PROBABILISTIC SCHEDULING FOR REPETITIVE PROJECTS

Photios G. Ioannou[1] and Chachrist Srisuwanrat[2]

ABSTRACT

The sequence step algorithm for probabilistic scheduling of repetitive projects is a generalized methodology for scheduling projects with activities that repeat from unit to unit and have probabilistic durations. In simple terms it can be compared to PERT but for resource-constrained scheduling. The sequence step algorithm addresses for the first time the problem of scheduling repetitive projects with probabilistic activity durations while keeping resources (crews) employed continuously. This algorithm can be implemented in most general-purpose simulation systems. The algorithm is presented in detail and is applied to an example project with 7 activities and 4 repetitive units using a simulation model developed in Stroboscope, an activity-based simulation system. Numerical and graphical results help explain the algorithm and provide insight into the underlying tradeoff problem between reducing the expected crew idle time and increasing the expected project duration.

KEY WORDS

Scheduling, repetitive projects, linear projects, line of balance, probabilistic scheduling, resource continuous constraints, uninterrupted work flow, simulation.

INTRODUCTION

Multiunit projects are commonly found in construction where identical or similar units require repetitive work from unit to unit. Examples of multiunit projects are multistory buildings, housing projects, highways, and tunnelling projects. In these projects, the same activities are repeated from unit to unit by the same crews. For example, in a multistory building, one crew installs interior partition studs from floor to floor, while another crew follows and installs drywall. The installation of drywall in the 4th floor, for example, requires that the studwork for the 4th floor has been finished (a technological constraint) and also that the drywall crew has finished its work on the 3rd floor (a resource constraint). Of particular interest in scheduling repetitive multiunit projects is the ability to keep crews working continuously without interruption. Otherwise, crews experience periods of idle time where they receive pay without producing output. Thus, the uninterrupted (i.e., continuous) utilization of resources is of prime importance in scheduling repetitive work.

Figure 1 illustrates the problem using a production diagram for three activities, A, B, and C, that are repeated over three identical units (e.g., floors). These activities must be performed sequentially, one after the other. The work in each activity is performed by a separate crew. Figure 1a is a production diagram where activities are allowed to start as

[1] Professor, Civil and Environmental Engineering Department, Univ. of Michigan, MI 48109-2125, Phone +1 734/764-3369, FAX +1 734/764-4292, photios@umich.edu

[2] Doctoral Candidate, Civil and Env. Engineering. Department, Univ. of Michigan, MI 48109-2125, Phone +1 734/764-3369, FAX +1 734/764-4292, christ_cv@hotmail.com

early as possible as is typical in CPM and in most simulation models. Figure 1b is a production diagram where activities are deliberately delayed using the Repetitive Scheduling Method (RSM) to ensure the uninterrupted utilization of resources.

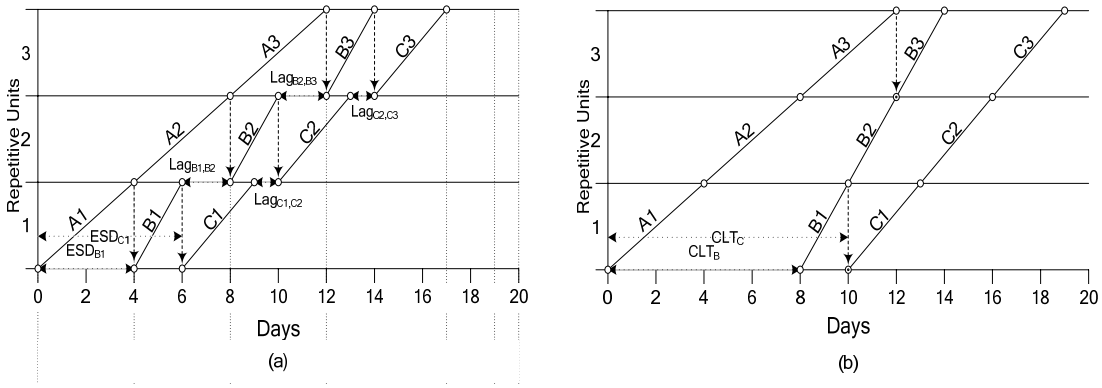

Figure 1: CPM vs. RSM Scheduling

As shown in Figure 1a, when activities A, B, and C, start on their early start dates (ESD), their differences in unit production rates (slopes) result in lags ($Lag_{I,J}$) that represent idle unproductive time for the respective crews. In practice, the crews for these activities are paid from the point they start work in the first repetitive unit, to the completion of the activity in the last unit. Thus, crew B is paid for 10 days (even though it works for only 6 days) and crew C is paid for 11 days (even though it works for only 9 days).

As shown in Figure 1b, delaying the arrival of crews B and C by CLT_B and CLT_C respectively, ensures that the respective crews work continuously without interruption. The tradeoff for saving 6 crew-days over Figure 1a is that the project duration has increased from 17 to 19 days. In most practical cases, the savings in crew costs far outweigh the slight increase in project duration.

Over the years, several methods have been proposed to solve the resource continuity problem with similar approaches that either postpone the start dates of activities or alter the number of resources (crew sizes) to balance activity production rates. Harris and Ioannou (1998) unified these methods into RSM and showed how to provide solutions for repetitive projects with *deterministic* activity durations using a graphical approach.

The resource continuity problem when activity durations are *probabilistic*, however, is considerably more difficult and has not been addressed before. Probabilistic activity durations are quite common in repetitive work and stem from variability in *crew production rates* and uncertainty in the *amount of work* to be performed by each activity in each repetitive unit. Probabilistic activity durations introduce a hard-to-quantify tradeoff between activity start lead-times, the probability distributions for work discontinuities and the probability distribution for project duration.

Tackling this problem using simulation requires a fine balance between the nature of resource-based models (where an activity starts immediately if the required resources are available) and the idea of deliberately postponing activity start dates (where activities do not start before a specified lead-time has elapsed, even though resources may be available). Simulation models that rely only on precedence constraints operate in the same manner as the critical path method in that they allow activities to start as early as

possible. Thus, they do not guarantee continuity in resource usage. Continuous resource utilization can only be achieved by deliberately holding the resources back by appropriately chosen lead times to make the work continuous without overly extending project duration (Ioannou and Likhitruangsilp 2005).

In this paper we present a new methodology, the *sequence step algorithm*, SQS-AL, for scheduling repetitive projects with probabilistic activity durations while maintaining continuous resource utilization. SQS-AL is based on generalized concepts that can be applied to most general-purpose discrete-event simulation systems. The example project presented here has been developed using Stroboscope, an activity- and resource- based simulation system (Martinez and Ioannou 1999). Numerical and graphical results help explain the algorithm and provide insight into the underlying tradeoff problem.

SEQUENCE STEPS

In project scheduling, a precedence diagram (also called an activity-on-node network) uses nodes (circles or rectangles) to model activities (tasks or work). Nodes are connected with lines (links) that represent precedence relationships between activities. An example precedence diagram with seven activities appears in Figure 2.

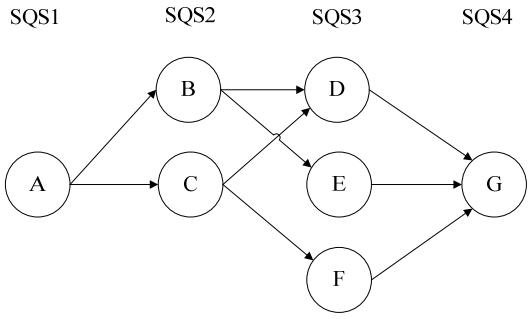

Figure 2: Precedence Diagram for Single Unit

An important property of activity-on-node networks is that they can be drawn in an organized manner using visual columns so that all precedence links have a clear left-to-right direction (Figure 2). This way, an activity to the left of a link represents a predecessor, while an activity on the right represents a successor. For example, the activities in Figure 2 belong to four visual columns, called *sequence steps*, indicated as SQS1, SQS2, SQS3, and SQS4. Every activity in a precedence diagram belongs to a *particular* sequence step, which is formally defined to be the *left-most* visual column in which the activity may be drawn and still maintain left-to-right precedence relationships within the network. Another way to find the sequence step for each activity is to assume that all activities have a duration of 1 and to perform the traditional CPM forward pass. The sequence step for each activity is then given by the resulting early finish dates.

The links in an activity-on-node network that has been organized by sequence step do not require arrow heads because all links have a left-to-right direction. In fact, this is the approach taken by most modern scheduling software.

An important property of precedence diagrams drawn by sequence step is that the traditional CPM scheduling calculations can indeed be performed by sequence step: i.e., left-to-right for the forward pass and right-to-left for the backward pass. Calculations for

activities in the same sequence step can be performed in any order (e.g., top to bottom). As explained below, this property is at the heart of the proposed *sequence step algorithm*.

CREW LEAD TIME AND CREW IDLE TIME

Crew lead time (CLT) is the *chosen* lead time by which the first start date of the corresponding activity is delayed. Crew idle time (CIT) is the *resulting* time a crew does not perform work during its total period of employment.

In the beginning of the proposed *sequence step algorithm*, CLT = 0, i.e., all crews are assumed available for work at time 0. Thus, the initial CIT is the early start date (e.g., ESD_{B1}) plus the sum of idle times from unit to unit (e.g., $Lag_{B1,B2} + Lag_{B2,B3}$).

As shown in Figure 1b, in order to eliminate crew idle time (CIT), it is necessary to delay the start of activities B and C by crew lead times CLT_B and CLT_C (as measured from the start of the project), respectively. Thus, to eliminate idle time for an activity, the crew lead time must equal or exceed its crew idle time (CLT >= CIT).

SEQUENCE STEP ALGORITHM

The sequence step algorithm schedules repetitive projects with probabilistic activity durations that reflect uncertainties in resource productivity or differences in work amounts among units, so that crews may work continuously without interruption. The algorithm consists of three general steps (Ioannou and Srisuwanrat 2006).

The first step is to simulate the network and collect crew idle times (CIT) for each activity in each project replication. After performing a number of replications, the collected CIT samples are used to construct cumulative relative frequency distributions, one for each activity. For example, in sequence step 1, after a total of 10,000 replications, we can arrange the 10,000 crew idle times for activity B in a cumulative relative frequency graph from which we can find for example that $P[CIT_B < 55 \text{ days}] = 94.27\%$.

In the second step, we select a desired confidence level (cumulative probability) for the crew idle times collected in the first step and we assign the corresponding time value to be the duration of crew lead time (CLT) for the activities in the next sequence step. For example, if for activity B 9,427 crew idle times out of 10,000 were less than 55 days, then assigning $CLT_B = 55$ should give a confidence of 94% that crew B should be able to work continuously. This assigned CLT (e.g., 55 days for crew B) remains constant through the end of the algorithm.

In the third step of the algorithm we reset the simulation model and clear all previously collected crew idle time (CIT) statistics for all activities. Using the already assigned crew lead times (CLT) for all activities in previous sequence steps, we move to the next sequence step and repeat the first and second algorithm steps until we reach the last sequence step.

Figure 3 is the flowchart of the sequence step algorithm. As shown in the flowchart, the algorithm contains two nested loops. The inner loop is the project replication loop while the outer loop is the sequence step loop. The inner loop represents step one, where the outer loop represents steps two and three. nSQSs and nReps are the total number of sequence steps and the total number of replications. SQS and Rep are the current sequence step and replication.

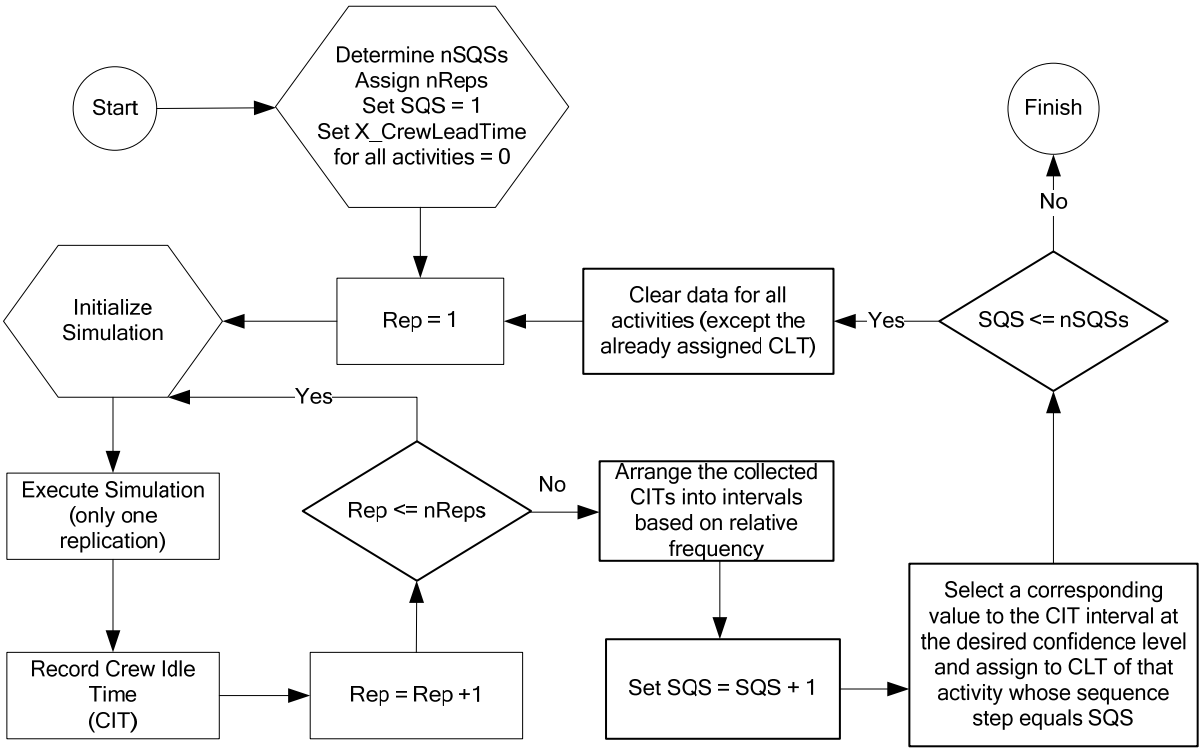

Figure 3: Sequence Step Algorithm Flow Chart

By repeating these three algorithm steps, we collect samples of crew idle times (CIT) that reflect the already chosen crew lead times (CLT) for activities on this and previous sequence steps. In turn, these CIT are used to select crew lead times (CLT) for the activities on the next sequence step using the appropriate confidence level. These CLT are then assigned to the activities in the next sequence step and the process is repeated.

EXAMPLE PROJECT

An example repetitive project consisting of 4 similar but non-identical units, with 7 activities each, is used to demonstrate the application of the sequence step algorithm. The example model and the sequence step algorithm have been implemented using the Stroboscope system.

PRECEDENCE NETWORK AND INPUT

The example project includes 4 non-identical units (e.g., floors, houses, etc.) each requiring 7 work activities (A, B, C, D, E, F, and G) performed by different crews. The activity-on-node network for each repetitive unit is the same and appears in Figure 1. As indicated in Figure 1, activity A is in sequence step 1, activities B and C are in sequence step 2, etc. The amount of work for each activity in each of the 4 units is different and is shown in Table 1. For example, the work amounts for activity A in units 1, 2, 3, and 4 are 100, 250, 150, and 200 work units respectively.

Table 1: Activity Work Amount In Each Repetitive Unit

	Activity						
Unit	**A**	**B**	**C**	**D**	**E**	**F**	**G**
1	100	150	200	150	100	150	50
2	250	100	150	200	150	250	200
3	150	200	50	100	50	50	50
4	200	150	200	150	100	100	150

Table 2: Moments of Daily Crew Production Rates

Activity	Mean	SD
A	10	1.0
B	20	2.0
C	15	1.5
D	15	1.5
E	25	2.5
F	15	1.5
G	20	2.0

In each of the 4 repetitive units, crew production rates (in work amounts per day) for each of the 7 activities are assumed to follow normal distributions with the means and standard deviations shown in Table 2. Consequently, the duration of each activity in each of the 4 repetitive units varies because of differences in work amounts from unit to unit and because of random production rates.

ASSIGNING CREW LEAD TIMES

The schedule resulting from the sequence step algorithm after the first simulation replication for sequence step 1 is shown in Figure 4. “SQS1” in the title indicates the first sequence step, while “n1” indicates the first replication. The resulting project schedule at the end of the first replication is shown as a production diagram.

At this point, the crew lead times (CLT) for all activities have been set to 0. Thus, the first data point for the crew idle time, CIT, for each activity (measured from the beginning of the project) is given by the ESD of the activity in the first unit plus the sum of all its lags between units. For example, the first data point for the B crew idle time, CIT_B, is given by the sum $ESD_{B1} + Lag_{B1,B2} + Lag_{B2,B3} + Lag_{B3,B4}$.

For this example, 10,000 replications were simulated within each sequence step. Thus, 10,000 data points for the crew idle times of activities B and C were collected by the end of processing sequence step 1. These 10,000 crew idle times for C and B each were then arranged by relative frequency in cumulative bins using an interval of 5 time units as shown in Table 3. Each row of this table shows a time value and the percent of the 10,000 crew idle times for activity B that were less than that value. Thus, if activity B is scheduled to start on day 55 there is 94.27% probability that its continuity in resource utilization will be maintained.

In this example, an 80% confidence level is used to select crew lead times (CLT) for all activities. Hence, the crew lead time for B is set at $CLT_B = 55$ days (the first value in Table 3 that exceeds 80%). Similarly, at 80% confidence, $CLT_C = 50$ days.

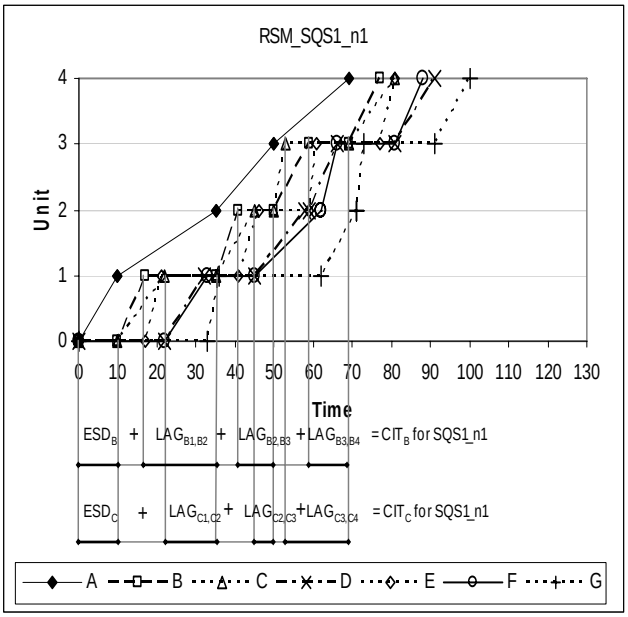

Figure 4: Measurement of Crew Idle Time (Production Diagram, 1st Replication, SQS1)

Table 3: B Crew Idle Times from SQS1

CIT_B	
Range	**% Frequency**
< 35	0.01
< 40	1.26
< 45	19.37
< 50	65.92
< 55	**94.27**
< 60	99.40
< 65	99.95
< 70	99.99
< 75	100.00

Once the crew lead times for B and C are set to 55 and 50 days respectively, the algorithm moves to sequence step 2. The assigned crew lead times CLT_B = 55 days and CLT_C = 50 days will remain constant until the algorithm finishes. The assigned crew lead times for all activities appear in Table 4.

Table 4: Lags Between Units, Crew Lead Times, and Average Project Duration

	Sum of Lags Between Units							Assigned Crew Lead Time (CLT)							Average Project Dur	Average Total Idle Time
SQS	A	B	C	D	E	F	G	A	B	C	D	E	F	G		
1	0	38	34	30	48	30	45	0	0	0	0	0	0	0	102	225
2	0	0	0	1	11	1	16	0	55	50	0	0	0	0	113	29
3	0	0	0	0	0	0	12	0	55	50	70	80	70	0	119	12
4	0	0	0	0	0	0	0	0	55	50	70	80	70	100	123	0

Figure 5 shows the final schedule where crew lead times for all activities have been assigned. As can be seen, all activities start at their respective CLT. From Table 4 we see that the average total idle time for all crews has been reduced from 225 crew-days to 0, while the average project duration has grown from 102 to 123 days.

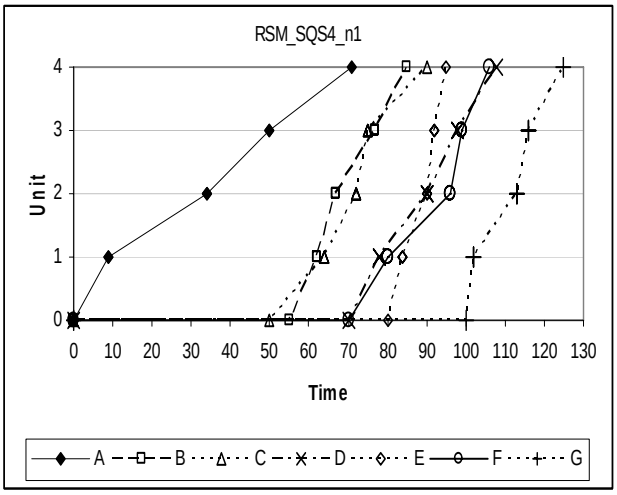

Figure 5: Production Diagram, 1st Replication, SQS4

DISCUSSION OF RESULTS– SELECTION OF CONFIDENCE LEVEL

Table 4 shows that as the sequence step algorithm progresses, the average project duration increases from 102 days to 123 days. The corresponding cumulative distributions for project duration for each sequence step in the algorithm are shown in Figure 6.

The greatest time shift between successive distributions occurs between sequence steps 1 and 2 which gives an average increase in project duration from 102 to 113 days. The reason can be observed in Figure 4. Activity A is the slowest activity in the project and as a result contributes the most to discontinuities in the work of other activities. Thus, to avoid discontinuities in its successor activities B and C, it is necessary to shift their crew arrival date significantly, which in turn increases project duration the most.

A key question in the optimal use of the sequence step algorithm is how to select an appropriate confidence level for the occurrence of crew work interruptions to balance the increase in project duration. This is an important issue because high confidence levels (to virtually eliminate idle time) can lead to significant increases in project duration.

To address this issue, Figure 7 shows 5 lines that relate average total crew idle time (total CIT in crew-days) and average project duration (in days). Each line corresponds to a different confidence level: 20%, 40%, 60%, 80%, and 100%. Moreover, each line consists of four points that correspond, from left to the right (or top to bottom), to the four SQS steps 1-4 of the algorithm. Thus, the expected total idle time between units decreases as the algorithm proceeds from one sequence step to the next. Clearly, selecting a greater confidence level decreases the expected crew idle time but also increases the expected project duration. Thus, allowing some interruption can benefit project duration.

It is very important to notice that irrespective of confidence level, the average total idle times at the completion of the algorithm (bottom point in each line) are *very small*. In particular, the expected idle time for confidence levels of 60%, 80%, and 100% is practically zero. Yet, the expected project durations for confidence levels of 60%, 80%, and 100% increase from 118 to 123 to 158 days. It is not hard to conclude that the optimal confidence level is between 60% and 80%, but definitely not 100%.

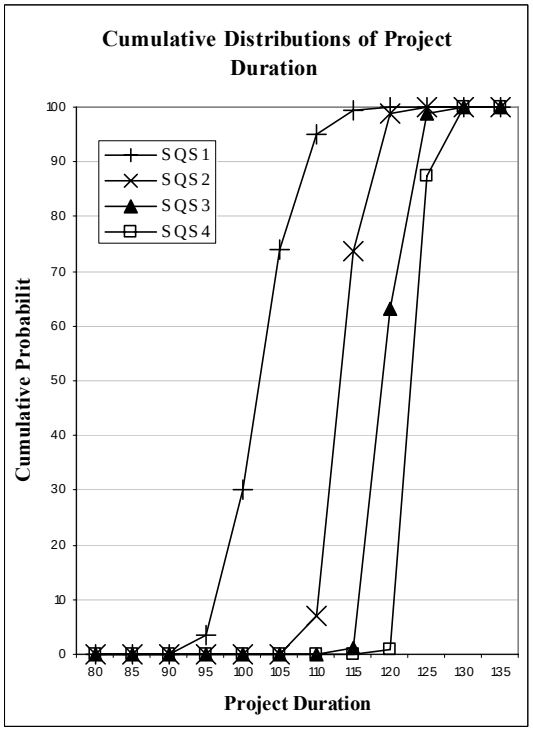

Figure 6: CDFs of Project Duration at 80% Confidence Level of Continuous Resource Utilization

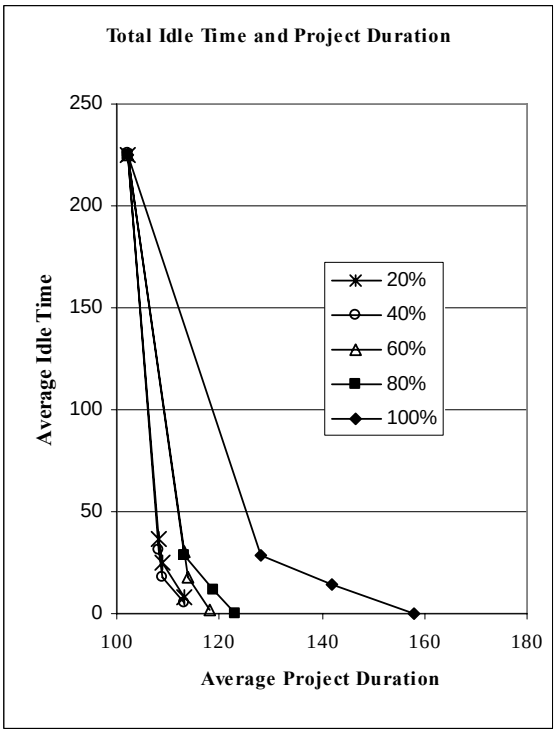

Figure 7: Average Idle Time vs. Average Project duration for Different Confidence Levels

Figure 8 shows the corresponding probability density functions for project duration for different confidence levels. Here it is again evident that there is little difference between confidence levels of 60% and 80% but there is substantial difference between 80% and 100%. Figure 8 also shows the probability density functions for project duration when the project is scheduled using CPM and RSM.

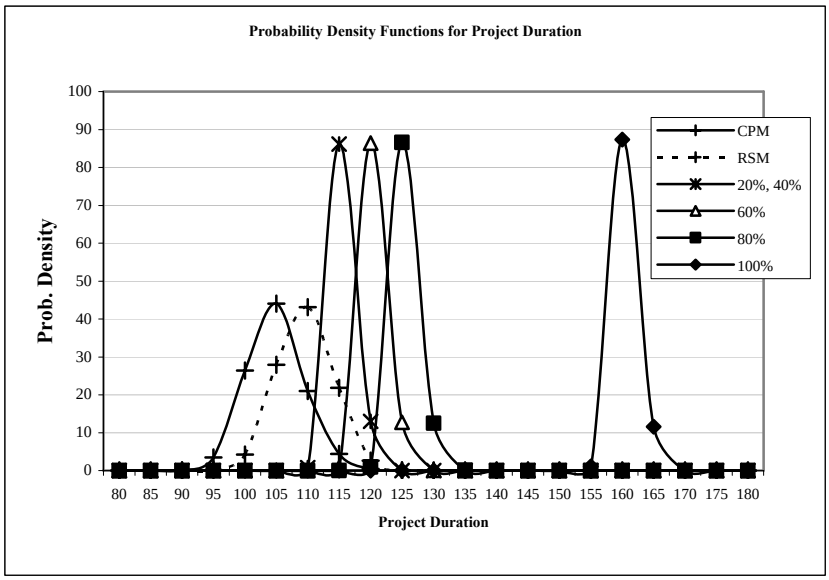

Figure 8: Project Duration PDFs for Different Confidence Levels of Continuous Resource Utilization

For the CPM case, activities in each replication are allowed to start as early as their predecessors allow. Thus, project duration tends to be the shortest but has large variability. For the RSM case, activities in each replication are scheduled *with perfect hindsight* so as to eliminate crew idle time. This produces a slight increase in expected project duration over the CPM case but completely eliminates idle time for that replication.

The difference between the RSM expected project duration and that for (say) an 80% confidence level represents ***the value of perfect information*** about the true activity durations that will be experienced during construction. If these were known ahead of time, then the project could be scheduled with even shorter crew lead times and have an even shorter project duration. However, that is not the case in real life, and hence the need for this algorithm.

CONCLUSION

For projects with *deterministic* activity durations, the repetitive scheduling method (RSM) delays the start of activities deliberately in order to achieve resource continuity. Doing so reduces activity floats, increases the number of critical activities and may even increase project duration. The problem can simply be stated as "how to minimize project duration subject to resource continuity constraints".

In an uncertain environment, it may be beneficial to delay the start of activities even beyond the levels recommended by deterministic RSM (i.e., to introduce buffers) to account for randomness in production rates and work quantities in preceding activities. The magnitude of such buffers and their placement are not easy to quantify. The proposed algorithm provides a direct solution to this problem by treating uncertainty explicitly.

The sequence step algorithm is the first to address the problem of scheduling *probabilistic* repetitive projects to eliminate crew idle time. The algorithm can be adapted easily to different resource-based simulation software by adding two nested loops: an inner replication loop and an outer sequence-step loop. Its application in Stroboscope has proven very effective and can help tackle a difficult problem that had hitherto eluded formal treatment. The problem of continuous resource utilization is central to many types of simulation models (and not just repetitive projects) that can now be analyzed and given the necessary attention.

REFERENCES

Harris, R.B. (1978). "*Precedence and Arrow Networking Techniques for Construction.*" John Wiley & Sons, Inc., New York, ISBN 0-471-04123-8, 50-51.

Harris, R.B., and P. G. Ioannou. (1998). "Scheduling projects with repeating activities." ASCE, *J. of Constr. Engrg. and Mgmt*., ASCE, 269-278, July/August Issue.

Ioannou, P.G., and V. Likhitruangsilp. (2005). "Simulation of multiple-drift tunnel construction with limited resources", In *Proc. of the 2005 Winter Simulation Conference*, 1483-1491. IEEE, Piscataway, New Jersey.

Ioannou P.G. and C. Srisuwanrat. (2006). "Sequence Step Algorithm for Continuous Resource Utilization in Probabilistic Repetitive Projects", In *Proc. of the 2005 Winter Simulation Conference*, IEEE, Piscataway, New Jersey.

Martinez, J.C. and P.G. Ioannou. (1999). "General Purpose Systems For Effective Construction Simulation." ASCE, *J. of Constr. Engrg. and Mgmt*., ASCE, (125)4, July/August Issue.