Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

INTEGRATION OF SIMULATION AND PARETO-BASED OPTIMIZATION FOR SPACE PLANNING IN THE FINISHING PHASE

Trang Dang
Hans-Joachim Bargstädt

Bauhaus Universität Weimar
Marienstr. 7a, 99423 Weimar
GERMANY

ABSTRACT

In order to improve the flexibility and adaptability of an automated model to various different projects under different circumstances, various solutions should be generated as proposed results, instead of only one solution as in recent research. This paper therefore presents a method for generating a series of reasonably detailed schedules for mapping the workplaces of activities over time. This model incorporates Pareto-based optimization and simulation. The optimization engine takes on the role for generating and choosing good schedules. The simulators, on the other hand, are responsible for manipulating activities to resolve spatial conflicts, to respect the limits of crews and investigate the efficiency of solutions. A prototype implementation is afterwards developed and implemented in software based on Building Information Modeling, which enables the model to automatically retrieve the required geometric data. The output solutions are finally analyzed through an example to prove their feasibility and adaptability to various potential situations.

1 INTRODUCTION

Construction projects involve various contractors. Communicating schedules and strategies among them are important tasks on a construction site. This is especially serious in the finishing stage of execution, when many of stakeholders, such as constructors, electricians, sanitations engineers, facility coordinators and other contractors work at the same time in a limited space. Besides, tasks in this period of execution have much time slack, so that there are various alternatives for the contractors to arrange for the schedule and adapt to their own conditions. If a detailed schedule including the mapping of subcontractors' work places over time is not created beforehand, space disputes between entities will occur. Through empirical studies of Mallasi and Dawood (2001), the lack of detailed planning has been indicated as 30 % of non-productive time. It has also been stated as one of the main reasons for the stacking of trades (Hana, Russell, and Emerson 2008). Generating a sufficiently detailed short-term schedule, therefore, is necessary for the finishing period, when several trades work together concurrently in a limited space.

A challenge for any efficient scheduling method is its adaptability. Every construction project has its own different characteristics such as its contractor's efficiency and capability to mobilize resources. In some cases, for example, in order to resolve a problem, compressed schedules can be considered as a solution if project managers are able to mobilize extra crews as well as extra financing. In other cases, however, it is easier to deal with the problem by leaving some activities suspended to give their work places to others first, if extra costs are acceptable, etc. Therefore, it is difficult to find only one specific solution which can fit every different context. This paper proposes a method which is able to generate various alternatives for arranging activities to deal with spatial conflicts. From this, construction managers can choose the best adaptive solution which is suitable to their situation.

The model presented is an integration of simulation and heuristic optimization. Firstly, two simulators are built, which simulate the way managers deal with conflicts on site. Secondly, an optimization engine is developed. They play a role as the control center of the model. The simulators respond to analyze the behavior of schedules considering workspace and crew requirements. They investigate how much spatial congestion occurs and how these congestions may impact on the schedule in return. In this process, activities are considered able to be interrupted (if possible). The optimization engine will be based on the evaluation results of the simulation process to generate a set of schedules that are "near" the Pareto front. Finally, feasibility and flexibility of solutions given out are analyzed via an example application.

2 RELATED RESEARCH AND MOTIVATION

Many researchers have been interested in space planning and confirm the necessity of workspace management during planning and scheduling in detecting workspace conflicts. This allows users to be aware of situations in advance which are occurring on site and then make decisions based on their experience and intuition. Along with the research that just focuses on generating workspace requirements and detecting workspace clashes (Akinci et al. 2002), some others consider searching for strategies to overcome these problems. Currently, there are two directions in dealing with spatial conflicts. The first direction is to create a schedule which eliminates all potential spatial congestions (Jongeling and Olofsson 2007; Zhang et al. 2007; Elmahdi, Wu, and Bargstädt 2011); and the second direction is to adjust a schedule to minimize congestion (Mallasi 2006; Winch and North 2006; Bansal 2011).

Application of a conflict-free method, which follows the first direction mentioned above, for look-ahead planning may be not feasible. In order to keep the flexibility of planning, detailed schedules are normally developed for three weeks ahead as an expansion of a part of macro-schedules in the execution phase. At this time, the time frame of a project is already determined. Manipulating activities to eliminate all workspace conflicts may then cause serious delays of the project. This makes the solution of the method insufficient.

The research in this paper follows the later direction proposed by Mallasi, Winch & North and Bansal, which tries to keep congestions minimal within an acceptable time frame of the project. In order to search for schedules with minimal congestion, researchers have used different algorithms. Mallasi integrates a simple genetic algorithm with different work rates and execution patterns to find out the best solution. However, Mallasi considers fixed start dates of activities. This assumption makes the approach inflexible to benefit from time slacks of activities in the finishing phase. Winch and North use the "brute force" algorithm. That means that they investigate almost all possibilities of adjusting the schedule before choosing the best one. Bansal suggests a model, in which users can manually adjust a schedule and the spatial requirements, or split activities to find a suitable solution. These two later models are time-consuming and not compatible with large-scale searching like approaches at the scheduling problem.

Moreover, all of these approaches, except the manual model of Bansal (2011), do not allow an activity to be interrupted. This means that whenever an activity is started, it will occupy resources, e.g. workspace, etc., until it is finished. This reduces the number of potential schedules to search for. It is also not able to reflect the priority of critical tasks. In reality, sometimes tasks must be interrupted in order to give additional resources to other tasks belonging to the critical path when they need it.

Another disadvantage of the researches mentioned above is the limitation of the number of results given. All of them provide one and only one solution. This limits the adaptability of the research in varied construction areas, because it can be much different from one project to another. Even in the same project, there might be different stages and situations, which need different measures.

3 INPUT DATA SYSTEM

In order to enable the model automatically to invoke input data, a 4D model has been developed by connecting a schedule (2D) and a product model (Figure 1). This issue has already been considered thoroughly in previous research (Tulke 2010; Tauscher 2011). So it is not discussed here in detail. We just use a simple XML file as a data template for information interacting between a schedule and a

product model. Then an activity can automatically be assigned to objects, which have corresponding categories defined in the template and which lay on the right floor as required. After that, based on the kind of trade and positions of the products, workspaces can automatically be generated (Dang, Elmahdi, and Bargstädt 2012). This kind of work has been programmed in Visual C#® with the support of the namespaces Revit API® and the control ActiveGantt®.

In this model, the data of an activity includes three types: 1) information of a schedule such as earliest start/end date, free slack, total slack, constraints with other activities, kind and number of crews required; 2) information of geometry data such as location of products and location of the products' workspace correspondingly; and 3) task property which defines whether or not this task can be interrupted.

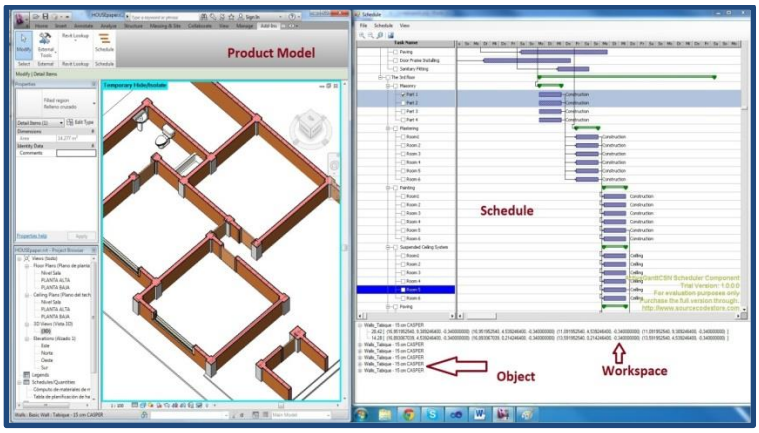


Figure 1: 4D Model developed in Revit Architecture environment

4 TIME-BASED SIMULATION MODEL

In order to investigate time-space conflicts and allocate workspaces to activities, a time-based simulation model has been developed.

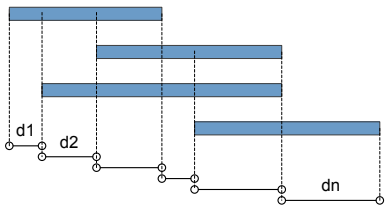


Figure 2: The time interval of the time-based simulator

In this model, the time interval is not constant during the simulation process. It is defined as the duration between the current simulated point and the nearest start/end date of the investigated activities (Figure 2).

During the simulation process, workspace and crew are considered as "near" hard constraints. This means a specific workspace and a crew are assigned to only one activity at a point of time. If two or more tasks, according to the schedule, require the same space concurrently, then only one task is allowed to occupy this place and the others must be moved to a later time, unless these tasks do not have enough time to be moved due to the overrun of their total slack and the accepted delay defined in the input data. In this case they must stay there and the workspace conflict will be counted in the procedure of the schedule evaluation. Similarly, if at a point of time, a crew requirement is beyond a crew's capability, some tasks must be moved in order to deal with this inadequacy unless they have no time buffers.

However, such a process causes a serious problem. Multiple congestions seem to have a tendency to meet together at the end point, at which the tasks have no time buffer to be manipulated. Therefore it is difficult to decide on the right solution on site based on this schedule. In order to diversify the results and suitably distribute the conflicts, this paper proposes two simulators: 1) the tense conflict simulation (TCS); and 2) the distributed conflict simulation (DCS).

It is important to mention that the input data for these simulation processes includes the acceptable delay duration of the project. In both of TCS and DCS, tasks that have start-dates earlier will have a priority of receiving workspaces if disputes occur, unless tasks, which have start-dates later, have no time-buffer.

The TCS considers that an activity can be interrupted (if its property is interruptible) and conflicts just occur if the corresponding tasks no longer have time-buffers to be moved. The simulation process can be presented in detail as follows.

currentDuration is assigned as d1 (figure 2);
do
{
1. List all tasks investigated taking place in the currentDuration
2. If any couple of tasks in the list requires the same workspace & none of them has not been moved to later & at least one of their (total slack + acceptable delay) is greater than zero, then
a. choose the task to be allowed to occupy the workspace based on the following priority
i. for the case: at least one of them is interruptible
1. Its property is not interruptible & its start-date is earlier than currentDuration
2. Its (total slack+accepted delay) is zero
3. Its start-date is earlier
4. Its (total slack + accepted delay) is smaller
5. If they have the same (3) and (4) values, then choose one of them randomly
ii. for the case: both of them are not interruptible & at least one of their start-dates is within currentDuration
1. Its start-date is earlier
2. Its (total slack + acceptable delay) is smaller
3. If they have the same (1) and (2) values, then choose one of them randomly
b. move the not-completed part of the not-chosen task at step (a) to the later time, so that after this step its (total slack + acceptable delay) is not smaller than zero and its new start-date is not later than one day from the end-date of the chosen one.
c. update the whole schedule based on its constraints after moving the not-chosen task.
3. List all tasks investigated taking place in the currentDuration (some of tasks have been moved in step 2).
4. If any crew group is inadequate, list all the tasks being conducted in this period requiring this resource, then
a. take a task out of the list if
i. either its property is not interruptible & its start-date is earlier than currentDuration
ii. or its (total slack + accepted delay) is zero
b. for the tasks still existing in the list, choose some tasks to be moved so that the crew requirement is not greater than its capability and the number of tasks moved is minimum. In the case that the requirement is still greater than its capability after moving all tasks in the list, the crew overruns will be generated.
5. currentDuration is assigned as the nextDuration (figure 2)
while (currentPoint is not the lastPoint yet)

Figure 3: The simulation process of TCS

Like the TCS, the DCS also allows to interrupt a task (if possible) in the simulation process. Unlike the TCS, however, a spatial congestion and a crew overrun in the DCS can occur even if the corresponding tasks still have time-buffers to be moved. In order to produce this kind of result, a change from TCS has been made. In step 2, before choosing a task being allowed to occupy the disputed place, a random number will be created with 20 % probability that they will stay together at that time; hence, the congestion will be generated. Of course, the number of the probability can be adjusted to be less or greater than 20 % but for the experiments so far, 20 % has been reasonable for this kind of problem.

5 PARETO-BASED OPTIMIZATION MODEL

The optimization model tries to adjust the schedule in order to find the solution which has acceptable values for their objectives, such as project lead time, conflict duration, and number of workers inadequate,

etc. In order to deal with multiple objectives, there are three principle methods (Mumford-Valenzuela 2005).

(1) Combine all the objectives into a single scalar value by using weighted factors corresponding to objectives, and optimize for the scalar value.
(2) Arrange the objectives in a priority order, optimize for the first objective, then if there is more than one solution, optimize these solutions for the second objective, and repeat for the third, etc.
(3) Consider all objectives equivalent; find a set of non-dominated solutions, in which when attempting to improve an objective further, the other objectives suffer as a result. This is called Pareto optimal and the set of non-dominated solutions is called Pareto front.

The methods (1) and (2) only give one solution. This may ignore good solutions which do not have the best value for a particular objective, but it makes good sense if all objectives are considered together. With the idea of providing the best information to managers, the approach of this research here chooses the method (3). Such a method involves no judgment and produces a set of viable alternatives from which a decision maker can reach an informed selection at a later stage.

5.1 Objective Definition

Five parameters are taken into account in this paper. They can be categorized into two groups. The first group called "project properties" contains: project lead time, conflict duration and crew overrun. The second group, which is called "feasibility properties", contains: split number and conflict number. The optimization process aims to keep the values of these five parameters at a minimum.

$$Objectives = \ to\ minimize \begin{cases} - \ project\ properties & \begin{cases} project\ lead\ time\ (1) \\ conflict\ duration\ (2) \\ crew\ overrun\ (3) \end{cases} \\ - \ feasibility\ properties & \begin{cases} split\ number\ (4) \\ conflict\ number\ (5) \end{cases} \end{cases}$$

5.1.1 Project Lead Time

Project duration is always one of the most important factors in construction management. In this approach, project duration is indirectly regarded as being under the parameter **the project lead time.** This parameter is considered an important item to evaluate the adequacy of a schedule. This value of the lead time always lies within the acceptable delay duration which is given as an input data before an investigating process. The longer an acceptable delay duration is given, the larger a searching domain is investigated. This issue is discussed more clearly in the section 5.7.

Project lead time *= actual end date of project – as planned end date of project* (1)

5.1.2 Conflict Duration

Although the manipulation of an activity in the simulation process is conducted to resolve workspace conflicts, spatial congestion still exists in a schedule depending on the acceptable delay duration. In this research **conflict duration** is regarded as the second objective, which must be taken into account by the optimization process. This value is counted as the number of days in which the schedule shows workspace conflicts. A better solution is associated with a shorter conflict duration.

$\boldsymbol{conflict\ duration} = \sum_{k=1}^{n} d_k$, where: (2)

- n is the number of occurrences of workspace conflicts
- d_k is the number of days of the k^{th} occurrence.

5.1.3 Crew Overrun

In the evaluation process, overrun of labor is also important to evaluate in a schedule. If this information is not taken into account, a good theoretical solution can be achieved by letting all activities, which are essentially the same kind of trades and have different workspace requirements, take place at the same time. Such a schedule, however, is not practical, since the number of crews is not infinitive. In reality, dealing with the limitation of the number of workers is also a serious problem in construction management on site. Therefore, in this approach, crew overrun is considered as the factor for evaluating a solution. It is defined as the total number of laborers exceeding capability.

$$\textbf{\textit{crew overrun}}\ (\%) = \max\left\{\frac{CR_i^k - CC_i^k}{CC_i^k} \times 100\right\}; i = 1 \div n; k = 1 \div m \quad \text{, where} \tag{3}$$

- n is the number of periods in which the laborer requirements are beyond capability.
- m is the number of crew groups whose capability is smaller than required.
- CR_i^k is number of Crew Requirements of the k^{th} crew group at the i^{th} duration
- CC_i^k is number of Crew Capability of the k^{th} crew group at the i^{th} duration

5.1.4 Split Number

Beside of the objectives which present clearly negative consequences of a solution, like the three parameters referred above, some others do not. But, they are very important to identify the feasibility of a schedule. One of them is relevant to the splitting number of a task. As mentioned in the simulation, an activity once split means that it has to be suspended in order to give its work place to another task with a higher priority. This may lead to a necessary reallocation of equipment and materials as a consequence. Therefore, if a task has too many interruptions, the schedule corresponding is not feasible to be chosen as a solution. In order to take into account this problem, the parameter named "split number" will be considered.

$$\textbf{\textit{split number}} = \max\{\textit{the interruption number of the task } i\}, i = 1 \div n \tag{4}$$

There, n is the number of tasks investigated.

5.1.5 Conflict Number

Beside the split number, the number of tasks conflicting at one point in time should be kept to a minimum in order to make a solution feasible. The more number of tasks dispute each other for the same work place, the more managers must deal with them. This parameter is called the **conflict number**.

$$\textbf{\textit{conflict number}} = \textit{max } \{(\textit{number of tasks having the same workspace at duration } i)\text{-}1\}\ (i=1 \div n)$$

(5)There, n is the number of durations, which is referred in Figure 2.

5.2 Chromosome Definition

A chromosome is a string of DNA and it is used in this paper to represent an actual part of a schedule (Figure 4). The number of DNA in a chromosome is one unit greater than the number of tasks which must be investigated considering workspace and crew requirements. Except the last DNA, which contains information about the acceptable lead time of the project for the simulation process, the others provide the information about the duration which is counted from the early start date to the actual start date of tasks.

In this approach, an individual is also considered as a chromosome. So in this paper, the terms "chromosome" and "individual" can be used interchangeably.

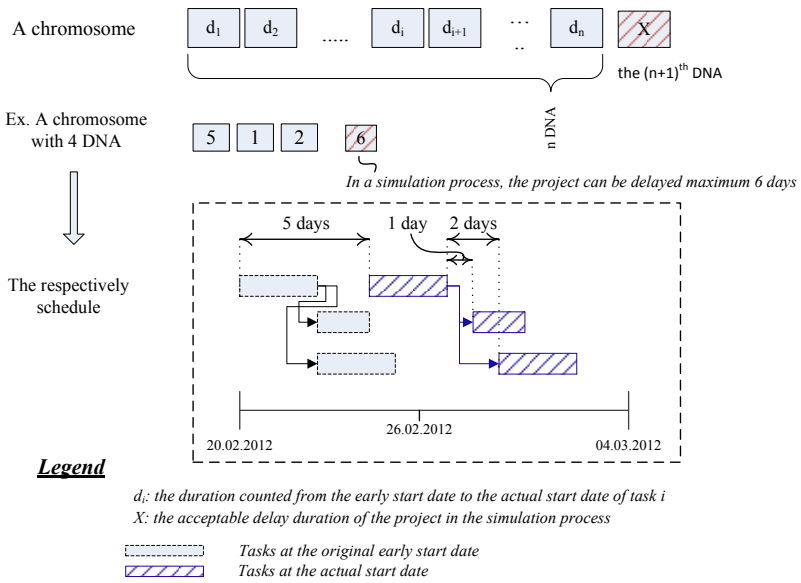


Figure 4: Chromosome

5.3 Original Generation

An individual in the original generation is created by taking (n+1) random numbers. There, n is the number of tasks which is needed to investigate. For the first n DNAs, d_i is defined by a random number which lies within zero and the total slack of the task i. For the last DNA, X is identified by a not-negative random number which is not greater than the acceptable lead time of the project which is predefined as an input data of the model.

According to what is presented in the section 5.2. Such a way to generate an individual may cause the project delay duration greater than the X value, after updating d_i to obtain actual dates of tasks. If this case occurs, X will be assigned again with the value of the project delay duration if it is not greater than the acceptable lead time of the project. Otherwise, this individual will be taken out of the population.

By using the X variable for each individual, it assures the diversity of population when the acceptable lead time is great. It should be noted that, however, the greater the acceptable lead time is, a larger the scale of the population should be required.

5.4 Crossover Operator

A simple crossover operator is applied to generate an offspring. From two parent schedules, a crossover position is randomly chosen, the first parent provides the first part of schedule for the offspring, and the second provides the rest. The offspring also takes the last DNA from either of its parents. In this approach, all of individuals in the population join the crossover process. After this process, the offspring has a 10% probability of being followed by a mutation process before being analyzed and evaluated.

5.5 Mutation Operator

In the mutation operation, a position (x), number of tasks (n) and the day to be changed (t) will be randomly created first. Notice that it may be either negative or positive. The mutation operator will be then applied for n continuous activities from the position x by moving them t days uniformly.

5.6 Selection Process

In order to generate successive generations which are moved ever closer to the Pareto front, the simple evolutionary algorithm for multi-objective optimization (SEAMO) (Figure 5), which was developed by Mumford (2010), is adopted. This algorithm uses a replacement strategy to move the solutions during the searching process ever closer to the Pareto front, and to widen the spread of the solution set. Like its name, this algorithm is kind of simple and as such this property suits the approach. Honestly, the simulation process, which is used to evaluate a chromosome (or rather a schedule in this approach), takes time. Therefore, a strategy, which is simple but able to generate diversified solutions in the end, is of priority to be chosen for the approach.

It should be mentioned that this algorithm however has been modified a little bit in the approach. Instead of checking all individuals of a population in order to know whether or not an individual exists, which is dominated by the offspring, the approach takes a maximum 20% of a population randomly. In this way, the modification gives individuals an equal probability of being replaced regardless of their positions in a population.

5.7 Working Mechanism of the Optimization Engine

The optimization engine is described in Figure 5. The process commences with generating an original generation. For each individual in this generation, the simulation process is then applied to evaluate its value. The record of the global best-so-far for objectives is created by taking the minimum number of its values from the individuals evaluated.

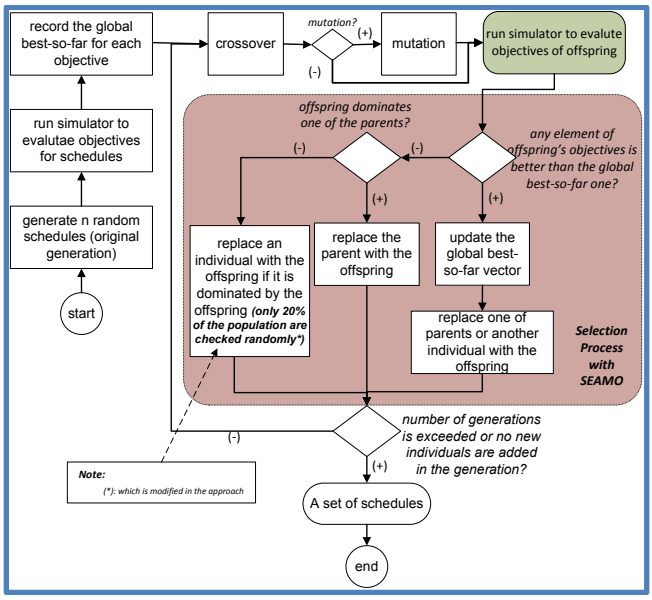


Figure 5: Optimization Engine

After this, all individuals take part in the crossover process and 10% of them continue with mutation to generate offspring. Once an offspring is born, it will be evaluated by a simulator and pass through the selection process to know whether or not it can exist by replacing an existed individual in the population. The optimization ends whether the possible number of evaluated generations is exceeded or no new individual comes into the population.

It should be reminded again that two simulators are applied in the approach (TCS and DCS). If both of them are used for one population, by choosing a simulator to analyze a schedule randomly for example, the population required must be large and it takes a long time to obtain the converged solutions. Therefore, two independent populations have been used. One uses TCS and the other uses DCS in the

analyzing and evaluating process of a schedule. After these two optimization processes, the results will be combined together and a filtering process will be applied in order to take dominated chromosomes out of the population. As a result, a set of schedules which is considered "near" the Pareto front is presented.

6 EXAMPLE APPLICATION

The model is experimented in the finishing period of a building floor. The trades involved in this experiment include masonry, plastering, painting, installing suspended ceiling systems, installing windows and doors, paving and installing sanitary facilities.

The masonry trade contains 4 activities which respond to 4 regions of the walls; the sanitary fitting is completed with only one activity; and the others are divided into 6 activities corresponding to 6 different rooms. In summary, 35 activities are investigated.

The optimization will be conducted with the acceptable lead time of 4 days. A population includes 50 individuals and the number of generations is 5. As a result, a set of schedules is generated. Each value vector of objectives may contain several schedules.

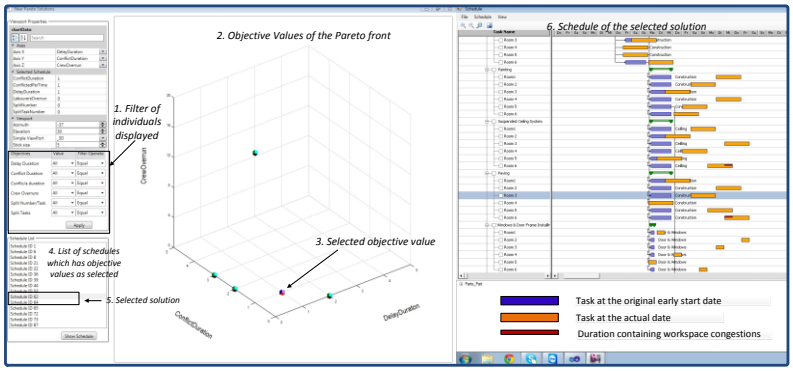


Figure 6: Screen shot of a result

It is also noticed that an objective vector contains 5 parameters and the graph is only able to show a maximum of three values at one point in time. Therefore, it is necessary to use the filter and the axis data setting to enable investigating all sides of the solutions.

According to the objective vectors in the graph, the "near" Pareto-front solutions have the minimum delay duration of zero and the maximum delay of two days. With two days delay, the schedule has no time-space conflicts and crew inadequacies. The following resulting schedules are considered in order to analyze the efficiency of solutions from a construction manager's perspective.

Case 1: the delay duration is zero (Figure 7). *For the schedule number 1*, the problems which managers must face include both workspace congestions and the inadequacy of the crew for installing windows and doors with an amount of 100% (in this case it means two laborers). However, if the trade plastering at the room 1 can be interrupted like what *the schedule number 3* has shown, the crew overrun would not occur. Therefore, the only problem to be dealt with is the workspace dispute.

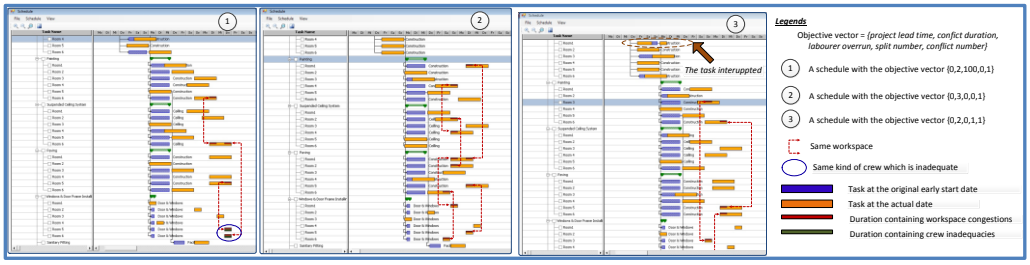
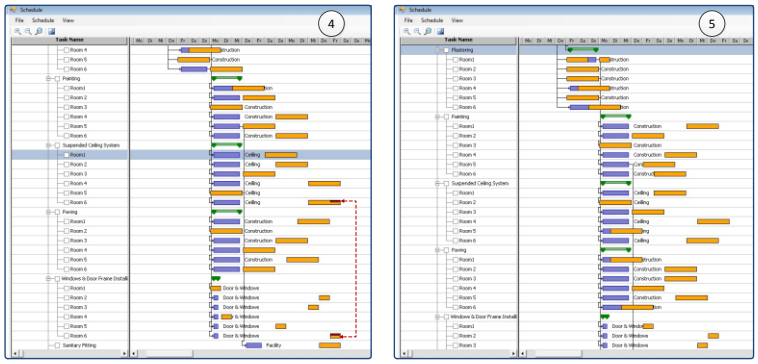


Figure 7: Solutions without delay

For the schedule number 2, the crews' requirements are always within their capability, i.e., no interruption of tasks is required. However, this schedule brings many spatial disputes, especially, the conflict between installing suspended ceiling systems and paving. This conflict is very serious since both of them normally require a whole room for their works. So this solution is not feasible.

***Case 2*:** the delay duration is one day (Figure 8). *For the schedule number 4*, congestion occurs only in one day between the trades installing suspended ceiling systems, and installing windows and doors. If a one-day delay can be accepted, this solution is considerable since this kind of congestion can be solved on site.



4 One of schedules having the objective vector {1,1,0,0,1}

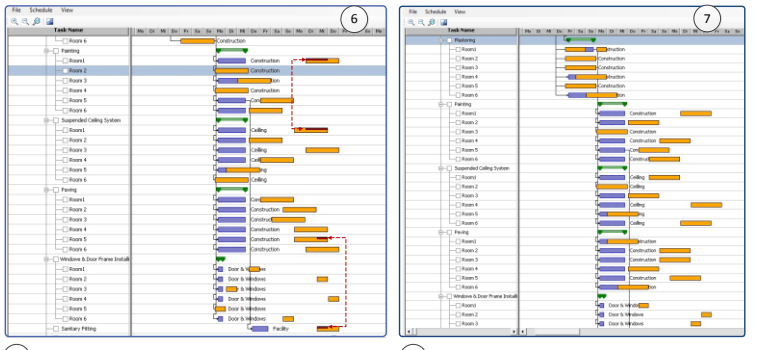
5 One of schedules having the objective vector {1,0,0,0,1}

Figure 8: Solution with one day delayed (4) and two days delayed (5)

Case 3: the delay duration is two days (Figure 8). *For the schedule number 5*, if it is the case that a two-day delay for the project is acceptable, this schedule is also feasible. It has no inadequacies of crews, no tasks must be interrupted and no time-space conflicts occur during the construction process.

7 DISCUSSION

The case study mentioned above is experimented with just 5 generations for the evolutionary process. The result would be really better if this case were to be carried out with 10 generations. Although the delay duration of the project is still 2 days in order to deal with all of the problems, the solution is more feasible when the delay duration is zero. When the delay duration is one day, with just an interruption of the trade plastering at room 1, workspace congestions or crew inadequacies no longer occur (Figure 9).



6 One of schedules having the objective vector {0,2,0,0,1}

7 One of schedules having the objective vector {1,0,0,1,0}

Figure 9: Solutions of an optimization with 10 generations

One of the matters, which should also be regarded here, is how better the SEAMO has been worked into the approach compared to a random searching algorithm. An experiment with a random search is implemented. For the SEAMO, the experiment has been conducted with a population containing 50

individuals and 10 generations. For random searching, the experiment has been conducted with a population of 12250 individuals; this number is equal to the maximum number of individuals, which has been checked in the experiment with SEAMO.

The results have confirmed the efficiency of using SEAMO compared to a random searching (Figure 10). With the SEAMO, the results have converged and the maximum delay duration is just two days in order to get other parameters' values to zero; otherwise, this number when using random searching is three days. In addition, when the delay duration and laborer overrun are zero, schedules resulted from SEAMO have two days containing spatial congestions; however, with a delay duration of zero, the minimum conflict duration in random searching is five days. With just two objective vectors extracted from the results, it is enough to recognize how much more efficiently the optimization using SEAMO works compared to random searching.

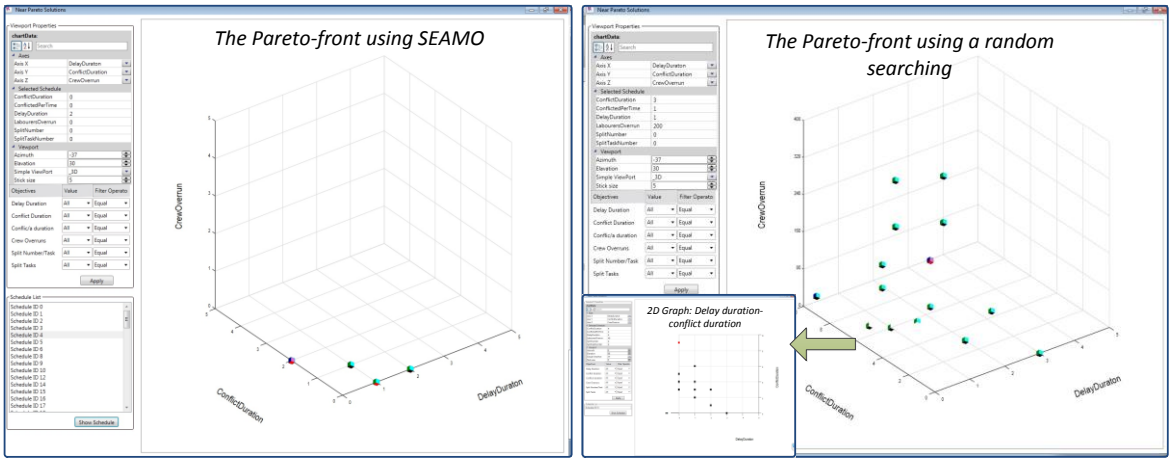


Figure 10: Pareto-fronts with different searching methods

8 CONCLUSION

The goal of the approach is to find a set of feasible strategies which resolve time-space conflicts and limits of crews. The integration of simulation with evolutionary algorithm has been successfully achieved. Like other results from random searching techniques, different schedules given through the proposed method in this research are really "diversified" and the searching process also converges quickly. Therefore, decision makers can choose the suitable solutions depending on their individual conditions such as crew size, and material quality and quantity, etc.

Besides, project managers are able to evaluate solutions efficiently and make their decisions based on the proposed methodology either for the whole schedule or for a selected part of the schedule (short term activities). However, the authors recommend the use of this methodology for short term activities. Investigating a whole project is quite time consuming and still requires much detailed information. Moreover, a detailed schedule for a whole project loses the flexibility of planning and is therefore not feasible in practice.

REFERENCES

Akinci, B., M. Fischer, R. Levitt, and R. Carlson. 2002. "Formalization and Automation of Time-Space Conflict Analysis." *Journal of Computing in Civil Engineering* no. 16 (2):124-134.

Bansal, V. K. 2011. "Use of GIS and Topology in the Identification and Resolution of Space Conflicts." *Journal of Computing in Civil Engineering* no. 25 (2):159-171.

Dang, T., A. Elmahdi, and H.-J. Bargstädt. 2012. Generating Workspace Requirements in a finishing execution phase. Paper read at 12th International Conference on Construction Application of Virtual Reality, at Taiwan.

Elmahdi, A., I.-C. Wu, and H.-J. Bargstädt. 2011. 4D Grid-based simulation framework for facilitating workspace management. In *11th International Conference on Construction Applications of Virtual Reality*. Weimar Germany: Bauhaus Universität Weimar.

Hana, A. S., J. S. Russell, and E. O. Emerson. 2008. "Stacking of Trades." In *Construction Productivity: A Practical Guide for Building and Electrical Contractors*, edited by Eddy M. Rojas, 75-110. J Ross Publishing.

Jongeling, R., and T. Olofsson. 2007. "A method for planning of work-flow by combined use of location-based scheduling and 4D CAD." *Automation in Construction* no. 16 (2):189-198. doi: 10.1016/j.autcon.2006.04.001.

Mallasi, Z. 2006. "Dynamic quantification and analysis of the construction workspace congestion utilising 4D visualisation." *Automation in Construction* no. 15 (5):640-655. doi: 10.1016/j.autcon.2005.08.005.

Mallasi, Z., and N. Dawood. 2001. Assessing space criticality in sequencing and identifying execution patterns for construction activities using VR visualisations. In *ARCOM doctoral research workshop: Simulation and modelling in construction*. Edinburgh University, UK.

Mumford-Valenzuela, C. L. 2005. "A simple approach to evolutionary multiobjective optimization." In *Evolutionary Multiobjective Optimization - Theoretical Advances and Applications*, edited by Lakhmi Jain Ajith Abraham, Robert Goldberg, 55-104. United States of America: Springer.

Mumford, C. 2010. "A multiobjective framework for heavily constrained examination timetabling problems." *Annals of Operations Research* no. 180 (1):3-31. doi: 10.1007/s10479-008-0490-3.

Tauscher, E. 2011. *Vom Bauwerksinformationsmodell zur Terminplanung: Ein Modell zur Generierung von Bauablaufplänen*. Dotoral Dissertation, Department of Civil Engineering and Construction, Bauhaus Universität Weimar.

Tulke, J. 2010. *Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen*. Doctoral Dissertation, Bauhaus Universität Weimar.

Winch, G. M., and S. North. 2006. "Critical Space Analysis." *Journal of Construction Engineering and Management* no. 132 (5):473-481.

Zhang, C., A. Hammad, T. M. Zayed, G. Wainer, and H. Pang. 2007. "Cell-based representation and analysis of spatial resources in construction simulation." *Automation in Construction* no. 16 (4):436-448. doi: 10.1016/j.autcon.2006.07.009.

AUTHOR BIOGRAPHIES

TRANG DANG is a Ph.D student in the Institute of Construction Engineering and Management at Bauhaus Universität Weimar. Her research interest is the automation of generating a detailed short-term schedule. This process involves Building Information Modeling (BIM), Simulation and Optimization. Her email address is trang.dang@uni-weimar.de.

HANS-JOACHIM BARGSTÄDT is Professor for Construction Engineering and Management at the Bauhaus Uiversität Weimar. He studied in Braunschweig, Atlanta and Marseille. He got his Ph. D. from the Technical University Braunschweig in 1988. His research fields are construction processes, construction management, simulation in construction, construction in the built environment and lifecycle considerations. His e-mail is hans-joachim.bargstaedt@uni-weimar.de.