Toward a Teaching Software Application for Crashing the Schedule: SPE^{TM} Beta v.1

B. Gokhan Celik PhD, LEED-AP
Roger Williams University
Bristol, Rhode Island and Providence Plantations

S. Gokce Celik
Havelsan Hava Elektronik Sanayi ve Ticaret A.S.
Ankara, Turkey

Gilbert C. Brunnhoeffer, III PhD, P.E.
Roger Williams University
Bristol, Rhode Island and Providence Plantations

The calculation of the minimum duration of a schedule (crashing the schedule), and identifying various milestones (durations) along the way, requires identifying the best order for reducing durations of the individual tasks. This paper continues a discussion that was introduced in April 2010 by Brunnhoeffer & Celik at the Associated Schools of Construction Annual Conference, in Boston, MA. It is the goal of this paper to introduce Schedule Path Explorer Beta version 1.0 (SPE^{TM} Beta v.1) as an initial step toward developing a crashing software program to be used mainly by instructors and students but also by professionals to improve the understanding of how to crash a schedule. The SPE that is introduced in this paper is a response to the issue of automatically identifying all the paths on a given network problem. Identifying the activities on each of the paths makes it possible for students to build spreadsheets and track the changes in critical or noncritical status, as the duration of certain activities are reduced based on their associated costs. This paper also introduces the future development concepts for the SPE program toward a complete teaching software program, which can be utilized in the scheduling and project management courses offered in higher education institutions.

Key Words: SPE Beta v.1, Schedule crashing, Paths, Minimum cost, Minimum time

Introduction

Project scheduling involves the development of a plan which specifies for each activity the logic connections, resources, and feasible start and completion dates, the amounts of the various resource types that will be needed during each time period and as a result the budget (Demeulemeester & Herroelen, 2002). Creating a schedule for a construction project rarely ends with the calculation of the critical path(s), an overall project duration, and a budget. Many times, schedulers, field supervisors, and project managers will evaluate the schedule for potential cost and time saving alternatives. One of these methods used to make the project duration shorter while increasing direct costs, and decreasing indirect costs, is called “crashing a schedule”. Brunnhoeffer & Celik (2010) introduced an algorithmic approach to crash a schedule by using a series of spreadsheets generated in Microsoft Excel (MS Excel). This process is explained as:

> “…a process of incrementally reducing the duration of selected tasks located on the critical path and, simultaneously, incrementally increasing the direct costs of the task being reduced. The new total time to complete value and direct costs to complete value are then calculated. The overhead cost value applied to the project is calculated in light of the reduced project schedule duration and added to the new direct cost value to obtain the new total project cost value. The data points representing the collection of incremental changes in the project duration and the associated project costs are plotted and the result of the data points usually shows a declining total cost (direct + indirect cost) until the incremental direct cost overwhelms the incremental decrease in indirect costs.” (Brunnhoeffer & Celik , 2010).

The first step of the crashing process, for complex projects, is to identify all the paths in a given construction schedule. This process helps the students get more familiar with the schedule and improve their understanding of the “critical path” and its impact on the overall project duration (Fig. 1).

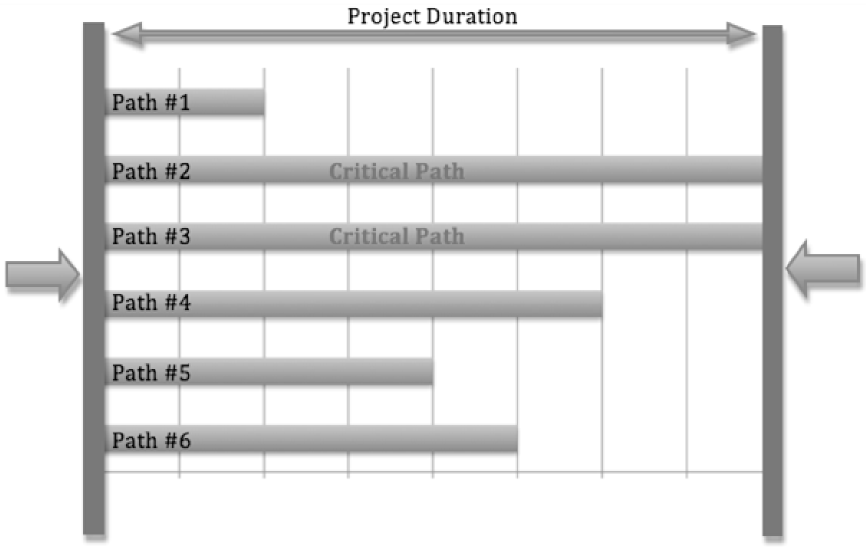


Figure 1: Crashing process in relation to all existing paths

The authors of this paper have observed that this can be a challenging task for students under certain circumstances. The difficulty of this task increases significantly when the number of activities and the relationships increases. There are multiple methods that students can use to identify the number of paths, depending on their level of familiarity with construction planning and scheduling. These methods can include a manual approach by the help of a spreadsheet application such as MS Excel, or by visually inspecting a network diagram. The latter can be achieved by drawing a precedence diagram or by using scheduling/project management (PM) software programs such as MS Project, or Primavera P6.

Identifying Paths in Microsoft Excel

Identifying the paths of a given network manually over a spreadsheet application was described in the authors' previous study and as shown in Table 1 and Table 2 (Brunnhoeffer & Celik , 2010)

Table 1

Sample layout for CPM network data (Brunnhoeffer & Celik, 2010)

	A	**B**	**C**	**D**	**E**	**F**	**G**
1	Task #	Task	**IPA** **I**mmediately **P**receding **A**ctivities	Normal Duration	Normal Direct Costs	Crash Duration	Crash Direct Costs
2	1	A	**-**	1 day	$ 6,000	1 day	$ 6,000
3	2	B	A	6.5 days	$13,500	5 days	$15,500
4	3	C	A,B	8 days	$12,350	6.5 days	$18,000
5	4	D	A,B	7 days	$22,145	5 days	$26,300
6	5	E	B,C	3 days	$ 2,675	1.5 days	$ 3,500
7	6	F	B,D	8.5 days	$34,000	5.5 days	$36,650
8	7	G	C,E	4 days	$ 1,250	2.5 days	$ 2,150
9	8	H	D,F	3 days	$ 5,005	1 days	$ 8,705
10	9	K	D,G,H	2 days	$ 2,500	2 days	$ 2,500

The first step to identify all the paths manually requires to start with the first task, consult the Immediately Preceding Activity (IPA) column to locate the first task, and list that task once for every time it is found on the IPA column. Students can to continue in this fashion until all tasks are accounted for and every path begins with the first task, Task A, and ends with the last task, Task K. The result of this process is a list of all available paths that exist in the given network problem (Table 2).

This process, although very accurate, can take a significant amount of time and has a high risk of mistakes due to intense manual repetition. Students spend significant amounts of time during exercises and exams to find all the paths and many times make mistakes that may alter their decision on which activities to crash and in which order to crash them. Details of this method and the additional steps to complete a crashing exercise can be found in Brunnhoeffer & Celik, (2010).

Another method to identify all the paths is to analyze the precedence diagram that can be generated manually (as an additional exercise for lower level courses) or with the help of a project management software program for the higher-level courses. Advantages and disadvantages of this method, from an educator perspective, are discussed in more detail in the following section.

Table 2

Setting up the path determination in on Excel (Brunnhoeffer & Celik , 2010)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
16	Path Duration	Path #	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
17		1	A	B	C	E	G	K
18		2	A	B	C	G	K	
19		3	A	B	D	F	H	K
20		4	A	B	D	H	K	
21		5	A	B	D	K		
22		6	A	B	E	G	K	
23		7	A	B	F	H	K	
24		8	A	C	E	G	K	
25		9	A	C	G	K		
26		10	A	D	F	H	K	
27		11	A	D	H	K		
28		12	A	D	K			

Identifying Paths in a Precedence Diagram

Drawing and analyzing a precedence diagram can also help identify the paths of a scheduling problem. Although this will reveal the entire network diagram thus all possible paths from the Start to the Finish activity, this method will still require students to visually identify each path. This visual and manual interference can lead to mistakes and omissions especially when the number of activities increases in the schedule (Fig. 2). Currently there are no software applications that list, in writing, each available path in a schedule, for the users to analyze. Available software programs highlight only the critical path, while they consider the feature of listing all paths pointless for practical applications. This assumption can be valid for the professional use of these computer applications, but for educational purposes, it presents a gap for students who can develop a better understanding of CPM scheduling by observing all paths available on a project schedule.

One of the advantages of using a software program such as MS Project is that it allows users to create a precedence diagram quickly and accurately. Although the use of similar applications is widely accepted and practiced in the industry, limited use of scheduling applications in the lower level courses can improve students' understanding of the basics of crashing a schedule.

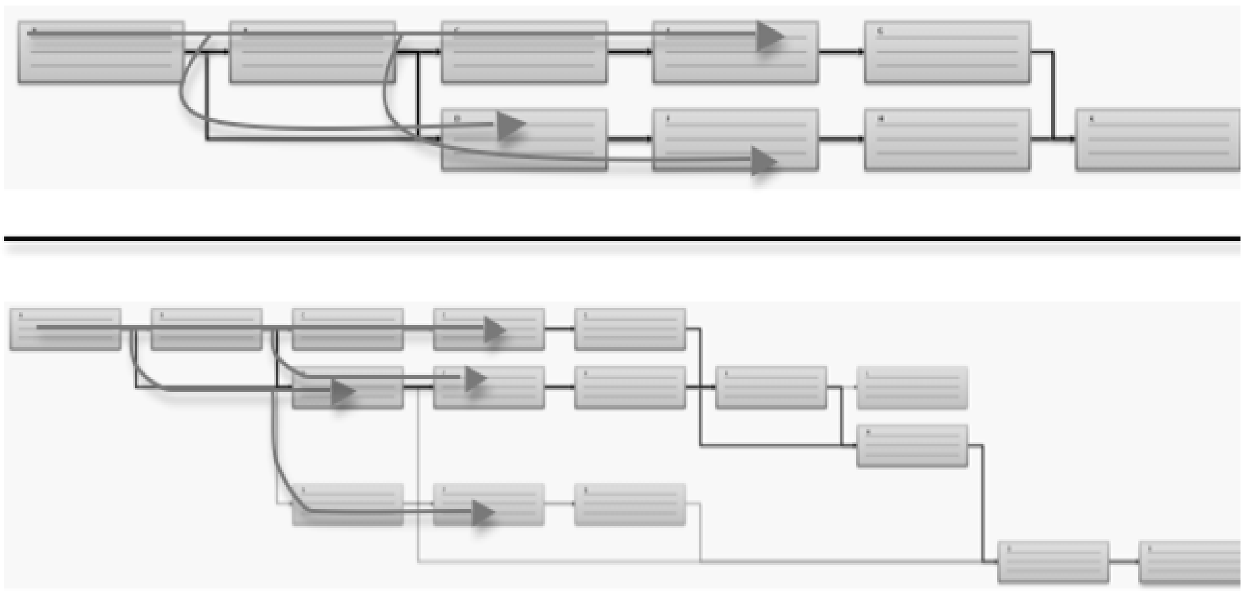
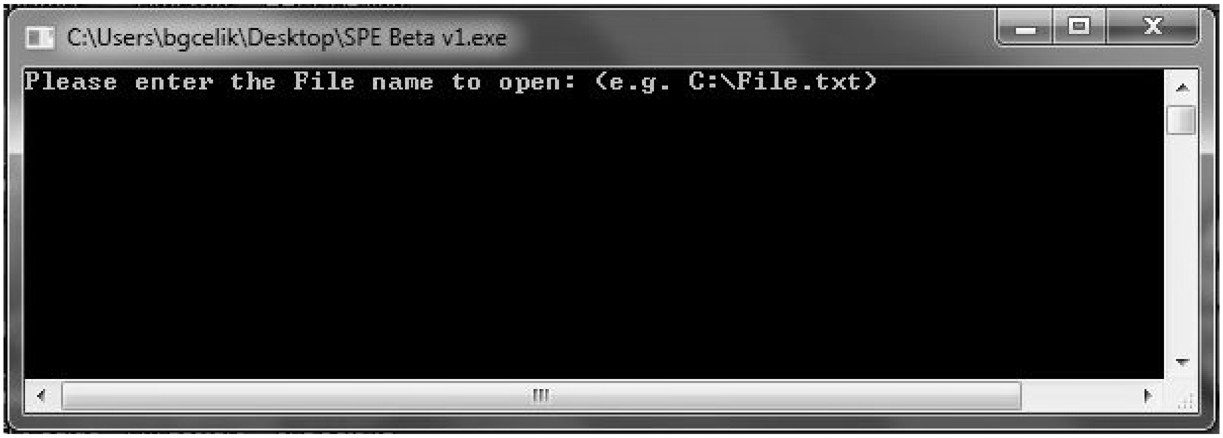


Figure 2: Complexity of visually identifying paths while the number of activities increase

Identifying Paths with SPE^{TM} Beta v.1

This paper introduces a simple computer application that would allow its users to identify all the paths of a network schedule. As described in the previous sections of this paper, this allows instructors, as well as the students to analyze all the paths of a project's schedule, from the beginning of the crashing process. SPE Beta v.1 was developed via Microsoft Visual Studio, utilizing C++ programming language. The programming of the application allows it to be used in various operating platforms such as Windows and Linux.

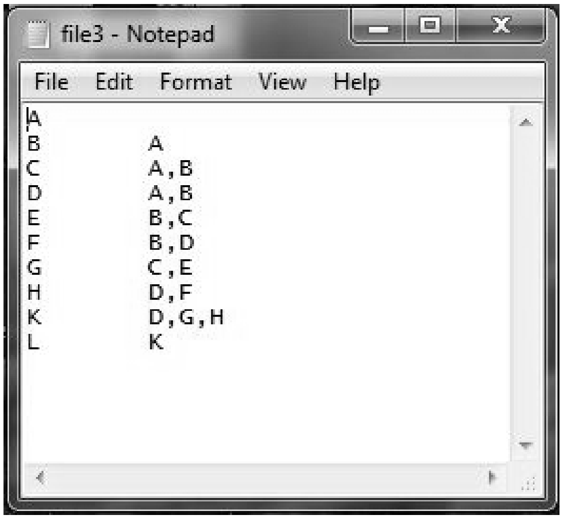
SPE Beta v.1 requires the user to choose a text file that includes all the activities on the network and their IPA information. Once the correct file is chosen, SPE Beta v.1 displays all the paths in a similar format to the results developed by the manual MS Excel method. Figure 3 illustrates the initial appearance of SPE Beta v.1 prior to displaying the paths.



```
Please enter the File name to open: (e.g. C:\File.txt)
```

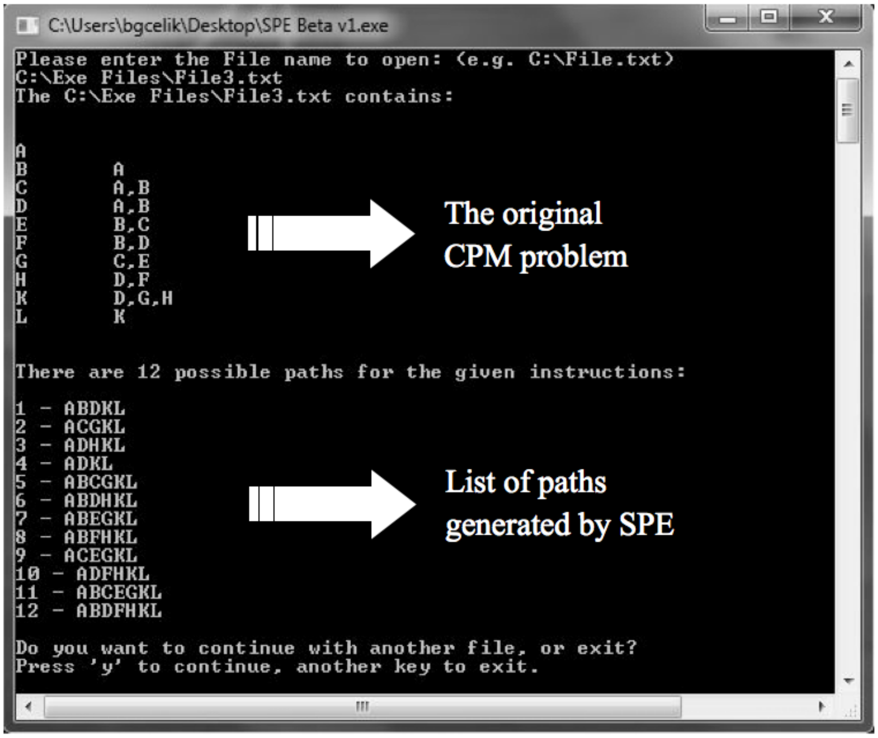
Figure 3: Opening/initial screen of SPE Beta v.1

Current configuration of SPE Beta v.1 requires the text file to include the name of the activity and its predecessors, which should be separated by a tab. Once the location of the text file is entered, application first lists the activities and the predecessors, and then proceeds with listing all available paths. Figure 4 illustrates a sample text file (.txt) for the same network problem given in Table 1. In the solution generated by SPE Beta v.1, there are a total of twelve (12) paths as identified manually by MS Excel (See Table 2). Application accomplishes this by going through a set of iterative algorithms in a way to complete each path when the paths reach the finish activity, K. Once the solution to the chosen text file is generated, SPE Beta v.1 allows user to continue by uploading another text file or exit the application (Figure 5).



```
A
B       A
C       A,B
D       A,B
E       B,C
F       B,D
G       C,E
H       D,F
K       D,G,H
L       K
```

Figure 4: Sample text file that includes the project information to identify all possible paths

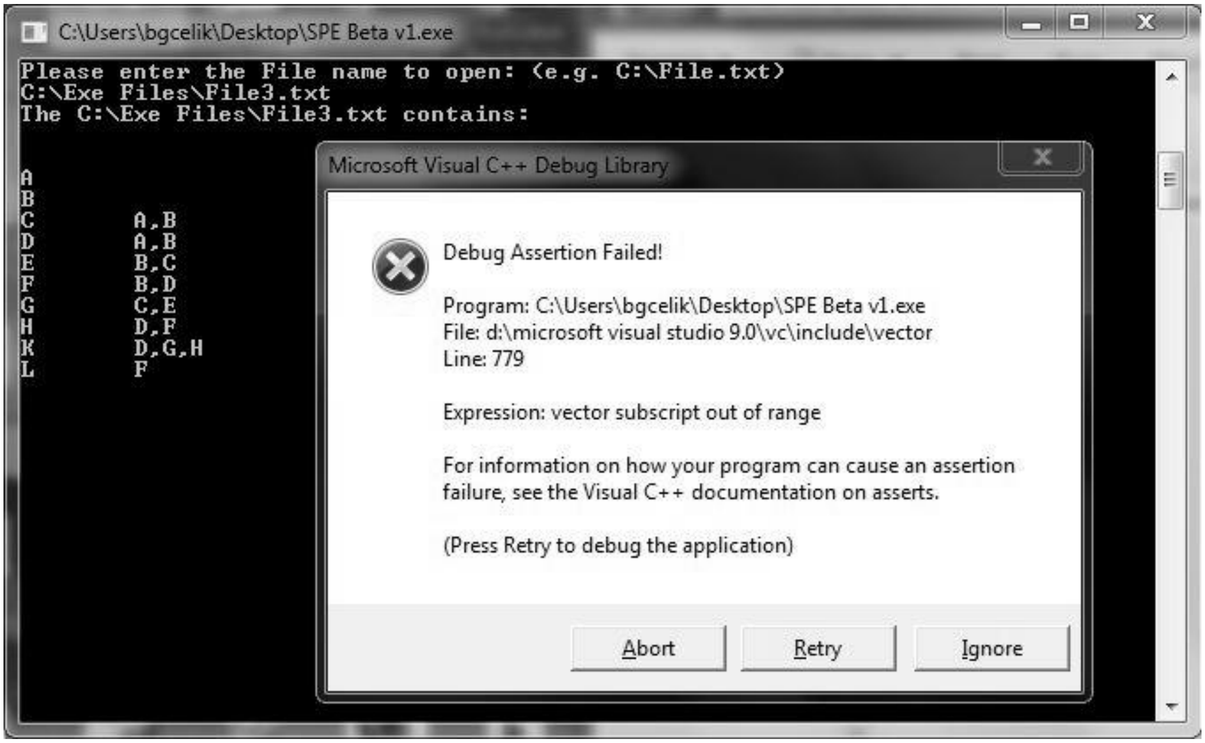


```
Please enter the File name to open: (e.g. C:\File.txt)
C:\Exe Files\File3.txt
The C:\Exe Files\File3.txt contains:


A
B       A
C       A,B
D       A,B
E       B,C
F       B,D
G       C,E
H       D,F
K       D,G,H
L       K

There are 12 possible paths for the given instructions:

1 - ABDKL
2 - ACGKL
3 - ADHKL
4 - ADKL
5 - ABCGKL
6 - ABDHKL
7 - ABEGKL
8 - ABFHKL
9 - ACEGKL
10 - ADFHKL
11 - ABCEGKL
12 - ABDFHKL

Do you want to continue with another file, or exit?
Press 'y' to continue, another key to exit.
```

Figure 5: The results screen of SPE Beta v. 1

Significance, Limitations, and Future Development

The significance of introducing SPE Beta v.1 derives from the fact that this application would allow instructors and students to analyze the schedule of a project prior to starting a crashing exercise. This application allows users to enter as many activities as possible, without an upper limit, and receive a list of all possible paths. Achieving these results by the help of manual logic methods can take a long time, and visualizing paths displayed on a project management computer application can be susceptible to mistakes (See Figure 2). The authors are planning to develop SPE Beta v.1 toward a more user friendly Graphic User Interface (GUI) which will allow the user to enters the activities and their predecessors directly into the software program. Currently the SPE application is nonresponsive to the situation where the users enter more than one activity with no predecessors (Start activity) or with no successors (Finish activity). This causes the algorithm to fail since the coding requires the network to have single "Start" and "Finish" activities (Fig. 6). This limitation will be resolved by creating a notification system within the newly developed GUI so the users can be directed to reenter/correct the activity relations to avoid multiple activities with no predecessor or successor.


C:\Users\bgcelik\Desktop\SPE Beta v1.exe
Microsoft Visual C++ Debug Library
Abort
Retry
Ignore

```
Please enter the File name to open: (e.g. C:\File.txt)
C:\Exe Files\File3.txt
The C:\Exe Files\File3.txt contains:

A
B
C       A,B
D       A,B
E       B,C
F       B,D
G       C,E
H       D,F
K       D,G,H
L       F
```

Debug Assertion Failed!

Program: C:\Users\bgcelik\Desktop\SPE Beta v1.exe
File: d:\microsoft visual studio 9.0\vc\include\vector
Line: 779

Expression: vector subscript out of range

For information on how your program can cause an assertion failure, see the Visual C++ documentation on asserts.

(Press Retry to debug the application)

Figure 6: Limitation of SPE Beta v.1 when two or more activities with no predecessors or successors, are introduced in the text file.

An additional step in the same direction is to advance SPE into a Schedule Path and Crashing Explorer (SPCE^{TM}) that would allow students to crash a schedule in multiple steps. It is the goal of the authors to keep the solutions generated by these applications, as raw and detailed as possible. This will make it possible for the instructors to use the SPE or SPCE applications also as solution generators for scheduling, and crashing questions. Similar solutions can be generated with also the help of simple linear programming (LP) models (Lima et. al., 2006). Development of an LP model by construction management (CM) students, in order to crash a project network, can be achieved by incorporating basic operations research topics into traditional CM programs.

The goal of the development efforts and the currently introduced SPE application is to allow students to work faster with fewer mistakes while still understanding the basics of the process for crashing a project's schedule.

Conclusion

Crashing a project can be accomplished with the help of various manual and computerized methods. Computerized methods, while faster, are not as educational as the manual ones (with the help of MS Excel) to help students understand the process and the logic behind shortening a project's duration. The goal of crashing can aim (1) to determine the minimum duration of a project (when cost is less of a concern) or (2) to determine the optimum duration of a project for the lowest possible cost and its associated duration. An effective way of teaching these procedures to students is by the help of manual MS Excel algorithmic solutions where the students crash the activities only on the critical paths, one by one to achieve various cost and duration combinations. One of the difficulties in achieving a complete manual solution is to determine all the paths in a CPM schedule. This phase of the solution takes the most amount of time and is the cause of the majority of mistakes. This paper presented a new application, SPE Beta v.1, developed with C++ language, as a second phase of a study initially introduced by the authors in April 2010 at the ASC Annual International Conference.

SPE can be used by professionals but is developed mostly for educators and students primarily in the CM programs of higher education institutions. This application can provide students the opportunity to move faster toward the actual crashing process while allowing them to understand the basic concepts in regards to different paths and their relationship to the final project duration. Applications like SPE can be used by instructors to break up their computerized approach to construction scheduling and crashing, into various segments for different levels of courses. SPE makes it possible to utilize computer technology without the necessity of utilizing a complete PM computer application. This can create a more transparent learning process for students who need to understand the process of crashing a schedule, as much as the final results that achieve the optimum or minimum durations.

References

Brunnhoeffer & Celik (2010), "Crashing the Schedule – An Algorithmic Approach with Caveats and Comments". Proceedings of Associated Schools of Construction 46th Annual International Conference, April 7-10, 2010, Boston, MA.

Demeulemeester, E. L., & Herroelen, W. (2002). *Project Scheduling: A Research Handbook.* New York, NY, USA: Kluwer Academic Publishers.

Lima, M., Silva, L., Vieira, R. (2006) "Project Crashing and Costs Laws in the Knowledge Age". Third International Conference on Production Research. America's Region 2006.