Combining forward and backward process simulation for generating and analysing construction schedules

Gergő Dori, André Borrmann
Technische Universität München, Munich, Germany

Kamil Szczesny, Matthias Hamm, Markus König
Ruhr-Universität Bochum, Bochum, Germany

Abstract

In the construction industry there is an essential requirement for efficient construction schedules that consider the availability of resources as well as constraints. It is standard practice in today's construction process planning to use the Precedence Diagram Method (PDM), including well-established methods such as the Critical Path Method (CPM). A possible means of improving the mostly manually executed generation process is to apply computer-aided methods such as discrete-event simulation. By combining this method with resource-constrained scheduling methodology it becomes possible to not only verify schedules but also to generate schedules that take resources into consideration. In addition to discrete-event simulation CPM is able to analyse float times, but is not able to consider resources. Discrete-event simulation on the other hand can consider resources, but is unable to calculate float times. The calculation of float times for construction tasks is an important task. An awareness of float times makes it possible to identify whether a delayed task can be compensated for or if the entire construction project will be late.

In this paper we propose using Backward Simulation methodology to develop an elaborated approach that extends discrete-event simulation by the ability to calculate float times. Here, the simulation starts at the virtual completion date of the construction project and runs backwards in time until the start date. To determine float times it is important that the task execution sequence is in the same order for forward and backward simulation. Consequently we present an extension to the simulation concept that controls the execution order of the tasks within the backward simulation. By combining the results of the forward and backward simulation, it is possible to determine float times for each task while taking into account resources. A comprehensive case study is described that illustrates the application of this new approach. One result of this is the determination of detailed float times for each task using discrete-event simulation.

Keywords: process simulation, construction scheduling, backward simulation, float times.

1 Introduction and description of the problem

The detailed scheduling of construction projects is more important today than ever before. The goal is to schedule the multitude of construction tasks efficiently with regard to different objectives. The float time of a task describes the time frame for the execution of the task: within this limited frame, the execution of the task can be moved or the duration of the task can be extended without any impact on the total duration or the execution of subsequent tasks.

In this paper we present a concept that makes it possible to calculate float times for all tasks in complex construction projects. The calculation of float times for single tasks is, however, a challenging task. There are various types of constraints that have to be considered in the planning process of construction schedules, e.g. resource restrictions, material availabilities or technological dependencies between construction tasks. Because the planning of construction schedules is mostly undertaken manually by a planner, the quality of the schedule depends of the planners' experience. Where material and resource restrictions do not need to be considered, i.e. only precedence relationships of tasks are crucial; the schedule can be analysed by applying Critical Path Method (CPM) analysis and represented by an activity-on-node diagram. For these kinds of scheduling problems, where material or resource constraints do not factor, the calculation of float times for tasks is a straightforward procedure. Starting with the first task, the execution time will be added to the start time. The result is the earliest end time of the first task as well as the start time of the next tasks. After finishing this forward calculation, the earliest possible start time is calculated for each task. Subsequently, a backward calculation is conducted, which works in a similar manner. Starting with the last tasks, i.e. those that have no successors, the latest start times of all the tasks are calculated. The difference between the latest and the earliest start times defines the total float time of a task. The tasks without float times constitute the critical path. A delay in any of these tasks will result in an increase of the project's overall time span and therefore in a delay of the project itself. For this reason it is important to determine float times for each task, so that it is possible to identify tasks that are part of the critical path.

In the case of construction projects, however, it is not sufficient to take only precedence constraints into account. Instead, additional restrictions such as resource and material constraints have to be considered. Using the CPM it is not possible to consider anything other than precedence constraints and as a result CPM cannot help in calculating float times for complex construction projects. Previous research investigated the generation of valid schedules for construction scheduling problems by applying constraint-based discrete-event simulation (Beißert et al. 2007; König et al. 2007). This approach is able to generate construction schedules that consider not only precedence relationships but also resource and material constraints as well. Using this simulation technique, the earliest start time for each task can be determined. In this paper a concept is introduced that extends constraint-based discrete-event simulation by the ability to determine float times for construction tasks. This approach combines the forward and backward simulation with additional methods that apply restrictions in order to obtain an identical order of tasks in the backward simulation as in the forward simulation.

The following section reviews approaches in the available literature. The latest research progress on the topic of float calculation by CPM and the methodology of constraint-based discrete-event simulation is also presented briefly. Section 3 describes the concept of calculating float times using a combined forward and backward simulation in detail. A comprehensive case study is presented in section 4, while the final section 5 contains concluding remarks and an outlook on future research objectives.

2 Related work

The standard approach used in today's construction-process planning is the Precedence Diagram Method (PDM), including well-established methods such as CPM. Raz and Marshall (1996) point out that applying CPM to float determination can lead to misleading interpretations of float times. This is the case when tasks rely on resources that only are available in limited quantities. As a result, some tasks may be scheduled later than their late dates because the required resources were not available earlier. This conflicts with scheduled early dates and late dates respectively, and the float times are no longer applicable. To solve this problem, Raz and Marshall propose new float definitions that retain the original meaning of float. The solution is based on a refinement of total float and free float. Using their approach it is possible to obtain a more precise and accurate measure of schedule flexibility and risk. Lu and Lam (2008) state that CPM is not able to handle multiple resource calendars for the total float determination. They propose a new method for eliminating this drawback based on forward pass analysis alone to accurately evaluate activity total float subject to calendar resource constraints. Moreover, Hegazy and Menesi (2010) point out that the CPM algorithm has no mechanism for considering multiple resource constraints, such as deadline and limited resources. It is therefore often difficult to generate schedules that are feasible for all constraints, i.e. one solution in response to one constraint may conflict with another solution for another constraint. Hegazy and Menesi's proposed solution is a mechanism that is based on a finer level of granularity. This is achieved by breaking down the task duration into separate time segments. As described above researchers have proposed improvements to CPM, for example to obey more and correct information about float times. In addition to the improvements to CPM, researchers have proposed alternative approaches (Hegazy and Menesi 2010). One promising and accepted approach is the use of discrete-event simulation models. Past research has focused on applying and improving such models and first efforts have been undertaken by Halpin (1977), Chang (1986) and Tommelein et al. (1994). Based on several different modelling concepts, processes, sequences and resources are described and simulated using sophisticated simulation tools. An innovative approach was also presented by Beißert et al. (2007) and König et al. (2007). Here, the constraint satisfaction approach was adapted for integration within discrete-event simulation. The scheduling problem is described in such a way that construction tasks are modelled as variables. Additionally, every restriction of a task, such as precedence relationship, material or resource constraints, is modelled as a constraint of the variables. The discrete-event simulation is able to find a value combination that fulfil the constraints for all variables. In Lu (2003) the methodology of discrete-event simulation is simplified for use in construction scheduling. As a result of this simplification, the use of construction simulation systems became an easily controllable tool. In Lu et al. (2008) a solution is presented for resource-constrained CPM. The proposed A simplified simulation-based scheduling system enables the automatic formulation of resource-constraint scheduling with the shortest total project duration. The lack of a flexible constraint satisfaction modelling approach and the clear separation between simulation and optimization are, however, two reasons why the constraint-based discrete-event simulation approach is favoured in this paper.

3 Concept

In order to calculate float times for construction-site schedules, a newly developed backward simulation method along with a coupling process extending the common forward discrete-event simulation is introduced.

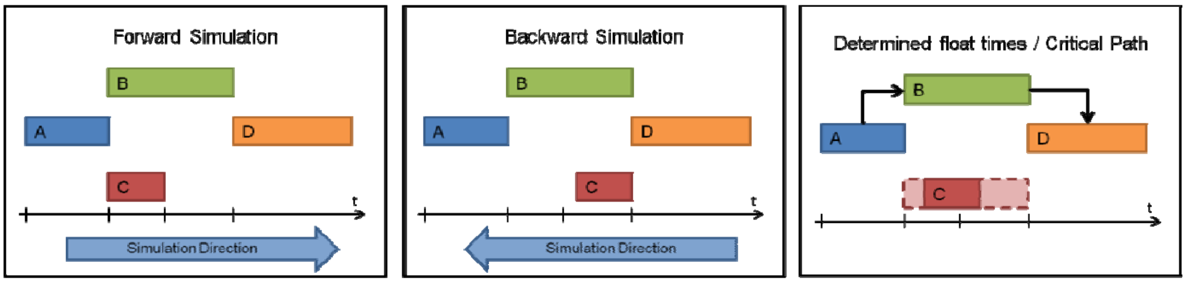


Figure 1: Concept of determining float time and critical path combining forward (left) and backward simulation (middle).

A schematic principle of float time calculation is presented in Figure 1. Four tasks are to be simulated: A, B, C and D. B and C must follow A and C must follow B and C. The execution time for each task is different. When simulating these tasks using forward simulation, B and C will start right after A has finished, and D starts after B has finished (C finished earlier). If we simulate these same tasks backwards, B and C will performed right after D has been completed and A will performed after B has been completed (C finished earlier) due to the changed simulation direction. The two results can be combined with one another because the complete execution time and the order of the tasks are the same. Task C has a float time between the finish of task A and the start of task D. The critical path of the schedule is therefore A→B→D because they have zero float time. Our goal is to make the constraint-based discrete-event simulation able to calculate these float times for each individual task in a schedule. To implement this concept and obtain reliable results from the connected simulations, conceptual changes and restrictions have to be made in the existing simulation model.

3.1 Forward Simulation

The most prevalent approach to generating construction schedules is *Discrete-Event Simulation* (AbouRizk, 2010). The *Discrete-Event Simulation concept* was developed for mechanical engineering tasks but its advantages were quickly recognised and it was adapted for use in the civil engineering industry to simulate construction-site projects. Since the introduction of the first simulation language CYCLONE (Halpin, 1977), simulation models have been developed for typical construction systems providing engineers with a means to experiment with ways of achieving more productive, efficient and economical field operations (Lu, 2003). Conventionally, in discrete-event simulation the execution order of the processes are predefined. This leads to a rigid and inflexible simulation behaviour, which contrasts with the more dynamic behaviour observed in real construction projects. The C*onstraint-based Methodology* (Beißert et al., 2007) was introduced with the aim of introducing more dynamism into the selection process of task execution. In this approach the construction scheduling problem is modelled as a *Constraint Satisfaction Problem*. Here, the requirements of a task (precedence relationships, resource needs) are modelled as constraints. A list is created that contains all the executable tasks. The constraints of the tasks are checked and the tasks are then executed as soon as all their constraints are satisfied. The checking process stops when the list is empty and there are no more executable tasks available. The order of the tasks in this list has a big influence on the execution order of the tasks of the complete project. Therefore different steering methods have been introduced, e.g. random selection or ranking processes (Beißert et. al. 2008), to implement different simulation strategies for construction projects (e.g. as fast as possible, as cost-effective as possible).

In the simulation concept presented in this paper a list of tasks is imported (Dori et al., 2011) into DESMO-J, a discrete-event based simulation engine (Page and Kreutzer, 2005) that has been extended with the constraint-based methodology, resulting in the generation of a construction schedule of these tasks. To determine the execution order of the tasks a FirstIn-FirstOut (FIFO) list is used. A task is only put into this list when all its precedence constraints are fulfilled so that for the items in the FIFO list, only the resource constraints need to be checked.

The scheduling process begins by resource checking all tasks without predecessors; these are then transferred to the FIFO list at the beginning of the simulation. As soon as one of these tasks is completed its successors will be informed (because one of the precedence constraints has been fulfilled). If the successor has any other predecessors it will wait; if not it will be transferred to the FIFO list for scheduling as soon as its required resources are available. When multiple equivalent tasks can be started at the same point in the simulation, the selection process either carries out tasks in the order that they appear in the *FIFO list*, or according to a pre-defined *strategy*. Strategies are implemented by using *Soft-Constraints* (Beißert et al., 2008) which can define *Priorities* for certain executable tasks. Using priorities, the constraint checking process will always start with the highest priority task, followed by the next highest priority task and so on. If some or all the tasks have the same priority, the execution order will be determined by the order they appear in the FIFO list. By introducing this scheduling method, the earliest possible start date of a task can be calculated taking into consideration the availability of required resources.

3.2 Backward Simulation

The concept of backward simulation is basically the same as that of forward simulation – the simulation actually runs forward in time but with reversed execution conditions. Using this approach, the resulting schedule is calculated based on a simulation that starts from a virtual completion date for the construction project and runs backwards in time until the starting point of the construction process is reached. The reversed conditions mean an inversion of the technological dependencies, e.g. in forward simulation concrete is filled after formwork assembly; in backward simulation formwork assembly follows filling with concrete (Fig. 2). This entails reordering the priorities and (as future work) reversing the resource calendar. To realize this inversion a function is written which reverses the order of the precedence constraint at the beginning of the simulation, changing the successor into predecessor and predecessor into successor.

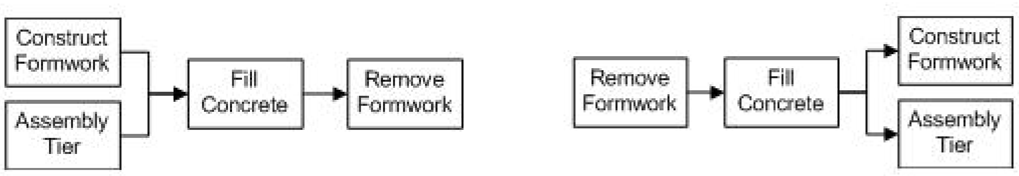
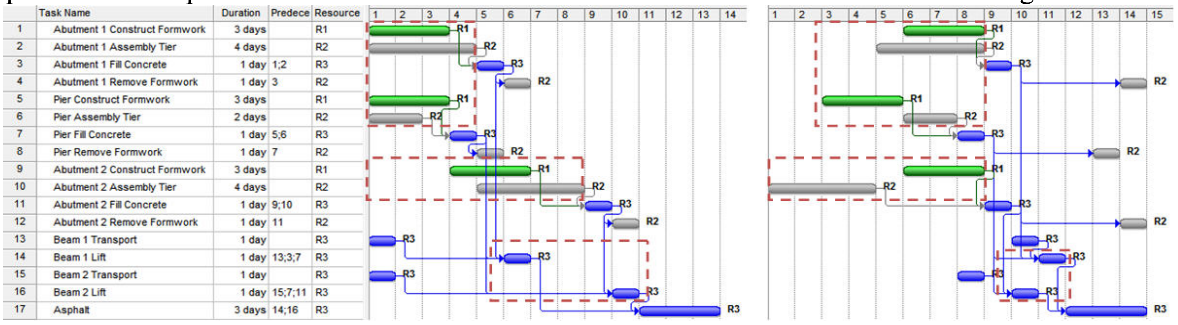


Fig. 2: Precedence relationships for forward simulation (left) and for backward simulation (right).

The first test of the backward simulation approach was realized using a simple bridge project with unlimited resources. This meant that the order of the executed tasks of the backward and forward simulation were identical and the difference between the start time of a task in the forward and the backward simulation is its float time. In this case the results are identical to those achieved using CPM.

By restricting the amount of available resources and letting the backward simulation run with the same configurations as the forward simulation causes an issue to occur. The order of the executed tasks is not identical to the result of the forward simulation. This occurs because the task execution process of the simulation is based on the order of the task list, but this is not the same in the case of forward and backward simulation. To illustrate the issue, an example is presented with two abutments, a pier and two superstructure beam elements. The different results can be observed in Fig. 3.



	Task Name	Duration	Predece	Resource
1	Abutment 1 Construct Formwork	3 days		R1
2	Abutment 1 Assembly Tier	4 days		R2
3	Abutment 1 Fill Concrete	1 day	1;2	R3
4	Abutment 1 Remove Formwork	1 day	3	R2
5	Pier Construct Formwork	3 days		R1
6	Pier Assembly Tier	2 days		R2
7	Pier Fill Concrete	1 day	5;6	R3
8	Pier Remove Formwork	1 day	7	R2
9	Abutment 2 Construct Formwork	3 days		R1
10	Abutment 2 Assembly Tier	4 days		R2
11	Abutment 2 Fill Concrete	1 day	9;10	R3
12	Abutment 2 Remove Formwork	1 day	11	R2
13	Beam 1 Transport	1 day		R3
14	Beam 1 Lift	1 day	13;3;7	R3
15	Beam 2 Transport	1 day		R3
16	Beam 2 Lift	1 day	15;7;11	R3
17	Asphalt	3 days	14;16	R3

Fig. 3: Different results of the forward (left) and backward simulation (right) without dependencies (colours: resources, arrows: precedence constraints, dashed boxes: changed order/earlier start).

In the results of the forward simulation, abutment 1 and the pier is started first with abutment 2 following later. In the backward simulation, abutment 2 is started first and only later the pier and abutment1. Due to this change in order, the lifting of the two beam elements is also swapped. Because of the changed order of the tasks, where some tasks start in the backward simulation even earlier than in the forward simulation (e.g. abutment 2 assembly tier), the results cannot be compared with each other and the float times cannot therefore be determined.

In order to create an identical order of executed tasks for the backward simulation and the forward simulation, the two simulations have to communicate with each other, which means that their simulation concept have to be modified.

3.3 Calculation of Float Times

In order to calculate float times with discrete-event simulation, the concept of the simulations needs to be modified. The goal is to obtain an identical order of tasks for the backwards simulation and the forward simulation. During the forward simulation, therefore, each of the executed tasks is assigned a priority according to the finish date of the task's execution (the later it is, the higher the priority). The backward simulation then uses these priorities to replace the FIFO list with a priority ordered list of executable tasks for every time interval, and then tries to collect the necessary resources for the one with the highest priority. If that is not possible the next task with the second highest priority is chosen and so on. Using this method the execution chain of the tasks will be identical to that of the forward simulation; the backward simulation follows the task order of the forward simulated schedule, so all the tasks start later or at the same time as they do in the forward simulation. This is an important confirmation step of the backward simulation. As mentioned before, the forward simulation calculates the earliest possible execution date of the tasks, so reversing the execution conditions of the simulation (e.g. the technological dependencies) will result in scheduling the tasks as late as possible in time. In order to calculate float times for each individual activity the results of the forward and backward simulation are exported to a database and compared with each other. If a task in the backward simulation starts later then in the forward simulation the task has a float time, i.e. the difference between the two results. If the results are the same, the task has no float time and is therefore part of the critical path. If the result of the forward simulation has no float time the backward simulation will result in the same schedule, so the start time of the task in the backward simulation will always be the same or later than in the forward simulation. It is important to note the effect of resource dependencies on the float times. If there are tasks that are not technologically dependent on each other but use the same resources and have float times, when an earlier task is delayed, other later subsequent tasks will have to move with it; or the float time of an earlier task will need to be made shorter than expected without resource consideration. An example for this dependency will be shown in the next section.

4 Case Study

To present the application of this new approach, we will examine it using a simple bridge construction project. The bridge is made of two abutments, a pier and two pre-cast beam elements. The simple construction tasks for the abutments and the pier are constructing the formwork and assembling the tiers, filling the concrete and removing the formwork (for simplicity's sake we have omitted the drying time for the concrete). The beam elements then have to be transported to the construction site before being lifted up into position, but only when the corresponding abutment and the pier have been filled with concrete (represented as precedence constraint). When the beams are in place the asphalt can be laid for the road. There are three categories of resources used:

• R1: Formwork-works:	7 person
• R2: Assembly-works:	6 person
• R3: Concreting-work and others:	6 person

The resources are calibrated in such a way that 2 tasks with the same resource requirements can always be executed in parallel. The diagram of the tasks used, and their resources are presented in

Fig. 4. The tasks are represented as boxes with their name, necessary resources and the duration of execution. The arrows represent the precedence constraints between the tasks.

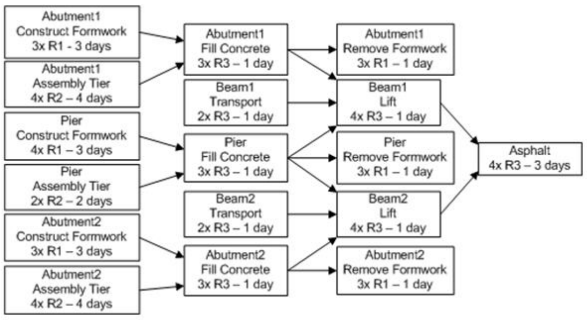
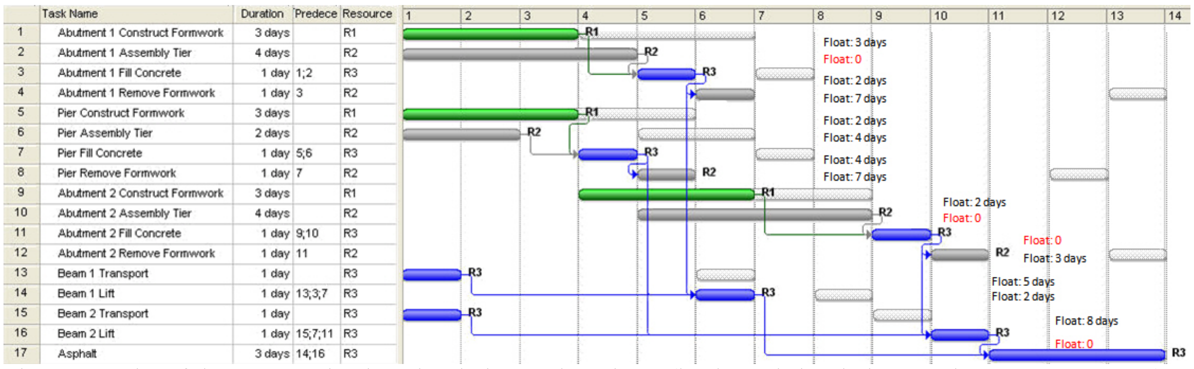

Abutment1 Construct Formwork 3x R1 - 3 days
Abutment1 Assembly Tier 4x R2 – 4 days
Pier Construct Formwork 4x R1 – 3 days
Pier Assembly Tier 2x R2 – 2 days
Abutment2 Construct Formwork 3x R1 – 3 days
Abutment2 Assembly Tier 4x R2 – 4 days
Abutment1 Fill Concrete 3x R3 – 1 day
Beam1 Transport 2x R3 – 1 day
Pier Fill Concrete 3x R3 – 1 day
Beam2 Transport 2x R3 – 1 day
Abutment2 Fill Concrete 3x R3 – 1 day
Abutment1 Remove Formwork 3x R1 – 1 day
Beam1 Lift 4x R3 – 1 day
Pier Remove Formwork 3x R1 – 1 day
Beam2 Lift 4x R3 – 1 day
Abutment2 Remove Formwork 3x R1 – 1 day
Asphalt 4x R3 – 3 days

Fig. 4: Task structure of the example project.

After exporting the results of the two simulations to a database and comparing the two results, the results of the backward simulation depending on those of the forward simulation and the float time for each individual task can be presented (Fig. 5).



	Task Name	Duration	Predece	Resource
1	Abutment 1 Construct Formwork	3 days		R1
2	Abutment 1 Assembly Tier	4 days		R2
3	Abutment 1 Fill Concrete	1 day	1;2	R3
4	Abutment 1 Remove Formwork	1 day	3	R2
5	Pier Construct Formwork	3 days		R1
6	Pier Assembly Tier	2 days		R2
7	Pier Fill Concrete	1 day	5;6	R3
8	Pier Remove Formwork	1 day	7	R2
9	Abutment 2 Construct Formwork	3 days		R1
10	Abutment 2 Assembly Tier	4 days		R2
11	Abutment 2 Fill Concrete	1 day	9;10	R3
12	Abutment 2 Remove Formwork	1 day	11	R2
13	Beam 1 Transport	1 day		R3
14	Beam 1 Lift	1 day	13;3;7	R3
15	Beam 2 Transport	1 day		R3
16	Beam 2 Lift	1 day	15;7;11	R3
17	Asphalt	3 days	14;16	R3

Fig. 5: Results of the communicating simulations: Float times (backward simulation results transparent grey)

The results in Fig. 5 demonstrate that the calculation of float times using discrete-event simulation is achievable. In this particular case most of the tasks have float times, those tasks without float times represent the *critical path* of this simple project: Abutment 1 Assembly Tier → Abutment 2 Assembly Tier → Abutment 2 Fill Concrete →Beam 2 Lift → Asphalt. It is important to note that if two tasks have float times and no technological dependency on each other but use the same resources, if the first is delayed, the second has to follow it in time as well as a result of the delay (e.g. Pier Construct Formwork and Abutment 2 Construct Formwork). Put another way, even if the formwork construction could be finished just before concrete filling according to the technological dependency, it has to be finished one day earlier so that the resources for starting the abutment 2 formwork construction at the latest possible time are available. Consequently, the consideration of resources in the calculation of float time for construction schedules is important as it can impact on (i.e. reduce) the actual float time for individual tasks. This could not have been identified without taking resources into account. Changing the configuration of the available resources will always result in a new schedule for both the forward and backward simulations, so all the tasks will have new float times and a new critical path will also need to be determined.

5 Conclusion

Calculating float times for construction projects taking into account the available resources is a challenging task. Conventional network planning techniques (e.g. critical path methods) are able to analyse and calculate float times but are not adequate for considering resources. By contrast, discrete-event simulation is capable of considering resources, but unable to calculate float times. In our research work we use the DESMO-J discrete-event simulation engine extended with a constraint-based discrete-event simulation methodology to generate construction schedules. In this paper, we introduced a new methodology that extends the constraint-based discrete-event simulation with the ability to calculate float times. For this we used the methodology of backward simulation in combination with forward simulation. The most challenging task in the backward simulation is to follow an identical schedule sequence to that of the forward simulation. In order to achieve schedule compatibility, every task has to start at the same time or later in time then in the forward simulation. To this end, we modified the task selection algorithm so that the backward simulation uses a priority-based approach to determine the next executable task. By comparing the results with the schedule of the forward simulation, the time difference between the earliest and latest start times of a task represents its float time. The tasks without float time comprise the critical path for the respective configuration of resources. Finally, we used a simple case study to illustrate the application of this new approach and showed that the determination of detailed float times for each individual task using discrete-event simulation taking into consideration available resources is realizable. It has been shown that taking resources in account is an important aspect in the calculation of float times as it can result in reduced float times for individual work tasks which would not have been foreseen without taking into account resource availability. In future work we will implement the possibility of using different resource calendars in order to be able to restrict the availability of some resources for different time intervals and examine the effect on the resulted schedules.

References

ABOURIZK, S., 2010. Role of Simulation in Construction Engineering and Management. *Journal of Construction Engineering and Management*, 136(10), 1140-1153.

BEISSERT, U., KOENIG, M. and BARGSTAEDT, H.-J., 2007. Constraint-based simulation of outfitting processes in building engineering. *In: 24th W78 Conference, 2007, Maribor, Slovenia.*

BEISSERT, U., KOENIG, M. and BARGSTAEDT, H.-J., 2008. Execution Strategy Investigation Using Soft Constraint-based Simulation. *In: IABSE Conference on Information and Communication Technology, 2008, Helsinki, Finland*.

CHANG, D. Y.-M., 1986. *RESQUE: A Resource Based Simulation System for Construction Process Planning*. PhD dissertation, University of Michigan, Ann Arbor, Michigan, USA.

DORI, G. and BORRMANN, A., 2011. Automatic Generation of Complex Bridge Construction Animation Sections by Coupling Constraint-based Discrete-Event Simulation with Game Engines. *In: ConVR2011, 2011, Weimar, Germany.*

HALPIN, D. W., 1977. CYCLONE - Method for Modeling Job Site Processes. *Journal of the Construction Division*, 103(3), 489-499.

HEGAZY, T. and MENESY, W., 2010. Critical Path Segments Scheduling Technique. *Journal of Construction Engineering and Management* , 136 (10), 1078-1085.

KOENIG, M., BEISSERT, U., STEINHAUER, D. and BARGSTAEDT, H.-J., 2007. Constraint-based simulation of outfitting processes in shipbuilding and civil engineering. *In: 6thEurosim Congress in Modeling and Simulation, 2007, Ljubljana, Slovenia*.

LU, M., 2003. Simplified Discrete-Event Simulation Approach for Construction Simulation. *Journal of Construction Engineering and Management* , 129(5), 537-546.

LU, M. and LAM, H.-C., 2008. Critical Path Scheduling under Resource Calendar Constraints. *Journal of Construction Engineering and Management* , 134(1), 25-31.

LU, M., LAM, H.-C. and DAI, F., 2008. Resource-constrained critical path analysis based on discrete event simulation and particle swarm optimization. *Automation in Construction*, 17(6), 670-681.

PAGE, B. and KREUTZER W., 2005. *The Java Simulation Book – Simulating Discrete Event Systems with UML and Java*. Aachen: Shaker Verlag.

RAZ, T. and MARSHALL, B., 1996. Effect of resource constraints on float calculations in project networks. *International Journal of Project Management*, 14(4), 241-248.

TOMMELEIN, I. D., CARR, R. I. and ODEH, A. M., 1994. Assembly of Simulation Networks Using Designs, Plans, and Methods. *Journal of Construction Engineering and Management*, 120(4), 796-815.