from the relative minimum value, called maximum crew lead time, Max(CLT).

$$\text{Max(CLT)} = \text{Max(CIT)} - \text{Min(SD)}$$

Step 5: Run the simulation again, only this time assign Max(CLT) to CLTi. This step is required because delaying resource arrival date may increase idle time of another resource, as in the case of unrecognized idle time (Figure 3). This step determines whether the assigned Max(CLT) would cover the upper bound of lead time that eliminates idle time entirely. If idle time still exits, increase Max(CLT) by the amount of idle time.

Step 6: Repeat Step 4 and 5 until crew idle time (CIT) of all resources are zero. Note that if crew idle time still persists, the user should be aware of the possibility of a convoluted idle time in resource-sharing activities (Figure 5).

Step 7: Setup input variables for the optimization process by creating a set of integers ranging from 0 to Max(CLT). This set will be referred as {CLT}, whereas the selected input variable by optimization is referred as CLTi. Then, run the optimization engine. Note that, if the value Max(CLT) is so high that the process of optimization is unbearably time-consuming, it is suggested to use larger intervals, such as 0, 5, 10, and 15 (weekly basis), instead of using 0,1,2,…,14, and 15 (daily basis) for that particular resource.

5 AN EXAMPLE OF FOUR FOUR-STORY BUILDINGS

An example of four four-story buildings, constructed concurrently and sharing 20 resources, is used to study the viability and effectiveness of the proposed method, RSIT. The example and detail are obtained from Yang (2002). The simulation model and RSIT have been implemented in ProModel, and the optimization is performed by SimRunner. The results from RSIT and deterministic approach are compared at different variability (coefficient of variance) in activity duration to show the viability of RSIT.

Note that the simulation model and code for the example are automatically created by Excel macro employing ProModel ActiveX. After the simulation model and code are created, users starts RSIT processes from step 4 described in section 4.

5.1 Precedence Relationships, Duration, and Resources

The example project consisted of four four-story buildings, constructing concurrently and using the same set of 20 resources. Each building consisted of four identical floors with 20 repetitive activities each. This project may be considered as a repetitive project with 16 similar units. The precedence relationships, means of activity duration, and resource names are shown in Table 1.

Activity durations are assumed to follow uniform distribution with the means shown in Table 1 and standard deviations of each activity derived from multiplication of the means and a given coefficient of variance as shown in Table 4 to 8. An example of ProModel code for activity duration is shown below.

```
Duration = Uniform(Mean, Mean x Coefficient)
```

Table 1: Activities and Resources data

ID	Description	Predecessors	Mean (days)	Res.
5	Pile Fabrication	-	3	PF
10	Pile Driving	5	3	PD
15	Concrete Slab	10	1	CS
20	Concrete Walls	15	1	CW
25	Roof	20	1	RF
30	Rough-in	25	1	RI
35	Partition	30	1	PT
40	Paint Ceiling	35	1	PNC
45	Exterior Block Wall	40	1	XBW
50	Plaster Interior Wall	45	1	PLIW
55	Plaster Exterior Wall	45	1	PLXW
60	Paint Interior Wall	50, 55	1	PNIW
65	Floor Tile	60	1.5	FT
70	Bathroom Tile	60	1	BT
75	Window Installation	65	1	WI
80	Electrical Wiring	65, 70	1.5	EW
85	Bathroom Accessories	65,70	1.75	BA
90	Closet Installation	75, 80, 85	1	CI
95	Door Installation	90	1	DI
100	Cleaning	95	1	CL

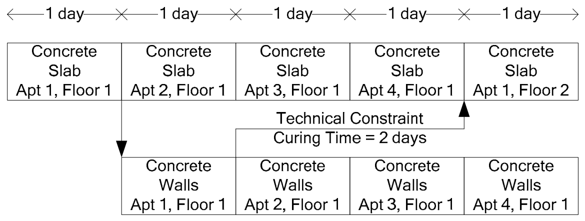

1 day
1 day
1 day
1 day
1 day
Concrete Slab Apt 1, Floor 1
Concrete Slab Apt 2, Floor 1
Concrete Slab Apt 3, Floor 1
Concrete Slab Apt 4, Floor 1
Concrete Slab Apt 1, Floor 2
Technical Constraint
Curing Time = 2 days
Concrete Walls Apt 1, Floor 1
Concrete Walls Apt 2, Floor 1
Concrete Walls Apt 3, Floor 1
Concrete Walls Apt 4, Floor 1

Figure 6: Lead time between concrete slab and wall

Pile Fabrication, Pile Driving, and Roofing activities are required only once per building. Most activities have finish-to-start relationships with zero lead time to their successors except for Concrete Slab, Concrete Wall, and Roof activities. Lead time between concrete slab and concrete wall in the preceding floor is two days due to curing time for concrete wall, shown in Figure 6. The precedence relationship between a concrete wall on the fourth floor and roof is the same as that of the concrete slab and concrete wall.

Resource CS and CW are allocated alternately among the four buildings. Resource CS will finish the first floor of all four buildings before starting to work on the second floor, shown in Figure 6. This is also the same for resource CW for concrete wall. By doing so, continuous resource utilization of resource CS and resource CW can be achieved.

Table 2: Offset constants and CLT variables

Resource	After the Arrival Of Resource	Offset	Max (CLT)	{CLTi}
PF	-	-	0	0
PD	PF	3	1	0-1
CS	PD	3	7	0-7
CW	CS	1	7	0-7
RF	CW	1	7	0-7
RI	RF	-10	8	0-8
PT	RI	1	8	0-8
PNC	PT	1	8	0-8
XBW	PNC	1	8	0-8
PLIW	XBW	1	8	0-8
PLXW	XBW	1	8	0-8
PNIW	Max(PLIW,PLXW)	1	8	0-8
FT	PNIW	1	2	0-8
BT	PNIW	1	8	0-8
WI	FT	1	10	0-10
EW	Max(FT, BT)	1	3	0-3
BA	Max(FT, BT)	1	2	0-2
CI	Max(WI, EW, BA)	1	13	0-13
DI	CI	1	13	0-13
CL	DI	1	13	0-13

5.2 Input Variables for Resource Arrival Date

After a simulation model is created and run, minimum start date, Min(SD), and maximum crew idle time, Max(CLT) for each resource are collected. At this point, resources are scheduled to the site at time 0. Max(CLT) is the difference between Max(CIT) and Min(SD) is shown in Table 2. The possible range of input variable CLT for optimization is shown in the last column of Table 2.

Offset constants for each resource are set equal to the duration of its predecessor. Resources of a successor cannot arrive sooner than its predecessors' resource, and there is no need for an optimization tool to test such cases. The only exception here is the offset of resource RI for rough-in activity. Although its direct predecessor, which is the roofing activity, has a duration of 1 day, the offset of RI is set at minus ten. The reason is that Concrete Wall activity, the nearest indirect predecessor of Rough-In activity, affects the possible arrival date of resource RI as much as roofing activity. Since Roofing activity is required only in the last floor, while concrete wall activity is required in every floor, the arrival of Roofing activity, solely, may not effectively determine the arrival of resource RI. Thus, the domain of arrival date for resource RI is expanded to cover to the completion of concrete wall activity in the first unit. Another alternative to this is setting the arrival of resource RI relative to CW, instead of resource RF. These are examples of the arrival date for resource CS (Concrete Slab) and CI (Closet Installation).

```
    Arrival of Resource CS = Arrival of Resource
PD + 3 days + CLT of CS
```

```
    Arrival of Resource CI = Max(Arrival of WI,
EW, BA) + 2 day + CLT of CI
```

Note that CLT of resources are input variables which will be selected from {CLTi} by optimization process in SimRunner.

5.3 Optimization in SimRunner

Optimization for the example is performed by SimRunner, which uses evolutionary algorithms. The objective function, shown below, focuses on reducing project duration and total idle time.

```
    Objective function =
    Minimize(Project duration + Total Idle Time)
```

Project duration is the overall duration in days for this four four-building project. Total idle time is the sum of idle time in resources. This simple objective is used for the sake of illustration and simplicity so that audiences can easily interpret and compare the outcome from RSIT and the deterministic repetitive scheduling method (RSM).

To optimize this repetitive project, the required input factors (or parameters) in SimRunner are {CLTi} for every resource, shown in Table 2. The default value and lower bound for each resource is set at zero, whereas the upper bound is set at Max(CLT), and their data type is integer.

Optimization parameters which are input to SimRunner are shown in Table 3. Default options for the optimization profile and convergent percentage are used, which are "moderate" and 0.01 respectively. Although a larger number of replications is desired, SimRunner only allows 100 replications per experiment. Run time is set at 2000 hours, sufficient to cover the project in the worse delay scenario incurred by delaying the arrival of resources. Note that the working period for this project is eight hours per day, and five days per week. Finally, confidence level is set at 95. For more information about SimRunner, please see SimRunner User Guide (2002).

Table 3: Optimization and Simulation Options

Option	Selected Value
Optimization Profile	Moderate
Convergence Percentage	0.01
Replications per experiment	100
Warm-up time (hr)	0
Run time (hr)	2000
Confidence level	95

Table 4: Results from RSM and RSIT

Scenario with different schedule	Coefficient of Variance	Project Duration + Total Idle Time (days)	
		Deterministic RSM	Optimized RSIT
1	0	65 (65, 0)	70 (56,14)
2	10	73 (65,8)	76 (61,15)
3	20	81 (65,16)	77 (62,15)
4	30	91 (66,25)	81 (66,15)
5	40	100 (66,34)	87 (69,18)
6	50	109 (67,42)	93 (71,22)

6 RESULTS AND DISCUSSION

Average values of project duration plus total idle time from a deterministic RSM and an optimized RSIT at different coefficients of variance are shown in Table 4. Six scenarios with different coefficients of variance are established to investigate the effectiveness of RSM and to RSIT under different scenarios.

For the deterministic RSM, resource arrival dates are derived from RSM schedule with a coefficient of variance of zero (SD = 0). These resource arrival dates are, then, used under the other five scenarios, where coefficient of variance ranges from 10% to 50%. Since deterministic RSM uses average durations in scheduling projects, it results in the same resource arrival dates even under different variability. As shown in Table 4 for RSM, as variability (coefficient of variance) in activity duration increases, idle time increases. This is because the schedule and time buffer are not changed according to the variability that incurs interruption between repetitive activities and idle time in resources. Increasing coefficients of variance results in greater idle time. However, project duration for RSM remains approximately the same because of the fixed resource arrival dates and characteristics of controlling sequence and repetitive activities in the late sequence steps. For the impact of controlling sequence, critical activities on project duration in repetitive scheduling, see Harris and Ioannou (1998).

Note that results from RSM in Table 4 show the adverse effect of changes in variability toward idle time. From scenario 1 to 6, the ratios of total idle time and project duration from RSM range from 0% to 63%.

For the optimized RSIT, resource arrival dates are derived from scenario 2 with 10% coefficient of variance. Then, they are used for the optimization of the rest of the scenarios to derive resource arrival dates for each scenario; there is no change in CLTs and optimization options. The results from RSIT in Table 4 show that as variability increases, relaxing continuity constraints becomes more necessary in order to minimize project duration and idle time. This is also true in increasing idle time. Comparing results between RSM and RSIT, it is evident that maintaining and relaxing continuity benefit the overall project and idle time.

Another benefit from optimized RSIT is that it provides more than one solution for each scenario. Rather than having only one solution, optimized RSIT provides several solutions close to the optimum solution shown in

Table 4. This set of solutions could be used for further analysis in order to achieve a practical schedule that suits the company, the resource availability, and other considerations.

From Table 4, the ratios of idle time and project duration from the optimized RSIT range from 22% to 31%, while those of RSM range from 0% to 63%. This is because solutions from the optimized RSIT are derived from optimization of each scenario, resulting in different resource arrival dates for that specific scenario. In contrast, results from RSM are derived from the same set of resource arrival dates. Accordingly, to compare the solutions from RSIT to RSM, RSIT solutions from each scenario must be tested at other scenarios as well.

Table 5 shows the results of the optimized schedule from RSIT at different scenarios, different coefficients of variance. Bold letters indicate the scenario from which the optimized schedule is derived. Ideally, the results obtained from the schedule optimized specifically from that scenario should provide the best result, in terms of minimizing the sum of project duration and idle time. For example, schedule 3 derived from scenario 3 yields the best results compared to other schedule under scenario 3. This is also true for the schedule 3, 4, and 6 derived from scenario 4, 5, and 6, respectively.

However, the results in Table 5 show that the assumption that the optimized RSIT schedule will give the best results under its scenario is not true for the optimized RSIT schedule for scenario 1 and 2 (schedule 1 and 2). Schedule 1 and 2 do not provide the best results under their scenarios, shown in Table 5. For example, under scenario 2 (10% coefficient of variance) the optimized RSIT schedule derived from scenario 3 and 4 performs better than the one derived from scenario 2 itself. This discrepancy is believed to be from the unexpected interaction between the nature of repetitive projects, the employed optimization methodology, and the selected optimization options. Remember that CLTs used to derive the schedules are the same and derived from scenario 2; therefore, it would have been expected that the optimized schedule derived from scenario 2 would, at least, provide the best result for scenario 2. Nevertheless, this is not the case. Further study must be carried, especially on optimization for repetitive projects and the selection of optimization options, in order to understand such situations and improve RSIT.

Table 6 shows the ratios of idle time and project duration. As mentioned, the increase in variability incurs greater idle time and also the ratios between idle time and project duration, since project duration is roughly the same. Table 6 suggests that if an increase in variability is not taken into account while establishing the project schedule and if it indeed occurs, idle time could change variably. On the other hand, a smaller ratio of idle time and project duration can be obtained if variability is actually

Table 5: Results from RSM and RSIT

Optimized RSIT Schedule from Scenario	Test with Coefficient of Variance					
	0%	10%	20%	30%	40%	50%
	Project duration plus idle time (days)					
1 (0%)	**70**	81	92	103	113	124
2 (10%)	71	**76**	82	90	99	108
3 (20%)	67	72	**77**	84	94	103
4 (30%)	73	75	77	**81**	88	95
5 (40%)	76	78	80	84	**87**	95
6 (50%)	78	80	81	85	88	**93**
RSM	65	73	81	91	100	109

Table 6: Ratios of Idle Time and Project Duration

Optimized RSIT Schedule from Scenario	Test with Coefficient of Variance					
	0%	10%	20%	30%	40%	50%
	Ratio of Idle Time and Project Duration (%)					
1 (0%)	**25**	43	60	77	94	111
2 (10%)	18	**24**	34	46	59	73
3 (20%)	12	17	**25**	35	51	64
4 (30%)	11	13	17	**22**	31	41
5 (40%)	12	14	17	21	**26**	37
6 (50%)	11	13	15	19	24	**31**
RSM	0	12	25	38	52	63

Table 7: Comparison of Results between RSM and RSIT

Optimized RSIT Schedule from Scenario	Coefficient of Variance (%)						Avg.
	0	10	20	30	40	50	
	Results from RSIT minus RSM (days)						
1 (0%)	**5**	8	11	12	13	15	11.8
2 (10%)	6	**3**	1	-1	-1	-1	0.2
3 (20%)	2	-1	**-4**	-7	-6	-6	-4.8
4 (30%)	8	2	-4	**-10**	-12	-14	-7.6
5 (40%)	11	5	-1	-7	**-13**	-14	-6.0
6 (50%)	13	7	0	-6	-12	**-16**	-5.4

less than expected. As shown in Table 6 for the RSIT solution of scenario 6, if the coefficient of variance is 0% or 10%, instead of 50%, the ratio of idle time and project duration decreases from 31% (22 days) to 11% (8 days), while the expected project duration changes from 71 to 70 days.

Table 7 shows a simple comparison between schedules derived from RSM and RSIT by subtracting the results (project duration plus idle time) from RSIT by RSM. The smaller the calculated value, the better the solution from optimized RSIT compared to deterministic RSM. If the coefficient of variance cannot be accurately predicted, however, known that it would range between 0% and 50%, RSIT solution from scenario 4 should be used, since it gives the lowest average value of the subtraction between RSIT and RSM.

7 CONCLUSION

This paper introduced an alternative method for optimizing probabilistic repetitive projects in simulation, called relative start and idle time (RSIT). The method does not impose limitations on repetitive project scheduling problems nor does it implement auxiliary algorithms in the simulation model or code. Leaving the simulation model as intact as possible, the method focuses on determining effective input variables for optimization. It exploits the nature of repetitive activities and the advance in simulation and optimization software in order to optimize the problem. In this paper, ProModel and SimRunner are used to illustrate the application of RSIT. The results show that the method can optimize the problems of probabilistic repetitive scheduling by maintaining and relaxing resource continuity constraints in order to obtain an optimum solution.

REFERENCES

AbouRizk, S., and Shi, J., 1994. Automated Construction-Simulation Optimization. *Journal of Construction Engineering and Management*, ASCE, June 1994, 374-385.

Alarcon, L.F., and Ashley, D.B., 1999. PLAYING GAMES: Evaluating the Impart of Lean Production Strategies on Project Cost and Schedule. In *Proceedings of the 7th International Group of Lean Construction*.

Alves, T.C.L., and Tommelein, I.D., 2004. Simulation of Buffering and Batching Practices in the Interface Detailing Fabrication-Installation of HVAC Ductwork. In *Proceedings of the 12th International Group of Lean Construction*.

Ashley, D.B.,1980. Simulation of Repetitive-Unit Construction. *Journal of Construction Division*, ASCE, Vol. 106, No. CO2, June 1980, 185-194.

Dabbas M.A.A., and Halpin, W.H. 1982, Integrated Project and Process Management. *Journal of Construction Division*, ASCE, Vol. 103, No. CO3, Sept 1977, 405-414.

Johnston, D.W., 1981. Linear Scheduling Method for Highway Construction. *Journal of Construction Division*, ASCE, Vol. 107, No. CO2, June 1981, 247-261.

Harris, F.C., and Evans, J.B., 1977. Road Construction – Simulation Game for Site Managers. *Journal of Construction Division*, ASCE, Vol. 103, No. CO3, Sept 1977, 405-414.

Harris, R.B., and Ioannou, P.G., 1998. Scheduling projects with repeating activities. *Journal of Construction Engineering and Management*, ASCE, July/Aug 1998, 269-278.

Ioannou, P.G., and Srisuwanrat, C., 2006. Sequence Step Algorithm for Continuous Resource Utilization in Probabilistic Repetitive Project, In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1731-1740.

Lutz, J.D., 1994. Planning of Linear Construction Projects using Simulation and Line of Balance. Ph.D. Dissertation, Civil Engineering Department, Purdue University.

ProModel Corporation. 2002. *SimRunner User Guide*, Utah.

ProModel Corporation. 2003. *ProModel ActiveX User Guide*, Utah.

Sacks, R., Espuenazi, A., and Goldin, M., 2007. LEAPCON: Simulation of Lean Construction of High-Rise Apartment Buildings. *Journal of Construction Engineering and Management*, ASCE, July 2007, 529-539.

Srisuwanrat, C., and Ioannou, P.G., 2007. Optimization of Probabilistic Repetitive Projects using Completed Unit and Genetic Algorithms, In *Proceedings of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M. -H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 2151-2158.

Srisuwanrat, C., and Ioannou, P.G., 2007. The Investigation of Lead-Time Buffering Under Uncertainty Using Simulation and Cost Optimization, In *Proceedings of the 15th International Group of Lean Construction*.

Tommelein, I.D., Riley, D.R., and Howell, G.A., 1999. Parade Game: Impact of Work Flow Variability on Trade Performance. *Journal of Construction Engineering and Management*, ASCE, Sep/Oct 1999, 304-310.

AUTHOR BIOGRAPHIES

CHACHRIST SRISUWANRAT is Ph.D. student in Construction Engineering Management at the University of Michigan. His research is in the area of resource utilization and construction simulation under the direction of P.G. Ioannou. His e-mail is <csrisuwa@umich.edu>.

OMER TSIMHONI is an Adjunct Assistant Professor in the Industrial and Operations Engineering Department at the University of Michigan, where he teaches simulation to undergraduate and graduate students. He is an Assistant Research Scientist at the University of Michigan Transportation Research Institute (UMTRI), where he studies the human factors of driving. His research interest is in the application of simulation and optimization tools to cognitive and physical modeling of human performance. His e-mail is <omert@umich.edu>.

PHOTIOS G. IOANNOU is Professor of Civil and Environmental Engineering at the University of Michigan. He received a Dipl. Civil Eng. from the National Technical University of Athens, Greece, in 1979, and a SMCE and Ph.D. in Civil Engineering from MIT in 1981 and 1984. His research is in construction engineering and management, and in particular in decision support systems and construction process modeling and simulation. His e-mail is `<photios@umich.edu>` and his website is `<www.engin.umich.edu/cem/Ioannou>`.