Ingenieurfakultät Bau Geo Umwelt
Lehrstuhl für Computergestützte Modellierung und Simulation

Reactive Simulation of Shoring and Excavation Processes based on Automated Performance Monitoring

Maximilian Bügler

Vollständiger Abdruck der von der Ingenieurfakultät Bau Geo Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. rer. nat. Ernst Rank
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. André Borrmann
	2. Univ.-Prof. Dr.-Ing. Markus König Ruhr-Universität Bochum

Die Dissertation wurde am 22.06.2016 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 17.10.2016 angenommen.

Abstract

This Thesis presents novel concepts for reactive simulation of shoring and excavation processes. In this context methods for construction schedule optimization are combined with techniques for real-time data acquisition. Furthermore uncertainty in the durations of activities and the performance factors of involved resources are considered. This Thesis presents a new schedule generation scheme, two heuristic scheduling algorithms, and a concept to automatically track excavation processes.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Dr.-Ing. André Borrmann for the continuous support of my research, for his patience, knowledge, and motivation. His guidance helped me throughout the four years of my research and the writing of this Thesis. I could not have imagined a better advisor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my Thesis committee: Prof. Dr.rer.nat Ernst Rank, and Prof. Dr.-Ing. Markus König, for their insightful comments and encouragement.

My sincere thanks also goes to Prof. Dr. Daniel Straub, Prof. Patricio Antonio Vela, Prof. Dr.-Ing. Dipl.-Wi.-Ing. Willibald A. Günthner, and Prof. Dr.-Ing. Christoph van Treeck, who gave me opportunities to collaborate in joint research endeavors.

I thank the Bavarian Research Foundation for funding the FAUST project which I worked on throughout my study.

Furthermore I thank my fellow colleagues for the stimulating discussions, the hard work on joint publications, the unforgettable experiences we had during conference travels, and the amazing four years we have spent together.

Last but not the least, I would like to thank my family: my parents and my brother for supporting me throughout writing this Thesis.

Contents

1 Introduction

Unexpected delays on construction sites, caused by internal and external influences, create huge costs every year and account for up to 30 percent of the overall costs of construction projects (Koehn et al., 1978). Common problems include coordination of different involved companies, weather, traffic, archaeological discoveries, leftover ammunition from wars, changes in the client's wishes, scarce resources, and neighborhood repugnance (Assaf & Al-Hejji, 2006).

> *"Construction planning is the most crucial, knowledge-intensive, ill-structured, and challenging phase in the project development cycle due to the complicated, interactive, and dynamic nature of construction processes."* - Halpin & Riggs (1992)

Due to the immense complexity of large scale construction sites, construction schedules are highly sensitive to any changes in the sequence of activities to be performed. Those detailed schedules often get rendered invalid after a short time and have to be updated, or new ones have to be created, taking newly arisen circumstances into account. In order to improve this error-prone process, ways to accompany construction using a simulation model and optimization techniques are investigated in this thesis to support the maintenance of the plans. Compared to a-priori simulation this approach requires data to be recorded on-site, to be able to determine the progress of the individual activities. Eventually this will cause the circumstances in the simulation model to adapt to the actual state of the project. This thesis aims to describe the development of the required methods and tools for the case of urban excavation and shoring projects with strong spatial constraints. This includes methods to extract the required data from the construction site in an predominantly automated way, determine the progress of the individual activities and make predictions on the further progress. When data in the simulation model is updated it has to be determined whether re-planning would be worth-while. Since the re-planning process is very time intensive for a human to perform, optimization algorithms were developed and analyzed in order to automate this process. Furthermore the uncertainty that is naturally attached to construction processes needs to be taken into account. The concept of the resulting simulation-optimization cycle is illustrated in Figure 1.1.

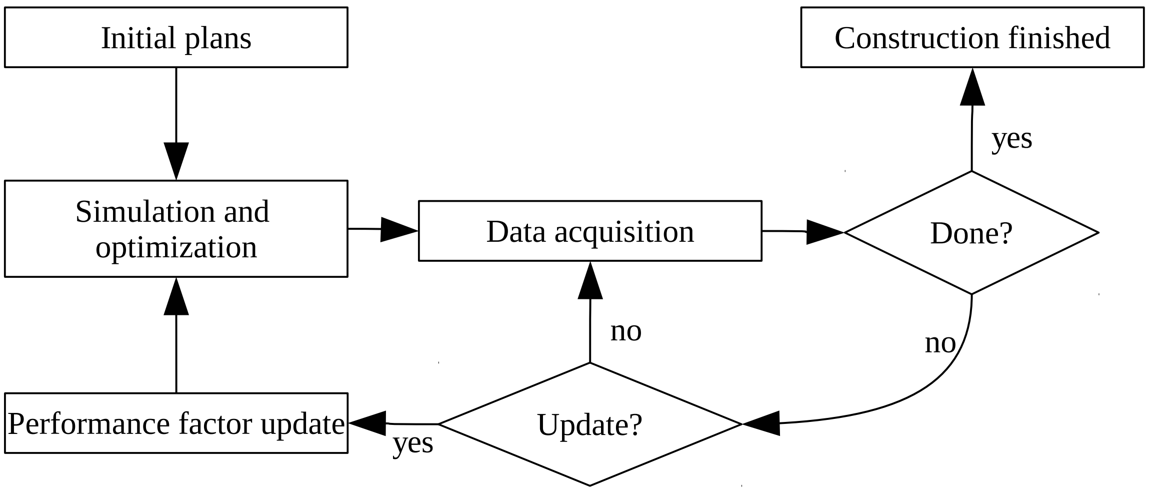


Figure 1.1: Flow chart of the optimization-simulation cycle. The process starts with an initial plan that undergoes a simulation and optimization process. During the execution of the plan, data is acquired about the progress. Unless the plan execution is done, the performance factors can require updating, which causes another simulation and optimization run, updating the plans respectively.

1.1 Problem Statement and Scope

A construction project consists of a number of activities that all need to be executed for the project to be finished. Additionally there may be milestones that have to be met to avoid fines. Each individual activity is described using a number of features. Those include the time required for completion, required resources, materials, and predecessor constraints. Resources include machines, workers, and physical space, while materials include concrete, bricks, and steel. The duration of an activity can furthermore be uncertain, hence subject to an underlying probability distribution.

As there are numerous possibilities of delays occurring during the execution of large construction projects, plans that are created a-priori are often subject to changes and can rarely be met in detail. This causes substantial costs for all involved parties, as missed milestones are fined and delayed project completion causes a decrease in payoff for investors. Therefore it is desirable to notice problems as early as possible and react in an adequate manner. This requires real-time data acquisition techniques, such as tracking of machines, or the progress of certain activities. Results include updated performance factors, activity durations, or progress ratios. Additionally methods to verify the plans' validity, given the new circumstances, are required. If the plans become invalid and cannot be updated easily, ways to generate new plans are of strong interest.

The problem of scheduling activities under precedence and resource constraints, in order to optimize some criteria is called the Resource Constrained Project Scheduling Problem (RCPSP) (Wiest, 1962). Figure 1.2 shows a graphical notation of such a problem used throughout the thesis, including resources, materials and geometric spaces. The rectangular boxes describe the individual activities of the project while the continuous arrows connecting the activities represent precedence relationships. For instance activity A needs to be executed prior to activity B. The round rectangles represent resources, while the finely dashed arrows imply resource usages. A number next to the arrows indicates the number of resources of that kind required. If no number is given, this defaults to one. In order to make representations of simple examples more lean, the colors and hatching of the activity rectangles also indicate the resource used. In this description we distinguish materials, which do not remain available after use, from resources, which can be used multiple times. Materials are illustrated using white hexagons, connected by dashed arrows. Similarly to the resources, numbers next to arrows, represent amounts. Finally the overlapping parallelograms indicate geometric resources. The overlaps indicate which areas or volumes overlap. A feasible solution to this kind of problem can be illustrated as a Gantt-chart showing the start and end times of the individual activities (Clark et al., 1922).

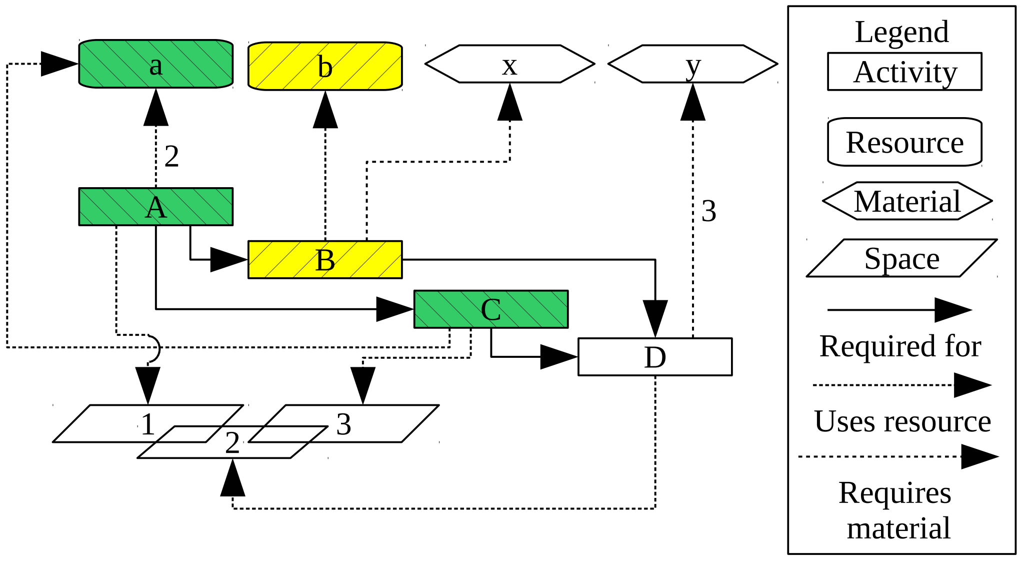


Figure 1.2: Scheme of a Resource Constrained Project Scheduling Problem (RCPSP). Rectangles represent activities, while their fill color and hatching indicates the resource that is required. The hexagons represent required materials while the rounded rectangles represent resources. The parallelograms represent geometric spaces. Dotted arrows indicate usages, while solid line arrows represent precedence constraints.

Additionally uncertainty is an important factor in the planning process of construction projects. Different soil conditions may cause varying performance of excavators or drills, while traffic congestion influences the time dump trucks need to transport excavated soil to a dump site. This kind of uncertainty needs to be included into the simulation and optimization process and in this way, can give rise to the robustness of an existing schedule. This can be done by representing the uncertainty using probability distributions. Furthermore a measure for robustness can be used as part of an optimization objective, forming a multi-criteria optimization problem. During the construction process, acquired data can be used to update the associated probability distributions and therefore incrementally improve the prognosis and allowing to create plans taking even the most recently encountered circumstances into account.

1.2 Hypotheses

This thesis discusses five main hypotheses:

- Reactive Scheduling can reduce the negative consequences of unexpected delays of excavation and shoring operations.
- The uncertainty in construction operations can be modeled and handled by the presented methods.
- It is possible to acquire the required data (semi-)automatically from construction sites.
- Heuristic optimization approaches can be used to create schedules taking the acquired data into account and minimizing the changes due to further delays.
- The new pseudo-serial schedule generation scheme offers a good basis for metaheuristic algorithms to solve RCPSPs.

1.3 Methodological Approach

This thesis focuses on reactive scheduling for excavation and shoring operations, taking uncertainty into account. In order to acquire data from real-life construction sites, a new approach, based on photogrammetry was developed and extensively field-tested. Furthermore it has been combined with video-based methods developed in cooperation with the Georgia Institute of Technology in order to get more detailed performance data on the individual resources involved in the excavation processes. Additionally an extension of the resource constrained project scheduling problem has been defined, taking uncertainty into account. Furthermore a new heuristic optimization approach, capable of obtaining near optimal solutions for the extended problem, as well as the new pseudo-serial schedule generation scheme (PSSGS), was developed and extensively tested on the commonly used benchmark problem set PSPLIB (Kolisch & Sprecher, 1997).

1.4 Structure

This thesis is organized into nine Chapters. The first Chapter defines the problem and scope, presents the key hypotheses, as well as the methodological approach.

The second Chapter introduces the main elements of construction project models, the concept of construction simulation, including discrete-event simulation, different representations, as well as possible data sources. Additionally it discusses the issue of choosing the appropriate level of granularity and the associated trade-offs. Eventually the advantages and limitations of the state-of-the-art of construction simulation are discussed.

The third Chapter presents the state-of-the-art in construction schedule optimization and in that setting defines the resource constrained project scheduling problem (RCPSP), and describes why it is extended further in order to be applied to real-life scheduling problems. Then the distinction of active, semi-active, and non-delay schedules is discussed, followed by existing deterministic and heuristic optimization approaches, followed by the introduction of uncertainty and robustness into the problem. Then the concept of reasonable swaps and the Dori algorithm for determining float times are introduced. Eventually available benchmark problems are presented.

The fourth Chapter introduces four contributions of this thesis. It start with the novel pseudo-serial schedule generation scheme, followed by its use with the recently published water wave optimization algorithm by Zheng (2015). Then an improved version of a swap based local search algorithm, the iterative swap-based limited depth search is introduced. Eventually a extension of the PSPLIB scheduling problem library is presented, adding uncertainty to the individual problems. The presented methods are evaluated on commonly used benchmark problems. Finally the fusion of the acquired data with reactive simulation models is discussed.

The fifth Chapter provides an overview of the state-of-the-art in data acquisition on construction sites. Starting with the traditional protocol-based tracking, it describes the extraction and evaluation of machine sensor data, as well as different vision-based approaches.

The sixth Chapter introduces two new procedures for vision-based progress monitoring of excavation sites based on photogrammtery and video analysis. Eventually the two methods are combined into a coupled system.

The seventh Chapter describes the overall concept of reactive scheduling and how the presented methods interlock into a syngeric approach. This is illustrated by an example problem.

The eighth Chapter provides three case studies for the presented algorithms to show their practical applicability.

The ninth and last Chapter gives a summary and an outlook for future research potential.

1.5 Terminology

This Section briefly describes important terms used throughout this thesis.

Activity: A process that requires a certain number of resources and specific predecessor activities to be finished, in order to be executed. It either takes a predefined time to be finished, or the duration can be uncertain according to certain underlying probability distributions.

Discrete-Event Simulation (DES): A method for simulating systems, where the state of the system only changes at discrete points in time, where events happen, such as the completion of an activity.

Float Time: The amount of time an activity, or a set of activities, can be shifted, or prolonged within a schedule, without changing the remaining schedule and the overall makespan.

Makespan: The time difference between start and end time of a schedule.

Material: A non-renewable entity that can be used by an activity when executed, for instance, concrete, girders, or bricks.

Metaheuristic: An approximate algorithm that aims to efficiently and effectively explore a search space by the use of a combination of basic heuristic methods (Blum & Roli, 2003).

Monte-Carlo-Simulation: The act of performing multiple simulation runs of a system subject to uncertainty in order to obtain statistical measures (Metropolis & Ulam, 1949).

Performance Factor: A measure for the efficiency of a machine, worker, team, or a combination of those, for instance the amount of soil excavated by an excavator and its operator per day. Generally this thesis uses the definition of the amount of work performed over time.

Prerequisite / Predecessor activity: An activity that has to be finished for another activity to be executed.

Project: A collection of activities, required and available resources, and constraints that need to be satisfied for each activities execution.

Resource: A renewable entity that can be used by an activity when executed, but afterwards becomes available to other activities, for instance, machines, workforce, or geometric areas or volumes.

Schedule: A plan for the execution of a project, assigning start and end times, as well as resources to each activity. Metrics for further analysis can be extracted from a schedule.

2 State-of-the-Art in Modeling and Simulation of Construction Operations

"Simulation is the imitation of the operation of a real-world process or system over time. Simulation involves the generation of an artificial history of the system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented."
- Banks et al. (2001)

The planning process of large construction projects involves large numbers of activities which are subject to numerous constraints and require different resources and geometric spaces. The process of analyzing, planning, scheduling, and estimating costs, taking all factors into account is very tedious and error-prone when performed manually. Simulation methods are efficient tools to help the involved personnel to reduce errors.

In order to simulate a real system, it has to be represented by an abstract model, that serves as input for a simulation system. Several methods for modeling of construction operations have been published in literature, but since computing power has rapidly increased within the last decades, more and more new methods, as well as new data sources have become available, making construction simulation increasingly interesting for the architecture, engineering, and construction (AEC) industry. This Chapter gives an overview about the evolution of modeling techniques and their feasibility for construction operations as well as for whole construction projects. As costs and uncertainty play an important role in construction planning, methods for representing those factors are discussed. Another elementary question when modeling a system is the required granularity of the model and the desired quality of the simulation results. Also this aspect will be taken into account.

The structure of this Chapter is illustrated in Figure 2.1. On the basis of a real system, in this case, the construction sites in focus, simulation models are developed in order to resemble the processes that take place during the execution of the respective construction projects. The Chapter will introduce process models which describe the nature of repetitive processes in different ways. These models can be used to acquire metrics about different aspects of the processes. The different process modeling approaches are described in detail in Section 2.5. As an input metric for those process models the efficiency of the involved resources have to be known. The efficiency of the resources is described in terms of performance factors, which are discussed in Section 2.1. The performance factors form the link between the work to be performed and the time required to do so. Once the processes involved in a construction project are known, a model of the entire construction project can be built up using different representations. This process is discussed in Section 2.6. The process of simulation of the resulting model is described in Section 2.6.4. The results of the simulation can then be used to improve the execution plans, as described in detail in Chapters 3 and 4, and in that way feedback into the real system. Furthermore the performance factors can be dynamically updated using on-site monitoring systems, as described in Chapters 5 and 6.

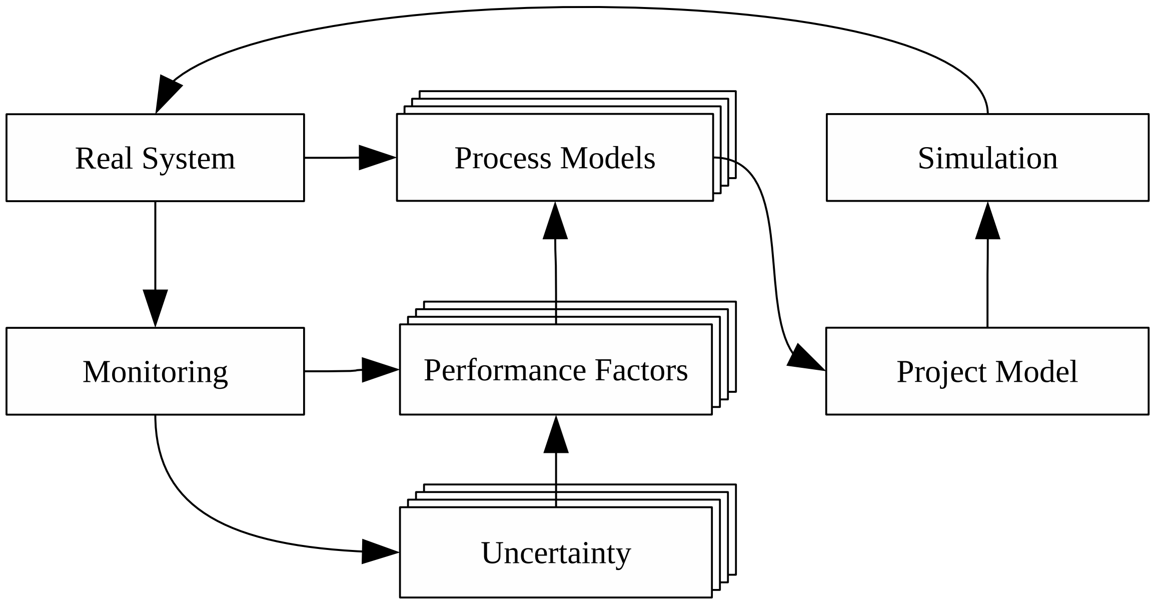


Figure 2.1: Flowchart of the reactive modeling and simulation cycle. The real system is modeled as a collection of process models, which take performance factors of involved resources into account. The performance factors are estimated and updated by monitoring the real system. The real system however is subject to uncertainty which in turn affects the resulting performance factors. The process models are combined into a project model (See Section 2.6), which is then simulated to generate schedules for the real system.

Once a model has been created it can be used to simulate the system using different simulation methods. For the case of construction simulation, this Chapter will focus on methods in the class of discrete-event simulation, which can be used to test different scenarios or hypotheses, as well as create prognoses for costs and duration of the project.

2.1 Performance Factors

In order to relate work that has to be performed to the time that is required to do so, a metric for the performance of resources is required. The performance of machines, workers and processes is commonly described by performance factors quantifying the amount of work performed over time. Pocock et al. (1996) also describes additional performance indicators such as cost performance, schedule performance, number of contract modifications and safety information. Atkinson (1999) referred to the factors time, cost, and quality as The Iron Triangle. For the scope of this thesis, the productivity of machines and workers is the only type of performance factors in focus, as it unites time and costs. In case of shoring projects the performance of excavation and shoring operations are vital (Park et al., 2005). The performance of excavation operations τ_e can be measured in volume excavated over time as described in Equation 2.1. Another example where such performance factors also play an important role is tunnel construction (Grima et al., 2000; Špačková & Straub, 2013; Yagiz et al., 2009).

$$\tau_e = \frac{\Delta v}{\Delta t} = \frac{v_{t_2} - v_{t_1}}{t_2 - t_1} \tag{2.1}$$

Navon (2005) followed a more general approach and analyzed the performance of construction projects based on four criteria. The positions of earth moving equipment and personnel was tracked. The movement of tower cranes was recorded and orders and deliveries of materials were tracked. Additionally the status of guard rails was monitored to ensure safety of the site. Generally performance was measured as the amount of work performed in a certain time. A downside of Navon's approach is that the measurements were very coarse and no absolute quantitative measurements of earthwork processes were taken. Navon (2007) presents an approach to obtain more detailed measures of excavation processes. The model still lacks absolute measurements of the excavated volume, such as those presented in Chapter 6. Furthermore performance factors are almost always subject to uncertainty (Baloi & Price, 2003; Salem & Zimmer, 2005; Szczesny et al., 2013).

2.2 Modeling Uncertainty in Construction Operations

In large scale construction projects even small deviations in the individual processes can cause problems in the synchronization of activities. Besides the performance factors also durations, idle times, delivery times or the occurrence of certain events, such as contract negotiations, are subject to uncertainty. Therefore the uncertainty of the individual processes must be addressed when modeling such large projects. An early approach for this problem, named Program Evaluation and Review Technique (PERT), was developed by the US Navy in order to forecast progress in weapons research and development programs (Malcolm et al., 1959). It uses a three-point estimation to model beta distributed process durations. PERT is addressed in detail in Section 3.3.4. Riley (2012) describes the shortcomings of this approach in a practical context and the need for improved approaches.

A more detailed representation of uncertainty can be realized through the generation of parameters for suitable probability distributions from real-life sample data. Such an approach was presented by Szczesny et al. (2013) and Szczesny & König (2015). Additionally the sample data does not have to be based on process durations, but instead can model the uncertainty in terms of the previously introduced performance factors, modeling the uncertainty in the performance of the involved resources. Brandt & Bode (2014) presented such an approach based on the use of the fuzzy sets possibility theory by Zadeh (1978) in the work on project scheduling by Masmoudi & Haït (2013).

The following subsections describe the four commonly used probability distributions, their characteristics, and their suitability for representing uncertain activity durations and performance factors.

2.2.1 Beta Distribution

MacCrimmon & Ryavec (1964) described that probability distributions used for construction operations should be continuous and limited between two extremes. Furthermore AbouRizk & Halpin (1992) showed that common earthmoving operations can be described by a beta distribution, which also satisfies the aforementioned criteria. AbouRizk et al. (1991) studied this matter deeply and developed the Visual Interactive Beta Estimation System (VIBES). VIBES uses a combination of four statistical characteristics of the process durations to be modeled in order to approximate a beta distribution for processes where no detailed sample data is available. Two of the input characteristics have to be the extreme points, the minimum and maximum durations that are possible to happen, while the other two, have to be two of the following characteristics: mode, mean, variance, or selected percentiles. VIBES then returns a parameter set for defining a beta distribution that approximates the provided data. These parameters are the beta distribution's shape parameters α and β, and the extreme values. If detailed sample data is available, the parameters can also be estimated using other methods, such as Maximum Likelihood Estimates (MLE) (AbouRizk et al., 1994). The definition of the probability density function is given in Equation 2.2. The mean of the distribution is defined in terms of α and β in Equation 2.3.

$$f(x \mid \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du} \tag{2.2}$$

$$\mu = \frac{\alpha}{\alpha+\beta} \tag{2.3}$$

As can be seen in Figure 2.2, manipulating the shape parameters causes the shape to change in different ways. For all cases an example process with a minimum duration of 2 minutes and a maximum of 6 minutes was chosen. Increasing the α value causes the mean to shift towards the maximum possible duration, while increasing the β value causes the mean to shift towards the minimum possible duration. Fente et al. (1999) showed that those properties of the beta-distribution are ideal for modeling the uncertain durations of construction operations, and due to the natural limitations in such processes both parameters must be positive and larger than one.

2.2.2 Normal Distribution

Golenko-Ginzburg & Gonik (1997) additionally consider normal and uniform random variables as activity durations. As the normal distribution is not limited between two

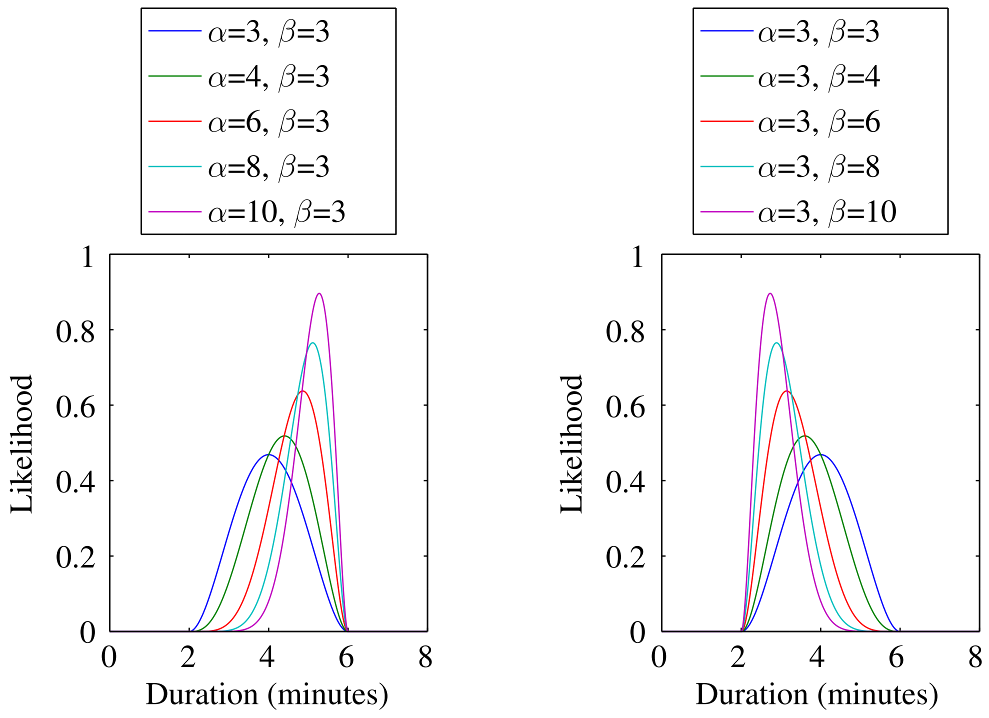


Figure 2.2: Shape change of beta distribution probability density function when manipulating the shape parameters α (left) and β (right).

extremes it needs to be truncated in order to prevent the possibility of too high, too low, or even negative activity durations. Figure 2.3 shows the effect of the normal distribution's parameters mean μ and standard deviation σ on the probability density function given in Equation 2.4, when being truncated. Same as in the previous example, the Figure shows activity durations limited between two and six minutes, while shifting the mean or changing the standard deviation σ. With a sufficiently large value for σ, the results will get closer to a uniform distribution where each value with in the bounds is as likely as any other. In order to estimate the parameters σ and μ of the normal distribution, Equations 2.5 and 2.6 can be used.

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.4}$$

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{2.5}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{2.6}$$

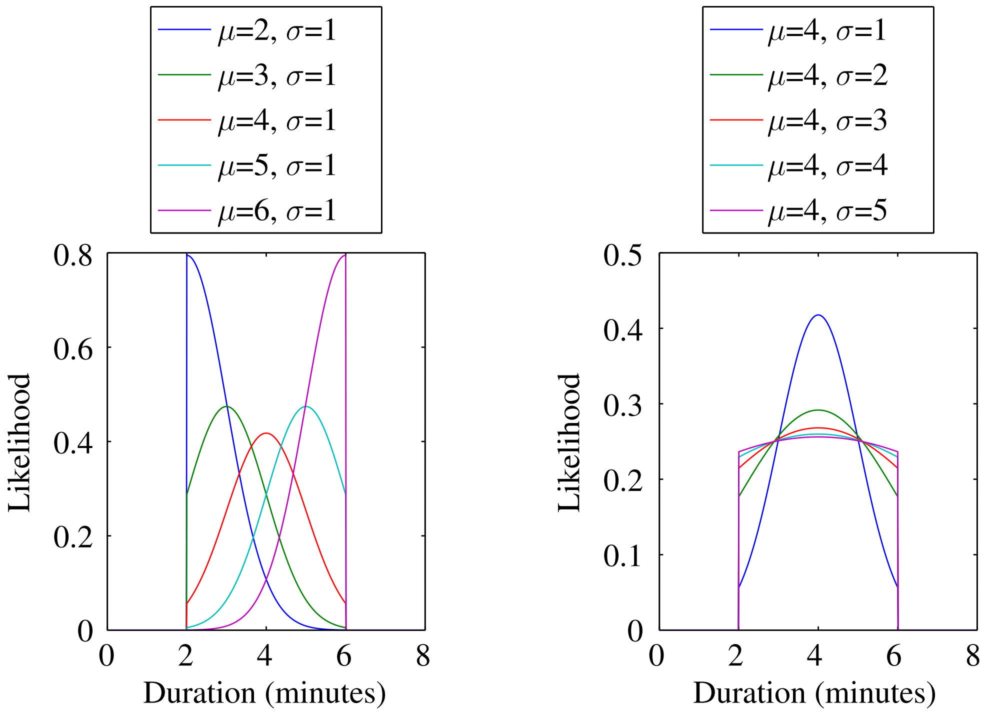


Figure 2.3: Change of truncated normal distribution probability density function when manipulating the parameters μ (left) and σ (right).

2.2.3 Gamma Distribution

Klein & Baris (1991) propose the use the Gamma and Weibull distributions in order to model service times, which is a very similar use case than construction activity durations. The exponential and the χ^2 distributions are special cases of the gamma distribution and therefore do not need special treatment. The gamma distribution is defined by the probability density function defined in Equation 2.7 and uses the gamma function as defined in Equation 2.8. The gamma distribution is only defined for positive values of x, but there is no upper bound. Therefore it is truncated in order to represent an upper limit on the activity duration. The shape parameter k determines the skewness of the distribution, while the scale parameter θ scales the distribution along the x parameter. This behavior is illustrated in Figure 2.4. The parameters of the gamma distribution can be estimated by making use of the fact that the mean μ is defined as the product of the two paramters k and θ as is shown in Equation 2.9.

$$f(x \mid k,\theta) = \begin{cases} \frac{x^{k-1}e^{-\frac{x}{\theta}}}{\theta^k\Gamma(k)} & x \geq 0 \text{ and } k,\theta > 0, \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Where:

$$\Gamma(t) = \int_0^\infty x^{t-1}e^{-x}\,dx \tag{2.8}$$

$$\mu = k\theta \tag{2.9}$$

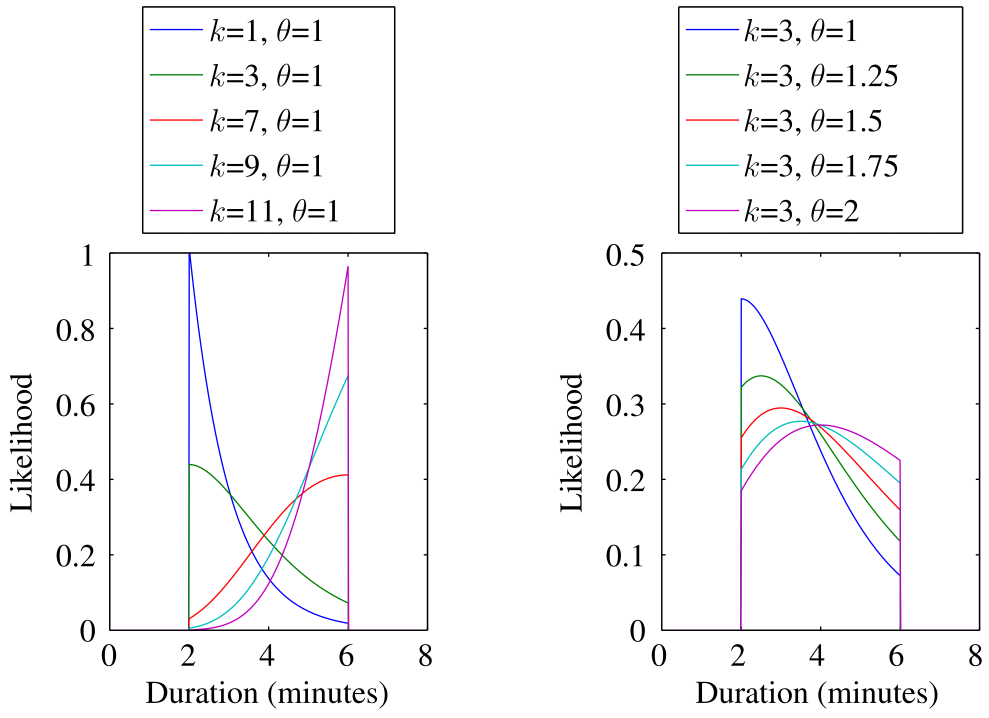


Figure 2.4: Change of truncated gamma distribution probability density function when manipulating the parameters k (left) and θ (right).

2.2.4 Weibull Distribution

Similar to the gamma distribution the Weibull distribution (Weibull, 1951) is defined by a shape parameter k and a scale parameter λ. Both parameters a positive real numbers. A value for $k < 1$ indicates that the probability decreases over time, while $k = 1$ implies that the probability of remains constant, and $k > 1$ implies that the probability increases over time. In terms of construction activities this means, that an activity is very likely to be finished in minimum time, is likely to be finished within a certain time window, or is likely to take close to the maximum time, but possibly finish earlier. The mean of the Weibull distribution is defined in terms of λ and k in Equation 2.11 and makes use the gamma function defined in Equation 2.8.

$$f(x \mid \lambda, k) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k} & x \geq 0, \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

$$\mu = \lambda\Gamma(1 + \frac{1}{k}) \tag{2.11}$$

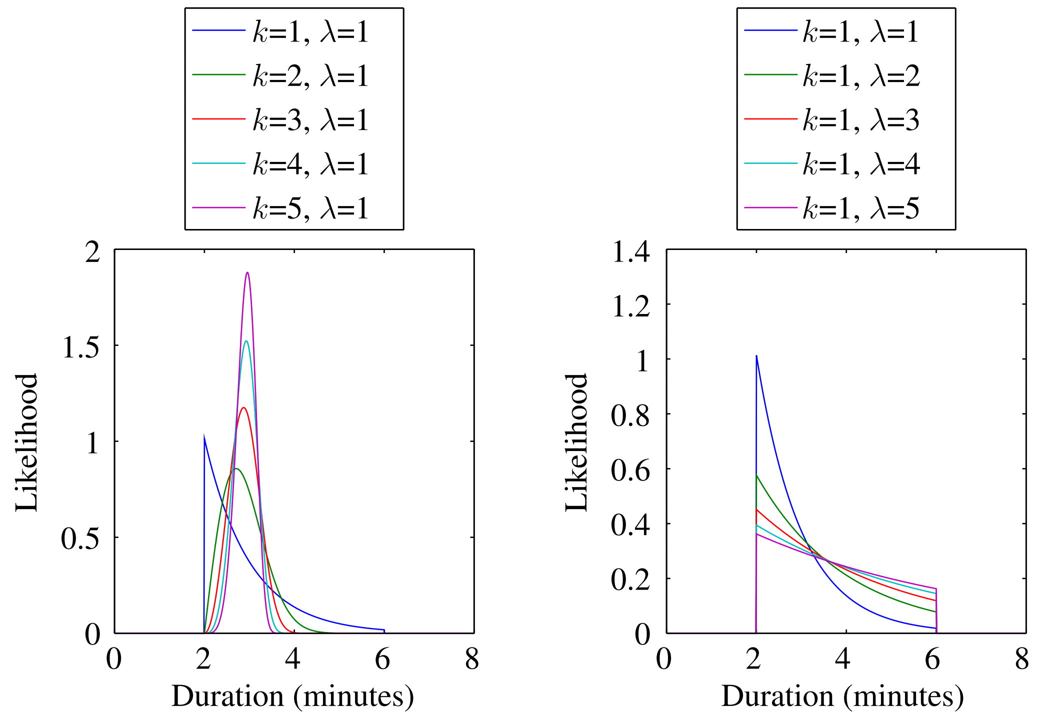


Figure 2.5: Change of truncated Weibull distribution probability density function when manipulating the parameters k (left) and λ (right).

2.2.5 Goodness-of-fit Tests

If detailed sample data is available for an activity or a class of activities, goodness-of-fit tests can be used in order to find the best matching probability distribution. If only limited data is available, methods such as VIBES can be used to estimate a beta distribution, as has been proposed by AbouRizk et al. (1991). Most commonly, either the χ^2-Test (Plackett, 1983; Yates, 1934) or the Kolmogorov-Smirnov-Test (Smirnov, 1948) are used in order to verify whether a distribution fits the given data. For illustration purposes sample data has been generated for each of the presented probability distributions. Figure 2.6 shows $10,000$ samples of each of the distributions.

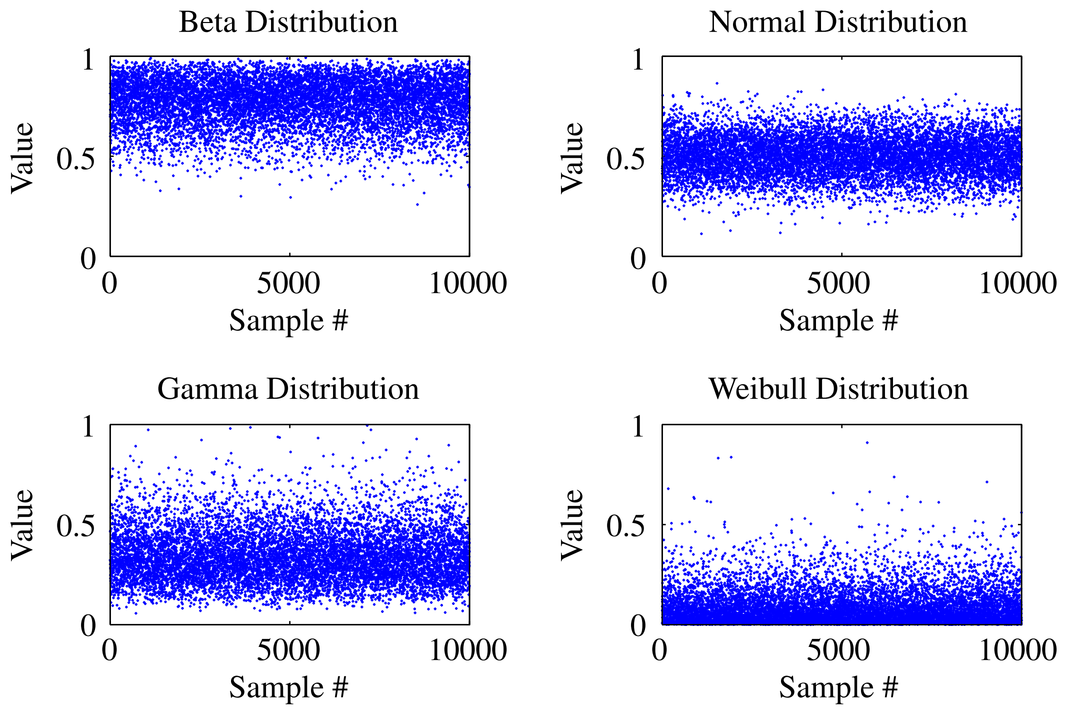


Figure 2.6: Sample data of four probability distributions. Beta distribution α=10, β=3 (Top left), Normal distribution μ=0.5, σ=0.1 (Top right), Gamma distribution k=7, θ=1 (Bottom left), Weibull distribution k=5, λ=1 (Bottom right). The plots show $10,000$ samples per distribution.

In order to show which distribution matches the given data, all samples were normalized to the range of 0 to 1, such that all distributions are feasible candidates for all sample sets. Then the χ^2-Test, as well as the Kolmogorov-Smirnov-Test have been applied to test every sample set against every candidate distribution. Each test returns whether the hypothesis, that the data stems from the tested distribution, is accepted or rejected, based on a p-value, which is the probability that the data, or more extreme (unlikely) versions of it, are samples of the tested distribution. The parameters of the distributions, the data was sampled from, are considered unknown and therefore not used for the tests. Both tests require a parameter α^* which is the probability of accepting a false hypothesis.

Tables 2.1 and 2.2 give the p-values resulting from the performed tests. Each column is labeled with the distribution the data is tested for, and each row is labeled with the

Hypothesis / Actual	Beta	Normal	Gamma	Weibull
Beta	0.6617	0.0000	0.0000	0.0000
Normal	0.0000	0.9970	0.0000	0.0000
Gamma	0.0000	0.0000	0.5964	0.0000
Weibull	0.0000	0.0000	0.4979	0.4977

Table 2.1: p-values of χ^2 Goodness-of-fit Test.

Hypothesis / Actual	Beta	Normal	Gamma	Weibull
Beta	0.9387	0.0000	0.0000	0.0014
Normal	0.3531	0.8613	0.0000	0.0000
Gamma	0.0000	0.0000	0.9720	0.0000
Weibull	0.0061	0.0000	0.9506	0.9632

Table 2.2: p-values of Kolmogorov-Smirnov goodness-of-fit test.

distribution the data has actually been sampled from. As can be seen, the beta, normal and gamma distributed samples can be easily identified with very high p-values. The Weibull data on the other hand, could as well be sampled from a special case of the gamma distribution as a comparison of Figures 2.4 and 2.5 suggests. While, the chi square test fails to attribute the data to either distribution, the Kolmogorov-Smirnov-Test returns positive results for both cases, but gives a higher p-value for the true case. All tests were performed using an α^* value of 0.05.

This shows how goodness-of-fit tests can be used to identify a well fitting probability distribution to describe sample data. The resulting probability distribution parameters form a vital part of a simulation model including uncertainty.

2.2.6 Monte-Carlo-Simulation

Given a problem with an associated solution, based on complex interaction of uncertain values, it can be difficult to obtain metrics on the solutions that are otherwise likely to happen. Obtaining a single outcome can yield a non-representative result. Monte-Carlo-Simulation describes a method that is based on obtaining a large number of outcomes while sampling new values for the uncertain values for each run (Metropolis & Ulam, 1949). This sampling is performed based on the underlying probablity distributions. Eventually statistical data is calculated from the collected data. This includes for instance the mean, standard deviation, and worst or best case solutions. Kroese et al. (2014) describes Monte-Carlo-Simulation as an important tool to gain insight into randomness.

2.2.7 Conclusion

The last Sections provided an overview of the four most commonly used probability distributions. Literature in the scheduling domain most commonly proposes the use of the beta distribution for representing uncertainty in process durations. As the beta distribution is bounded, sampled values cannot exceed a minimum and a maximum activity duration. Furthermore the parameters can be tuned to represent activities that are either more likely to finish early or late. As the performance factors directly correlate with activity durations, this representation is considered equally good for this purpose. Due to its high suitability, uncertainty is represented using the beta distribution throughout this thesis. Especially the extension of the Project Scheduling Problem Library PSPLIB, the UPSPLIB, which is presented in Section 4.5, is built upon the beta distribution. Furthermore the Monte-Carlo-Simulation method forms an important tool for dealing with those uncertain problems.

2.3 Modeling Costs of Construction Operations

Besides performance, uncertainty, and robustness, the costs are a main criterion for the success of a project and therefore form another important factor when modeling construction projects. Almost every activity that is performed on a construction site creates costs. Those costs can in part be modeled as attributes of the respective activities . On the other hand there are many fixed costs, such as working crews having to be paid, independent of their actual work load. However, not all costs are directly associated to construction activities. There are milestones that have to be met and if missed, cause fines. This can be solved by introducing milestones as functional elements into the model. When they are executed prior to their deadline, they create no costs, but afterwards, a fine is added. Missing a milestone can additionally create running costs, such as a part of a building that cannot be used yet, and therefore there is no rent being paid.

Furthermore additional resources, such as more machines or workers may be deployed on the site, in order to speed up the process. Hiring new machines might involve one time costs for transportation and running costs for rent, fuel and expendable materials. When a resource is not required for a certain amount of time, it may be worthwhile to free it to be used on another site, to save costs. For short periods of time on the other hand, the transportation cost may exceed the savings.

Different kinds of materials may lead to different production times of building elements, such as different types of concrete being used. In that way, there is a correlation between time and costs, creating a trade-off problem as described by Li & Love (1997) and Feng et al. (1997). Adeli & Karim (1997) makes use of this correlation in their neural dynamics model, by finding a global optimal schedule and subsequently minimizing the costs for this schedule. Bogenstätter (2000) even included the costs for the entire life-cycle in the early design phase. All these considerations form a strong motivation to include costs into the modeling process of construction projects.

2.4 Granularity

While it is theoretically possible to implement a continuous simulation model, describing the precise movement of each machine, personnel, and materials, it is hardly feasible on the large scale. Therefore the granularity (Level of detail) of the model is an important decision to be made. This decision forms a trade-off problem between the effort to create and maintain the model, the effort to work with the model in terms of computational power and the precision of the results. In order to model construction projects, the granularity is mainly determined by the aggregation of individual activities.

For instance a bored pile wall can be modeled by the individual activities required to produce a single bored pile, the production of a single bored pile can be modeled as one activity requiring all involved resources, or the production of the entire bored pile wall can be modeled as one aggregated activity. The main downside of aggregating activities is the coarser representation of the usage of required resources. While the drilling rig is only required for the activity of drilling the hole, it would be blocked for the entirety of the bored pile activity when aggregated. Same holds for blocked areas on the construction site. In practice though, moving resources between different areas on a site, during a dedicated activity is rarely effective, and not commonly done in practice.

Brooks & Tobias (1996) defined criteria for selecting the best model to represent a given problem. They define four main criteria: Results, future use of the model, verification and validation, and required resources. The results should be accurate and adequately describe the behavior of interest. If it is in focus of the user to determine the optimal order of the bored pile production, then modeling each involved activity individually can be helpful as it for instance allows exploitation of geometric features to speed up the progress. When an entire large scale construction project is modeled, the possible time savings of an optimized bored pile wall production might be negligible, while the model might have an unfeasible size without aggregation of activities. When there is no major interaction between the bored pile wall production and other concurrent activities, the bored pile wall production might be modeled and optimized individually and then be included in the model of the entire project as one aggregated activity.

As most construction sites are unique, future use of models is rarely of interest. On the other hand many packages of activities are used on many occasions, and could therefore be formed into (parametric) modules or process patterns (Sigalov & König, 2015) that can be reused on other sites. Such a method was presented by Wu et al. (2010) in the domain of bridge construction and in a more general context by Benevolenskiy et al. (2012). It would therefore be an option to for instance create a bored pile wall module, that can be parameterized in order to be used with different depth, thickness, wall length, or drilling rigs. Furthermore Wu et al. (2010) used the notion of Levels-of-Detail (LoD), that allows to get an model at a desired degree of abstraction. Verification and validation of the simulation results can be either be done using historical data, or by tracking excavation sites to measure how successful the plans, based on the model, can be put into practice.

The required resources are computational power, and time required by the planners to build the model . A modular approach can simplify the process drastically, but might require more fine tuning. Creating a coarse model and refining critical activities has been shown to be an effective approach (Panchal et al., 2008). Schruben & Yücesan (1993) has shown that the complexity of simulation models can be determined using graph theoretic approaches. It was shown that several graph theoretic measures can be used on the event graph to measure the complexity of a model (Brooks & Tobias, 1996). Furthermore, the characteristics of the resulting model can be formally analyzed as described in Section 3.4.2.

2.5 Modeling and Simulation of Construction Processes

There are copious modeling approaches in literature, each having different notations, different scopes and different advantages. The following Sections introduce different approaches that have been published throughout the history of computational modeling and simulation. The capabilities for representing construction projects are clarified using an illustrative example of a system producing a bored pile wall.

2.5.1 The Bored Pile Wall Example

In order to illustrate the difference between the presented modeling approaches a simple example of a common procedure in shoring operations, the production of a bored pile wall has been chosen. A bored pile wall is commonly produced by repeating five sequential steps. First the drilling rig moves to the position where the next pile should be drilled. Then the drill is aligned to be exactly horizontal before the drilling process starts. Then the hole is drilled. After drilling, in most cases a cage of reinforcement steel is lowered into the hole, which is eventually filled with concrete. Before the hole can be filled with concrete the drilling rig needs to move away from the hole. For simplification, the reinforcement step is not included in this example. The example procedure is depicted in Figure 2.7. It involves two machines, the drilling rig and the concrete truck. The area around the hole to be drilled can be blocked by either of the machines. In large construction projects the production of bored pile walls can be parallelized using multiple machines, implying that the order of the piles to be drilled is not fixed. For overlapping bored pile walls, a partial drilling order is given. Also drilling next to a freshly drilled pile is discouraged, as the wet concrete would pour over. For simplification this is not taken into account in this example.

2.5.2 General Purpose Systems Simulation (GPSS)

IBM described the first general purpose systems simulation (GPSS) for computation purposes in the early 1960s (Gordon, 1961). The simulation models are created using a graphical modeling language. It consists of 26 different block types that can be used to generate block diagrams describing the system to be studied. There are entering and removing blocks, such as the "Call" and "Terminate" blocks. The "Call" block enters transactions from a list of available transactions into the simulation model at predefined time intervals or whenever the transaction is accepted by the following block. When a

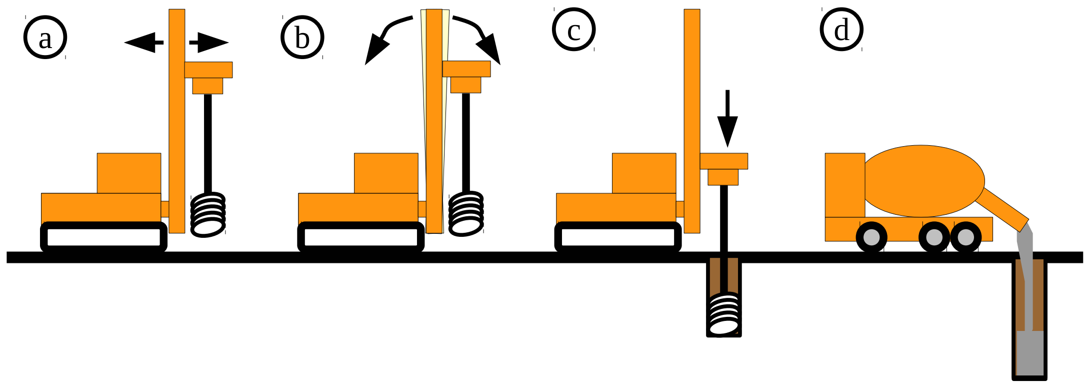


Figure 2.7: Bored pile wall example illustration. The drilling rig is positioned above the drill location (a), then the rig adjusts to be exactly upright (b), the hole is drilled (c), and then filled with concrete (d). This illustration was inspired by https://de.wikipedia.org/wiki/Datei:Foundation_pile_scheme.svg

transactions enters a terminate block it is removed from the simulation. The "Advance" block represents a simple activity that is executed whenever a transactions enters into it and takes a certain amount of time. A "Gate" block either accepts or blocks an incoming transaction depending on the availability of a resource. A loop with a preceding dummy "Advance" block creates a loop waiting until a resource becomes available. The number on the right side of a "Gate" block selects the resource to be queried. A "Seize" block describes the action of seizing a resource until a "Release" block releases it again. The number in the right side triangle names the resource to be seized or released. A "Hold" block is similar to an "Advance" block, but during execution holds on to the resource named in the right side double triangle. Each block in the diagram is uniquely labeled by a number.

The bored pile wall example is illustrated as a GPSS block diagram in Figure 2.8. It consists of 15 blocks. The first block, a "Call" block, contains a list of piles to be produced. Then a "Gate" block causes the system to wait for the drilling rig to be ready, then seizing the drilling rig in the "Seize" block 3. The "Advance" and "Gate" blocks 4 and 5 wait for the area around the location to be drilled to be free, the "Seize" block 6, then seizes the location, after which the drilling rig moves to the location using the "Advance" block 7. Block 8 adjusts the drilling rig and block 9 drills the hole. After that block 10 releases the rig and blocks 11 and 12 wait for the cement truck. Then the "Hold" block pours concrete into the hole, while holding the cement truck resource. Finally the location is released in block 14 and the pile is terminated in block 15.

This way of describing the example delivers a good representation of the system. As soon as the drilling rig is released in block 10, the "Call" block 1 can enter the next pile into the simulation and it can be drilled while the previous hole is still being filled with concrete. Hence it allows for parallel execution of the involved processes. Since GPSS is a general purpose model it has a very high level of abstraction making it difficult to create valid representations for non-experts. Furthermore it becomes much more complex when more flexibility is required, such as multiple drilling rigs being available.

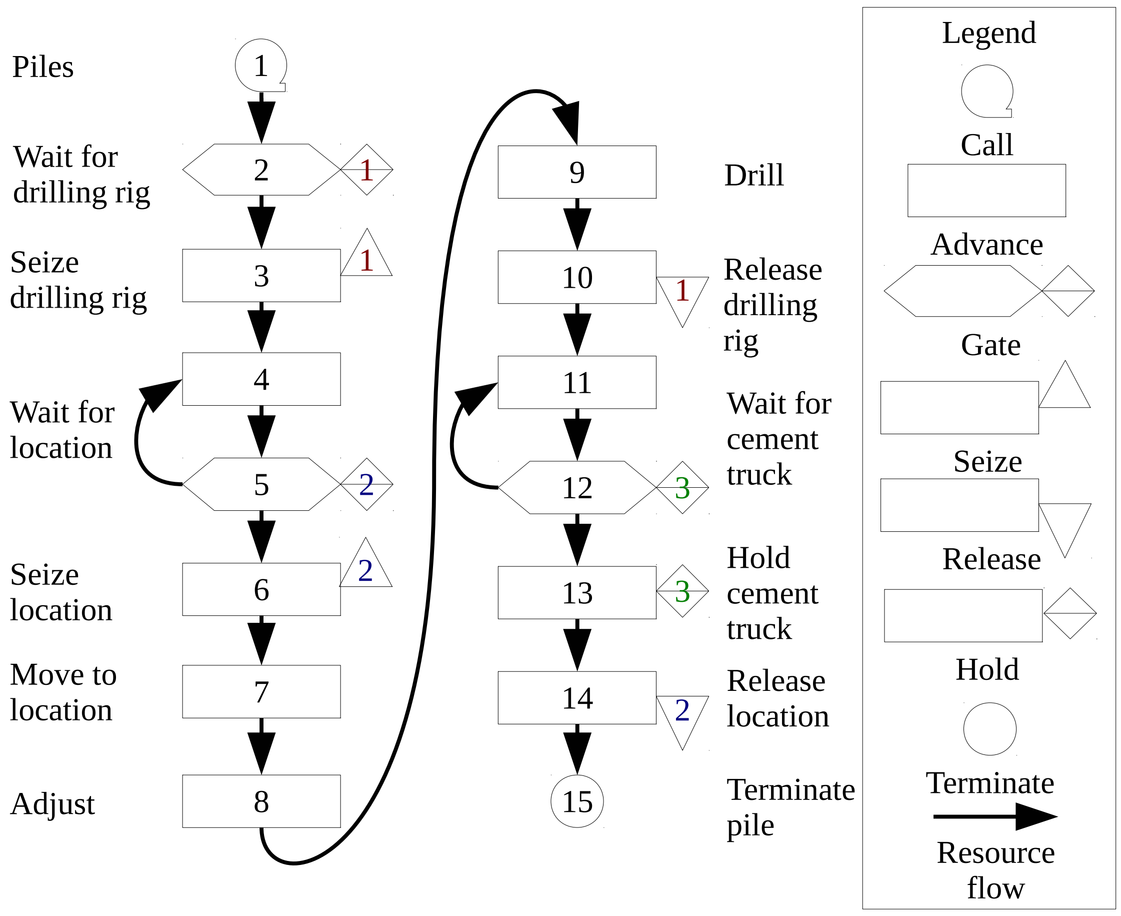


Figure 2.8: Bored pile wall example as a block diagram in the notation for the General Purpose Systems Simulation (GPSS). The call element on the top left outputs all the piles that need to be produced one by one. following the arrows, the pile is then produced and is finished once the pile instance reaches node 15, which terminates the production of that pile.

Figure 2.9 shows the GPSS model extended by an additional drilling rig. For this purpose 3 more block types are used. The "Branch" block allows to send resources along either of the outgoing arrows. It allows statistical selection among the possible choices, or using the first outgoing arrow that is not blocked. The "Tag" block allows to tag a resource with a number, and the "Transfer" block redirects the resource to an output according to a previously assigned tag. The updated diagram adds a "Branch" block number 2, that selects either drilling rig 1 or 2, then seizing the respective rig in blocks 4 or 7. Furthermore the current pile is tagged with the assigned drilling rig. When the drilling rig is to be released the assigned tag is used in the "Transfer" block 15, to release the respective drilling rig. As can be seen in this example, adding additional resources to an existing diagram is troublesome and requires tedious work. Adding multiple cement trucks would add even further complexity. When multiple resources, with multiple instances are to be used at the same time, the concept of tagging resources quickly becomes very confusing and error prone. Understanding such a model as a non-expert can be very difficult. For

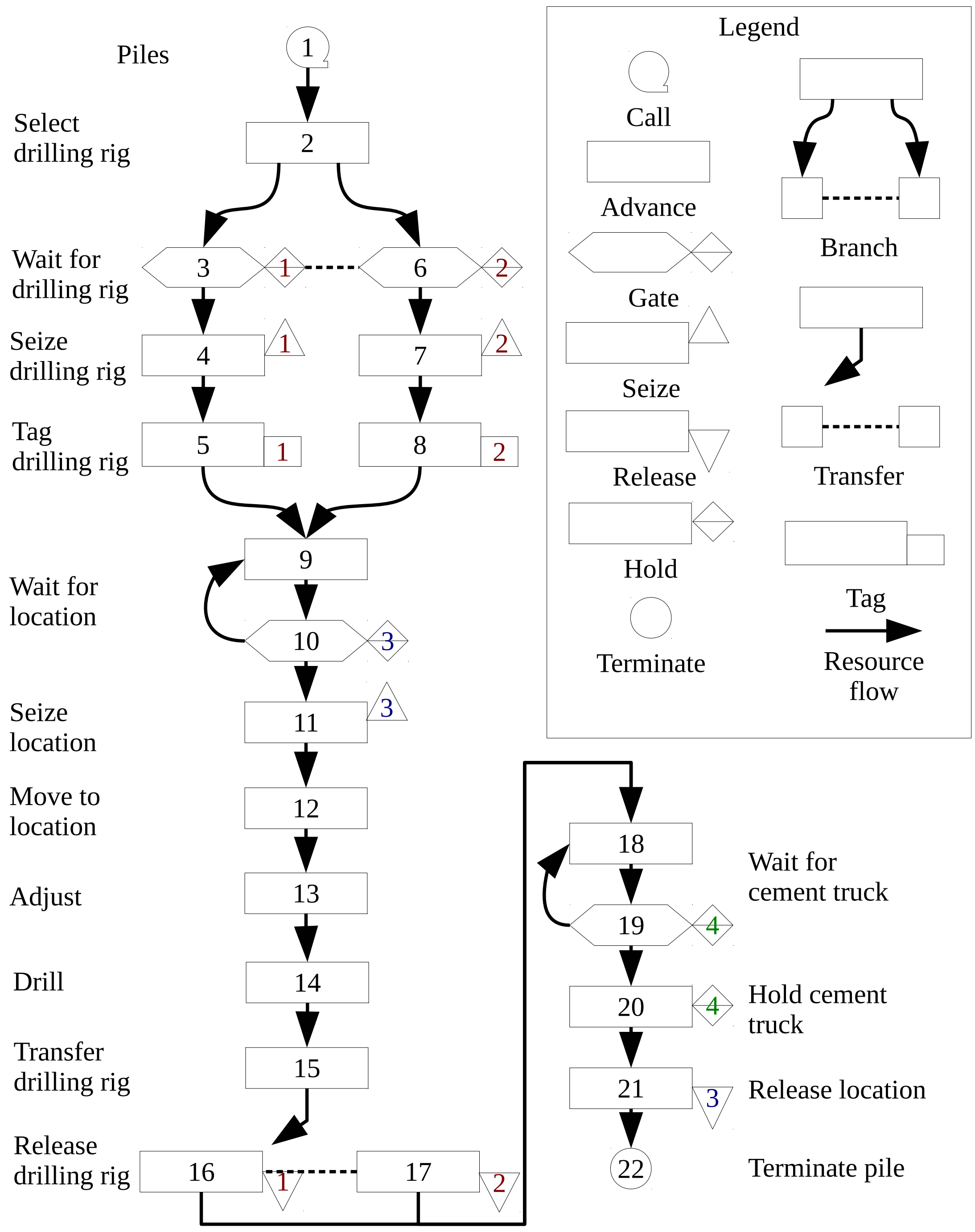


Figure 2.9: Bored Pile Wall Example as a block diagram in the notation for the General Purpose Systems Simulation (GPSS) with two drilling rigs. The call element on top outputs each pile that needs to be produced one by one. In node 2 a drilling rig is selected for each pile, after which the respective rig is used for production, creating the two parallel streams for seizure of the rig. Once the drilling is seized, the streams merge again, producing the pile, and eventually releasing the seized rig. Eventually each pile's production terminates in node 22.

this reason more specialized models were developed.

2.5.3 Activity Cycle Diagram (ACD)

The Activity Cycle Diagram (ACD) was introduced by Tocher (1963) to model interactions of system objects. It consists of two states, idle and active, that can be used to describe simulation entities. different arrows among the entities describe the state changes of resources. Hence, the arrows indicate the flow of the resources between active and idle states and therefore indicating the assignment of resources to the individual processes. In order for a state to become active, all incoming idle states must be holding the required resources.

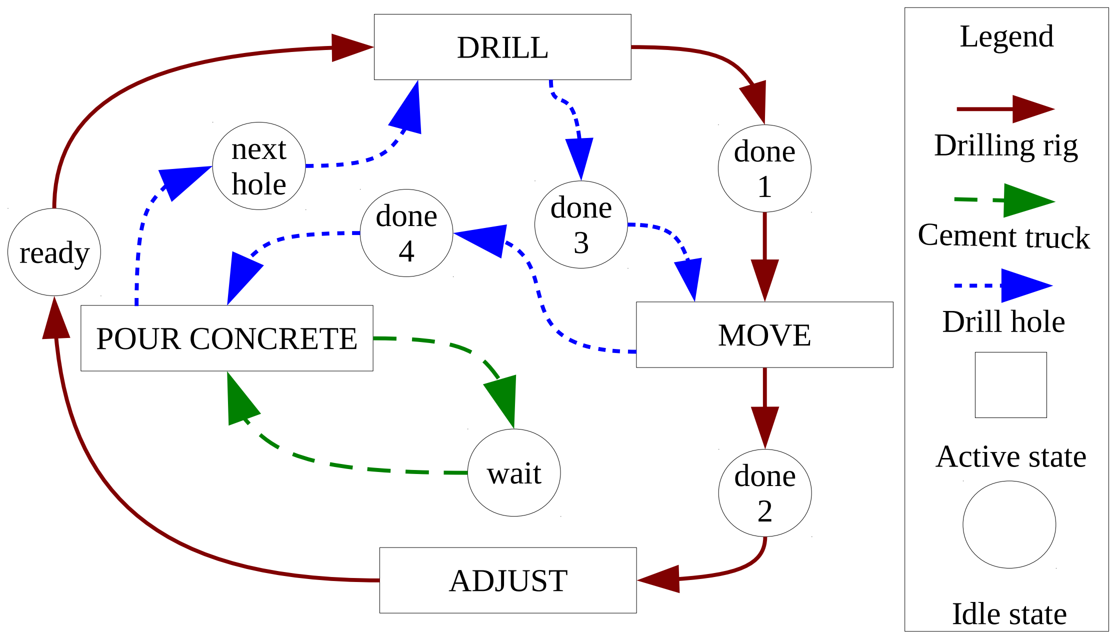


Figure 2.10: Bored pile wall example as an Activity Cycle Diagram (ACD). This diagram shows the flow of the resourced as differently colored arrows. First a hole is selected in the "next hole" idle state, going into the "DRILL" active state, requiring the drilling rig to be in the idle state "ready". Once drilling is done, the drilling rig can move to the idle state "done 1", while the drilled hole can move to the idle state "done 3". Then the drilling rig enters the "MOVE" active state, freeing the drilled hole, which goes into the "done 4" state. The drilling rig moves into the "done 2" state, being ready to be adjusted for the next hole. The drilled hole, can now be filled with concrete, hence enter the "POUR CONCRETE" active state, given the cement truck is not currently filling another hole, hence is in the "wait" state. Once the hole is filled, the hole resource can proceed to the next hole.

For the bored pile wall example, depicted in Figure 2.10, three types of arrows were introduced, forming three individual cycles for each of the involved resources, the drilling rig cycle using red solid arrows, the cement truck cycle using green dashed arrows, and the drill hole cycle using blue dotted arrows. The drilling rig moves along the cycle of the active states "MOVE", "ADJUST", "DRILL". The cement truck moves along the cycle "MOVE", "DRILL", and the drill hole cycles between the active states "DRILL", "MOVE", and "POUR CONCRETE". When the drilling rig is in the idle state "done 1" following the active state "DRILL", it can commence to move to the next drilling position. After having moved it continues to the idle state "done 2", after which it directly goes into the active state "ADJUST". After that it is ready for drilling the next hole. Before the system can enter the active state "DRILL", the idle state "next hole" needs to hold the "Drill hole" resource, which is only freed after the concrete has been poured. Once the "DRILL" state has been finished, both the drilling rig and the drill hole resource are freed and move to the respective idle states "done 1" and "done 3". After that the active state "MOVE" can be entered, for the drilling rig to move on to the next location. As the cement truck must be in the idle state "wait" and the drill hole resource moved to the idle state "done 4" after the drilling rig moved, the active state "POUR CONCRETE" can now be entered. Once the concrete has been poured, the drill hole resource moves to the "next hole" state and the cement truck moves into the "wait" state. At this point the cycle can start over.

As can be noticed this diagram forces the pouring of the concrete to be finished prior to the next hole to be drilled, which would practically be possible in parallel to the next drilling process. This is due to the artificial drill hole resource, which is required to synchronize the concrete pouring cycle and the drilling rig cycle. This shows that the ACD notation is only feasible for strictly cyclical problems, such as excavation activities, but in this case does not allow sufficient synchronization of the concrete and drilling cycle.

2.5.4 CYCLic Operations NEtwork (CYCLONE)

The first modeling approach that has been widely applied to construction processes was published by Halpin (1977) and is called CYCLONE (CYCLic Operations NEtwork). Compared to the previously described Activity Cycle Diagram (ACD), CYCLONE introduces several new elements that allow the modeling of more complex systems. Compared to the normal active state, CYCLONE distinguishes between normal and combi elements, as well as queue and function elements. Furthermore a formal counter element is introduced for statistical evaluation purposes. A normal state has a single incoming arrow from an arbitrary other element, that when handing over a resource immediately allows entering that state. A combi state, on the other hand can have multiple incoming arrows, that all need to hand over a resource in order for the state to be entered. A queue element collects incoming resources and supplies them through the outgoing arrows. A function element spawns new resources to be supplied through outgoing arrows.

The bored pile wall example as a CYCLONE diagram is depicted in Figure 2.11. Beginning in the state "MOVE", the drill rig can immediately enter the state "ADJUST" after moving to the appropriate position. After adjusting, it will drill a new hole, which is supplied through the function element "holes 1". After the drilling process is completed, the drilling rig can move on to the next state, hence the state "MOVE" becomes active. At the same time the drilled hole is handed over to the queue element "holes 2". Given that the concrete truck is being held in the queue element "truck", the "POUR CONCRETE" combi state can be activated. Once the concrete has been poured, the hole is handed to the

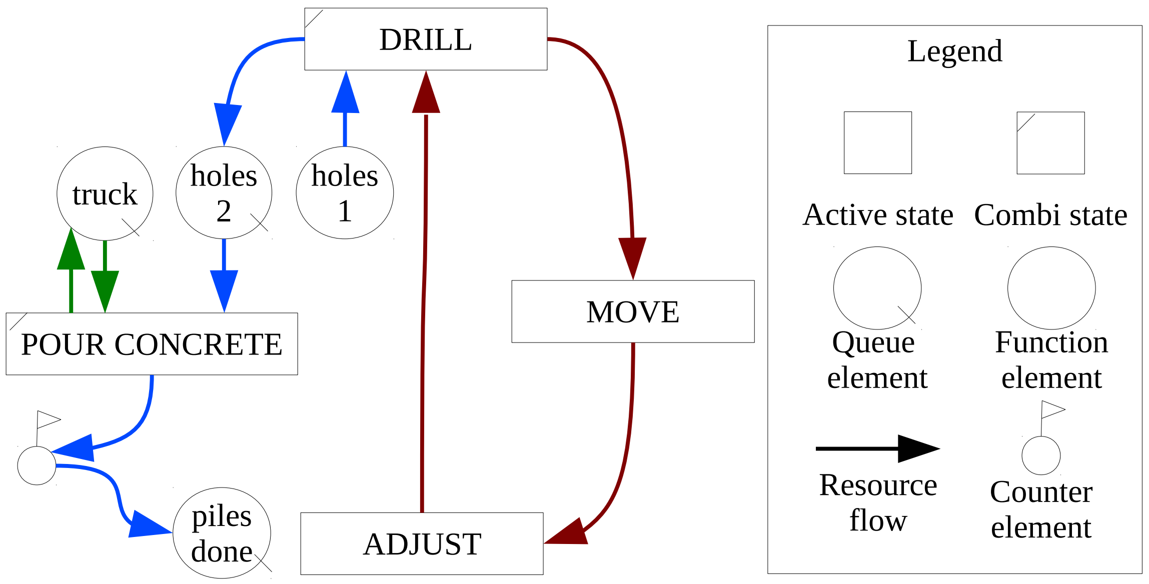


Figure 2.11: Bored pile wall example as a CYCLONE diagram. The colors of the arrows indicate the resource cycle, same as in Figure 2.10. The function element "holes 1" outputs the piles that need to be produced. the combi-state "DRILL" is then executed given the drilling rig is available. Once the drilling is finished the drilling rig moves away in the "MOVE" active state and adjusts in the "ADJUST" active state. The hole goes into the "holes 2" queue element, where the holes queue up, waiting for the cement truck. The cement truck moves to the "truck" queue element when finished and then becomes available for the combi-state "POUR CONCRETE". Once a hole is filled with concrete, it moves to a counter element and proceeds to the "piles done" queue element.

counter element, counting the number of finished piles, then it ends in the queue element "piles done" and the truck is fed back to the queue element "truck". At the same time the drilling rig can have drilled new holes and can be filling up the queue element "holes 2". Due to the queuing and function element functionality of CYCLONE, the bored pile wall example cycle can be adequately modeled in this diagram, while the drilling and concrete cycles being synchronized through the "holes 2" queuing element. If the drilling cycle advances faster than the concreting cycle, the "holes 2" queuing element will collect drilled holes, ready to be filled with concrete, while in the opposite case, the concrete cycle will idle until a new hole is ready to be filled. CYCLONE is still used frequently due to its simplicity, while still being capable of representing complex scenarios (Huang & Hsieh, 2012). A system called PROject SImulation Dragados Y Construcciones (PROSIDYC), which is based on CYCLONE has found its way into real life use, where it was able to facilitate significant improvements in the project execution (Halpin & Martinez, 1999).

2.5.5 STate and ResOurce Based Simulation of COnstruction ProcEsses (STROBOSCOPE)

A decision, project managers are facing regularly is the selection of resources for different processes. A machine might be able to perform an operation more efficiently, but at the same time create more costs, or as a downside block a larger area on the construction site. For this purpose CYCLONE was extended by a forking element, resulting in the STate and ResOurce Based Simulation of COnstruction ProcEsses (STROBOSCOPE) (Martinez & Ioannou, 1994). This extension allows multiple different ways of reaching a certain goal. The fork element allows the resource flow to leave through any one of the outgoing arrows. This can mean different orders of executing sub-processes or using different resources for a process, resulting in different performance factors.

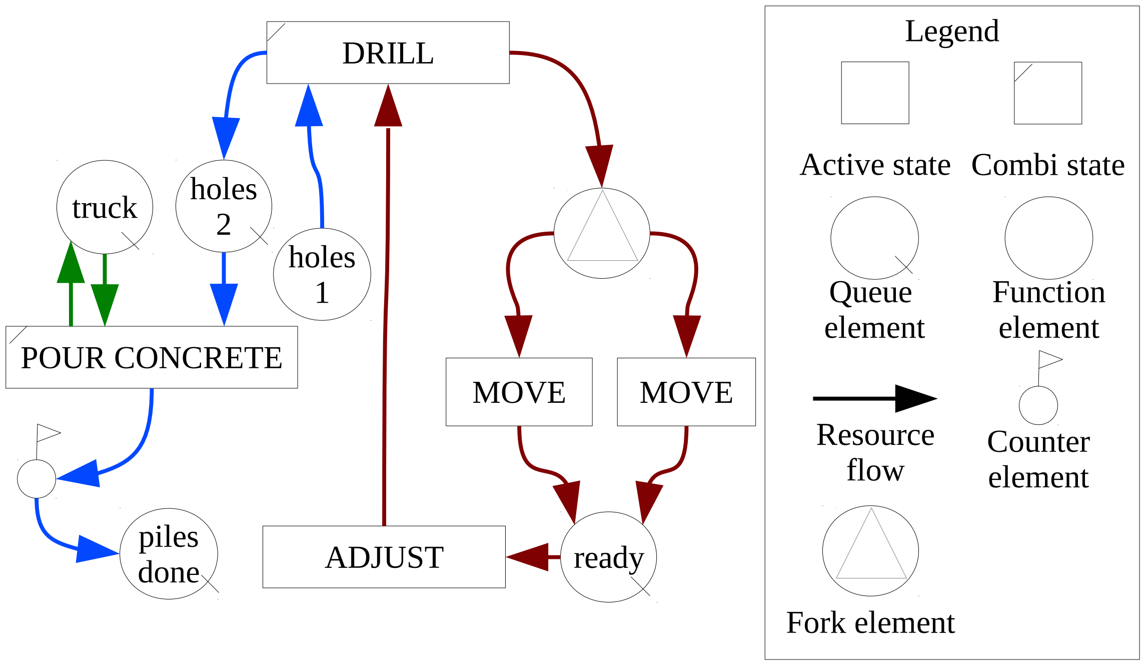


Figure 2.12: Bored pile wall example as a multi-mode STROBOSCOPE process diagram. Compared to the CYCLONE diagram in Figure 2.11, the diagram was extended by the fork element, offering a choice between moving to two different piles to be drilled next. For instance, the next one to the left or to the right.

Figure 2.12 shows a STROBOSCOPE process diagram, where the resource flow of the drilling rig encounters a fork element after drilling, offering two choices of which way to go to the next hole. In a larger context the fork element can also be used to switch between multiple resource flows, assigning different processes to different machines. Though it is very complex to allow the production of the individual piles in an arbitrary order, as all those piles would need to enter the system at the same time and many fork elements would need to model the choices. In the example the choice can be viewed as moving to the next pile that needs to be drilled, either to the left or to the right. If additional piles or machines would be introduced, the entire model would need to be updated.

2.5.6 Activity-Based Construction (ABC)

Shi (1999) presented the Activity-based construction (ABC) modeling and simulation method. Compared to the previously described approaches ACD and CYCLONE, ABC uses an activity-based representation of the construction processes. The representation of a construction project is reduced to a collection of activities and resources. Each activity contains certain attributes, including duration of execution, logical dependencies and resource demands. Compared to ACD and CYCLONE, there is no distinction between active and idle states as those arise from the constraints, such as resources and precedence relationships, being met. A process that requires a resource that is unavailable, is naturally forced to be idle. Resources are indicated by colors and arrows indicate precedence relationships.

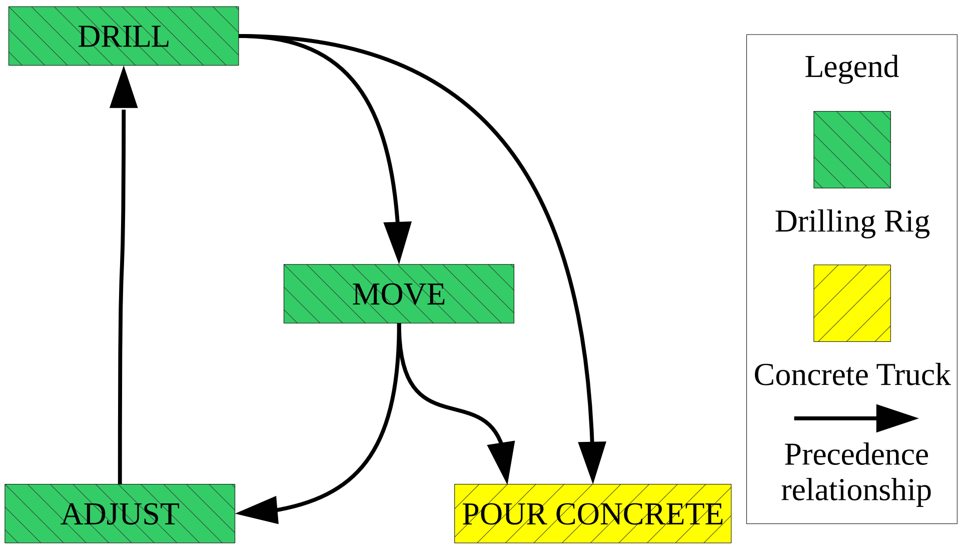


Figure 2.13: Bored pile wall example as an activity-based construction diagram. The diagram only consists of the four activities "DRILL", "MOVE", "ADJUST" and "POUR CONCRETE". The arrows indicate precedence constraints. The fill color of the rectangles indicate the required resource. In order to drill, the rig needs to be adjusted, in order to move to the next location it needs to have finished drilling, and in order to adjust it needs to have moved to the next location. Before the concrete can be poured, the hole needs to be drilled and the rig needs to have moved away from the hole.

The bored pile wall example in this notation is illustrated in Figure 2.13. It shows only four activities in two colors. Green implies that the activity requires the drilling rig to be available, while the yellow activity requires the concrete truck to be executed. Again, starting with the "MOVE" activity, the "ADJUST" activity can start directly afterwards. Immediately after that the "DRILL" activity can commence. After the "DRILL" activity is finished, the drilling rig needs to move to the next location, hence the "MOVE" activity needs to be finished, for the "POUR CONCRETE" activity to have satisfied precedence constraints. Given the concrete truck is available, the activity can be executed. Similar to CYCLONE, this approach allows for the bored pile wall example to be modeled, allowing parallelization of the two cycles, while being synchronized through precedence relationships. When the drilling rig moves to the first location to be drilled at, only one of the two precedence constraints of the "POUR CONCRETE" activity is satisfied, only after the first hole is drilled, and the drilling rig has moved on, the activity becomes executable. This example shows that ABC offers a much more intuitive way to model construction operations, that are much more accessible to non-professionals.

2.6 Modeling and Simulation of Construction Projects

As construction projects in practice can hardly be modeled as a collection of cyclic activities, a more complex approach to modeling entire projects is required. Building up on the previously described Activity-Based Construction approach, a non-cyclic variation of this representation is shown in Figure 2.14. In this model, each activity is only executed once, then marked as done. Therefore each pile to be drilled creates its own individual chain of activities. This approach is very similar to the event-graph based simulation modeling approach by Schruben (1983). The shown example assumes there to be just one drilling rig, and therefore execution of the individual piles is strictly sequential, hence, no two piles can be drilled at the same time, as each "MOVE" activity needs the previous "DRILL" activity to be marked as done, as is indicated by the precedence relationships, shown as arrows. The "POUR CONCRETE" activities are only dependent on the "MOVE" activity, hence that the drilling rig has moved on to the next drill location.

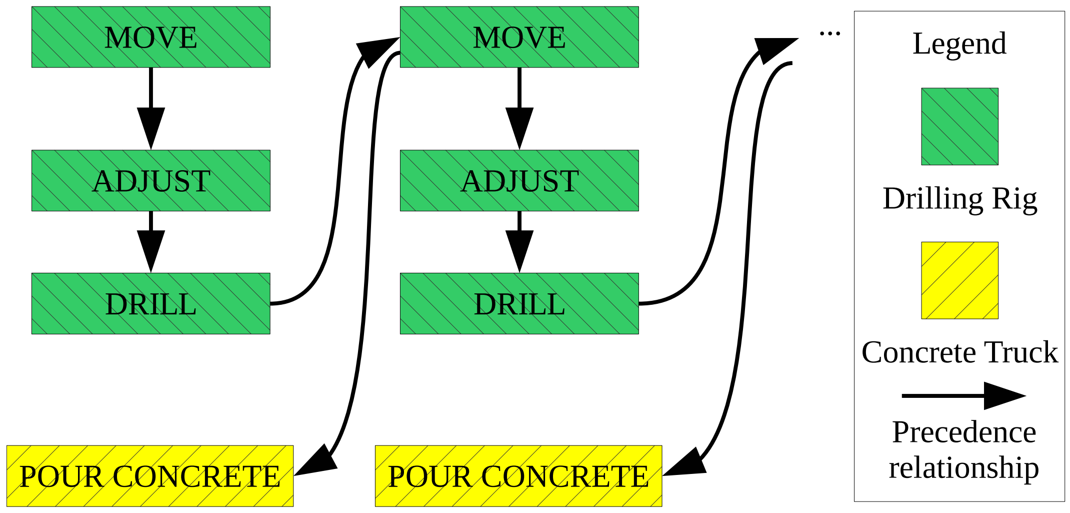


Figure 2.14: Bored pile wall example as a non-cyclic sequential process diagram. In this representation, each rectangle is an activity that will only be executed once, and then marked as finished. The order of execution needs to respect the precedence relationships indicated by the arrows and the availability of resources, which are indicated by the fill color of the activity rectangles. The diagram enforces the order the piles are produced in.

This representation, though, is impractical for general execution, as there could be more than one drilling rig involved in the execution. Furthermore the bored pile wall process would usually be included in a larger project.

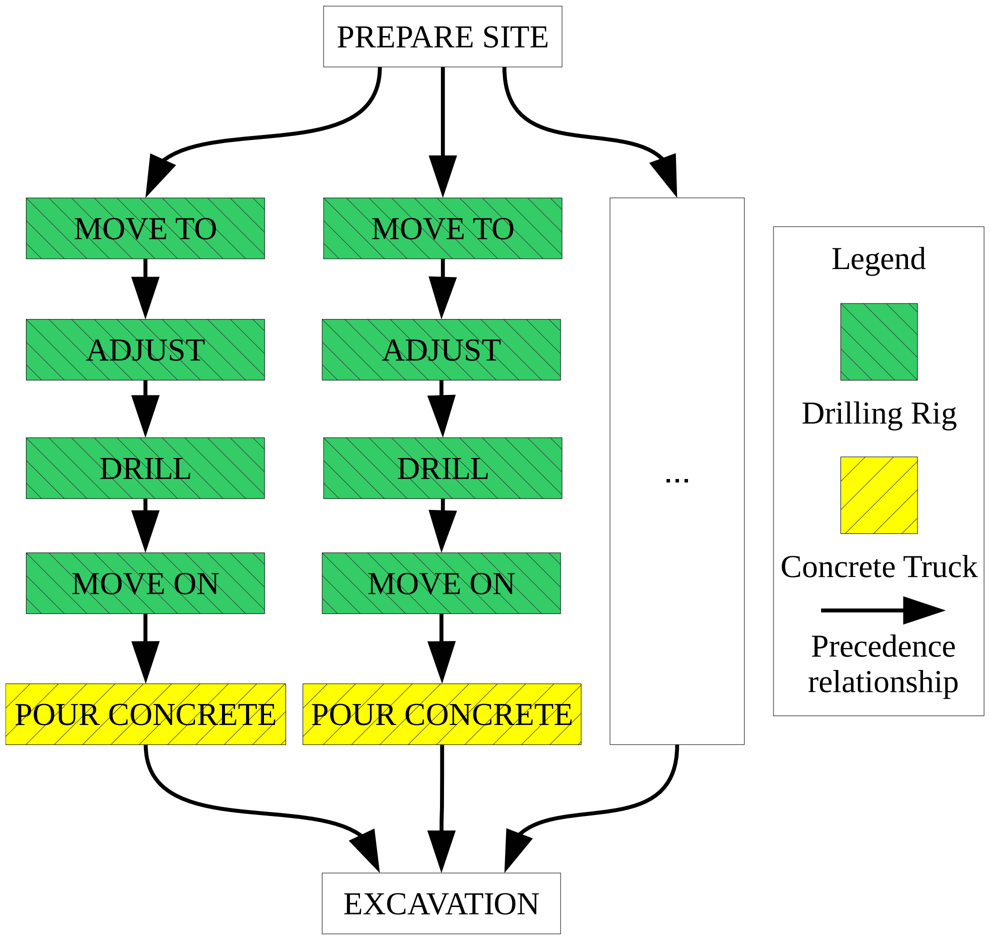


Figure 2.15: Bored pile wall example as a non-cyclic parallel process diagram. Compared to the diagram in Figure 2.14, this diagram does not enforce the order of pile production and allows full parallel execution if multiple drilling rigs or concrete trucks are present.

The representation shown in Figure 2.15 allows full parallelization of the individual drilling and concrete activities, while being embedded in a larger context. First the site needs to be levelled in preparation, when this is done, drilling rigs can move to the individual location where holes need to be drilled. After adjusting and drilling, the rig needs to "MOVE ON" in order to free the hole for a concrete truck to be able to execute the "POUR CONCRETE" activity. The additional "MOVE ON" activity is required as it is not specified where the drilling rig moves after finishing drilling. When all piles are finished, the pit, now secured by the bored pile wall can be excavated. This representation allows full parallelization with an arbitrary number of drilling rigs and concrete trucks, though it does not take into account that the drilling rig, usually blocks an area blocking not only a single drilling location. The number of parallel paths in the graph is determined by the number of piles to be produced, that do not depend on each other.

Figure 2.16 additionally includes a simple geometry representation, which on the one hand does not require the artificial "MOVE ON" activity and on the other hand allows the

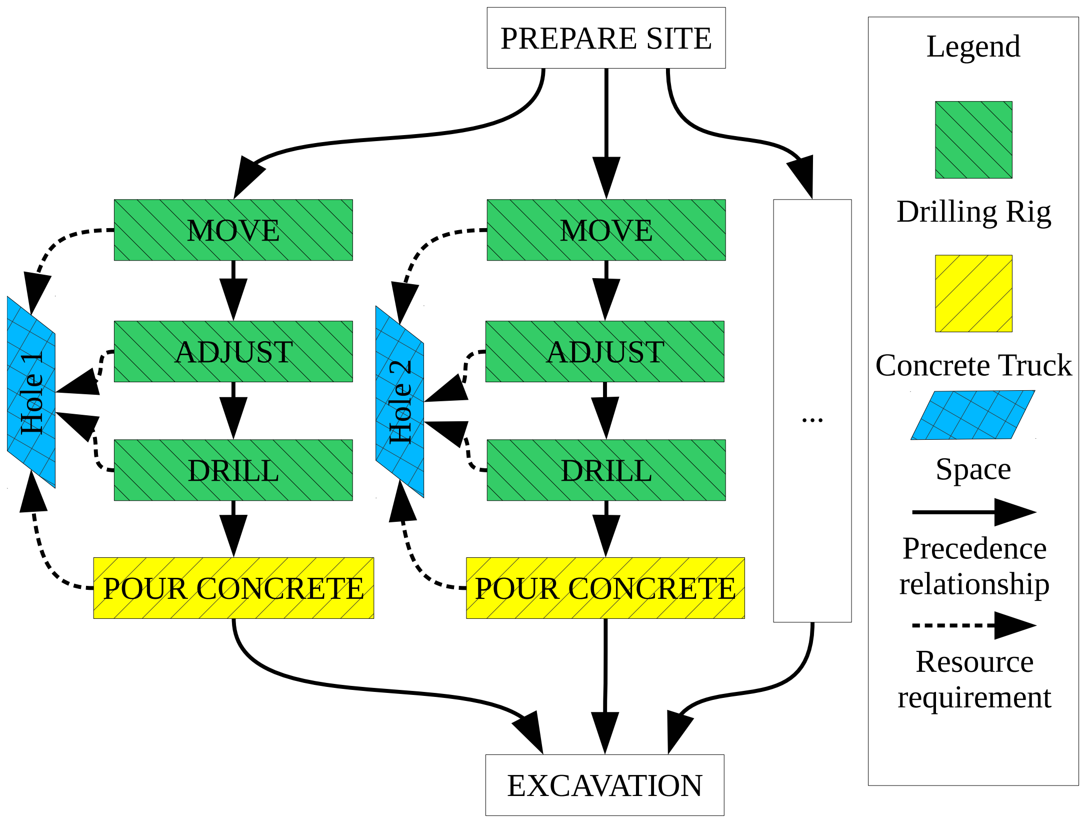


Figure 2.16: Bored pile wall example as a non-cyclic parallel process diagram with geometry. The diagram in Figure 2.15 extended by a representation of geometric space, indicated by the blue parallelograms. The spaces may be overlapping as the drilling rig might block multiple holes while positioned above one hole.

representation of overlapping areas. In order to visualize this, the parallelogram shape is introduced, representing a specific area or volume on the construction site. The dotted arrows indicate activities blocking that geometric space. Therefore only one activity blocking a certain geometric space can be active at the same time. Geometric resources are handled exactly like any other resource, except that blocking one geometry might block multiple other geometries due to overlap. This allows for detailed modeling of work spaces, machine sizes, and safety zones.

2.6.1 Activities-on-the-Arrows (AoA) versus Activities-on-the-Nodes (AoN) representation

In Figure 2.16, the problem is illustrated as a directed acyclic graph, where each node represents one activity and edges represent precedence relationships. This notation is called the Activities-on-the-Nodes (AoN) notation. In contrast to this notation, the Activities-on-the-Arrows (AoA) notation represents activities by edges and the nodes represent states (Ponz-Tienda et al., 2015). Figure 2.17 shows an AoA representation of the bored pile wall example. As the AoN representation is more intuitive to be read and both representations can be transferred from each other (Yang & Wang, 2010), the AoN representation is used throughout this thesis.

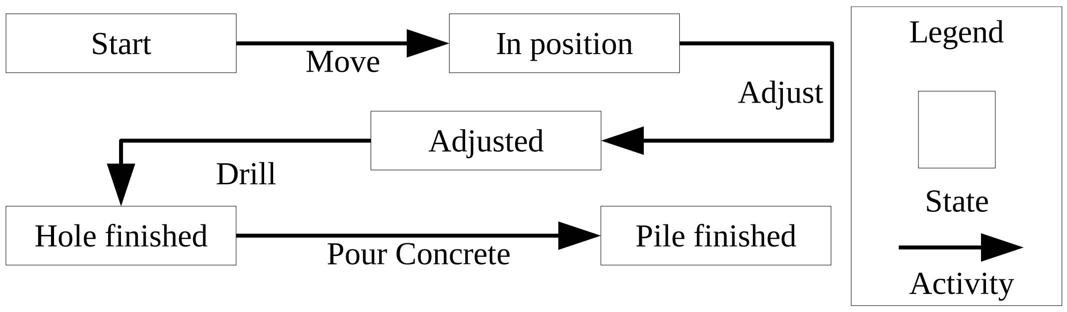


Figure 2.17: Bored pile wall example as an Activities-on-the-Arrows (AoA) diagram. Each rectangle represents a state and the arrows indicate activities, causing a state change.

2.6.2 Elements of a Construction Project Model

In order to adequately describe a construction project several elements are required, which need to be acquainted by different attributes. This Section provides an overview of the different classes of elements and their capabilities.

2.6.2.1 Activities

Activities are actions performed on and around a construction site, contributing to the overall progress of the project (Zhang et al., 2005b). Each activity is defined by a set of parameters. The duration specifies how much time it takes to execute the action. Furthermore an activity may require certain resources, such as machines, or materials, such as concrete, to allow execution. Additionally it may be required that certain other activities must be finished or in progress, for the activity to be executed. Most activities create costs by using resources or materials, so these costs are also an important parameter of an activity. The costs may also be a function over the duration. In order to determine robustness of schedules, the uncertainty of each activity's duration is an important measure (Artigues et al., 2013; Deb & Gupta, 2006). Therefore a probability distribution for the duration is part of the required parameters and may influence the cost parameter, as a prolonged activity might increase running costs, like fuel or wages. More precisely those costs can be modeled on a resource level, such that the activity itself does not create costs, but the resources and materials that are used do. A milestone can be viewed as a special kind of activity that does not have costs, duration, or uses any resources, but is dependent on some other activities to be finished.

2.6.2.2 Resources

Resources describe renewable entities on the construction site that can be temporarily used by activities. Resources are commonly available on site in limited quantities and therefore form one of the strongest constraints in construction scheduling. The usage of resources may be accompanied by costs, such as fuel or wear and tear. Unless modeled on an activity-level, the costs for using, hiring, and transporting resources are parameters of those resources.

2.6.2.3 Materials

Materials are non-renewable entities on site, that can be consumed by activities. Despite materials having associated costs, the amount of materials used is usually constant on a construction site, hence it does not change when activities are performed at different times. Including costs of materials into the project model still might be relevant as it is possible to calculate the cost flow during the execution phase, as well as there might be strict budgets for the different construction phases.

2.6.2.4 Geometric Space

Every activity that is performed on a construction site will require some space to performed on, that will be unavailable for other activities during its execution. Those spaces can be represented either as two dimensional areas on the site, or as three dimensional volumes inside a building. As geometric conflicts form a strict limit on the feasibility of construction schedules, it is important to include geometry into the project model.

2.6.2.5 Precedence Constraints

Most of the constraints limiting the execution of construction schedules are logical constraints, such as a slab requiring underlying columns in order to be produced, or a wall, that needs to be plastered prior to being painted. This creates start-finish (SF) precedence constraints, as described in Section 3.3.1. Additionally safety considerations might create further limitations (Melzner et al., 2012) that can for instance prevent concurrent execution of certain activities. Furthermore soft-constraints can be included, representing practical aspects of certain activities, such as dirty activities should be finished before painting the walls (Beißert et al., 2010). A special kind of constraint is the start-start (SS) precedence constraint which requires another activity to have started execution, but it does not necessarily have to be finished (Section 3.3.1).

2.6.3 Computer-Aided Design (CAD) and Building Information Modeling (BIM) as Data Source

Construction processes can be derived from the product that has to be produced. The required durations are mostly results of product metrics, such as volume or surface areas. Those metrics can be extracted from digital representations of the product.

For this purpose Hendrickson et al. (1987) developed a system called CONSTRUCTION PLANEX, an expert system for the automatic generation of schedules based on a database of construction techniques and resource requirements. It formed the first approach towards the creation of a product model based approach to construction scheduling, taking all the most important factors into account, including precedence relationships, activity durations, and costs.

Another approach to this problem was published by Navinchandra et al. (1988), introducing a system called GHOST, which started from a very optimistic plan with components, connected by few or no precedence constraints, then using a knowledge base to refine that model by introducing constraints to remove violations. This allows to detect shortcomings in the available models and develop rules for improving future models.

Winstanley et al. (1993a,b) designed a similar system, called OARPLAN, that linked activities to components. So instead of assuming that one component is created by one process, multiple activities can be required for a single component or vice versa. The information required to do so was extracted from CAD models.

AbouRizk & Mather (2000) presented the idea of simplifying the development of simulation models for earthmoving operations by an integration with CAD modeling. Chahrour (2006) generalized this idea and developed an integrated CAD-based simulation model, that allowed planners to enrich the CAD model by data, required for an integrated simulation model, during the modeling process. Chahrour & Franz (2006) argue that this step greatly aids the practical usability of simulation models, as CAD systems are a commonly known environment among construction planning experts.

Building Information Modeling (BIM) is the process of creating digital representations of existing buildings, or during the planning phase of new buildings. The models, besides geometry, should contain physical and functional characteristics of the building. The concept was first envisioned by Law & Jouaneh (1986) and Björk (1989, 1992). Popov et al. (2006) presented possibilities to use those digital building models during the entire process of design, scheduling and construction. Lately this idea gained increasing popularity in the construction industry (Dossick & Neff, 2010), while even more capabilities of BIMs were discovered, including safety checking (Melzner et al., 2012), progress monitoring (Bosché et al., 2015; Braun et al., 2015a; Matthews et al., 2015), and productivity assessment (Poirier et al., 2015).

Tauscher (2011) and Tauscher et al. (2014) took the approach of Chahrour & Franz (2006) into the BIM age by presenting an approach to automatically extract construction sequences from building information models. The increase in information available through the BIM approach allowed to minimize the required human interaction for construction simulation modeling.

Dzeng & Tommelein (2004) developed a case-based system, that tries to identify previously encountered patterns on the basis of a knowledge-base, and apply solutions that were executed in the past. The identification is performed on the basis of product models similar to the approach by Fischer & Aalami (1996)[1]. Later the idea was developed further into a system tailored to work with Building Information Models by Sigalov & König (2015).

Query languages, such as QL4BIM (Daum & Borrmann, 2014) can use spatial con-

[1]Up to this point, this Section was in part inspired by Huhnt & Enge (2006).

straints as a filter criterion applied to building information models. The resulting constraints are based on geometric relationships, such as a slab resting on a set of columns, which need to be produced prior to the slab (Braun et al., 2015a). Those kinds constraints can be used to create a rough precedence graph of a building to be constructed. Figure 2.18 shows an example of a building model with six stories. The generated precedence graph is displayed next to the building. Each of the nodes represents one activity and the connections represent precedence relationships. The graph narrows down on each slab, as the following columns and other elements can only be constructed after the slab has been produced.

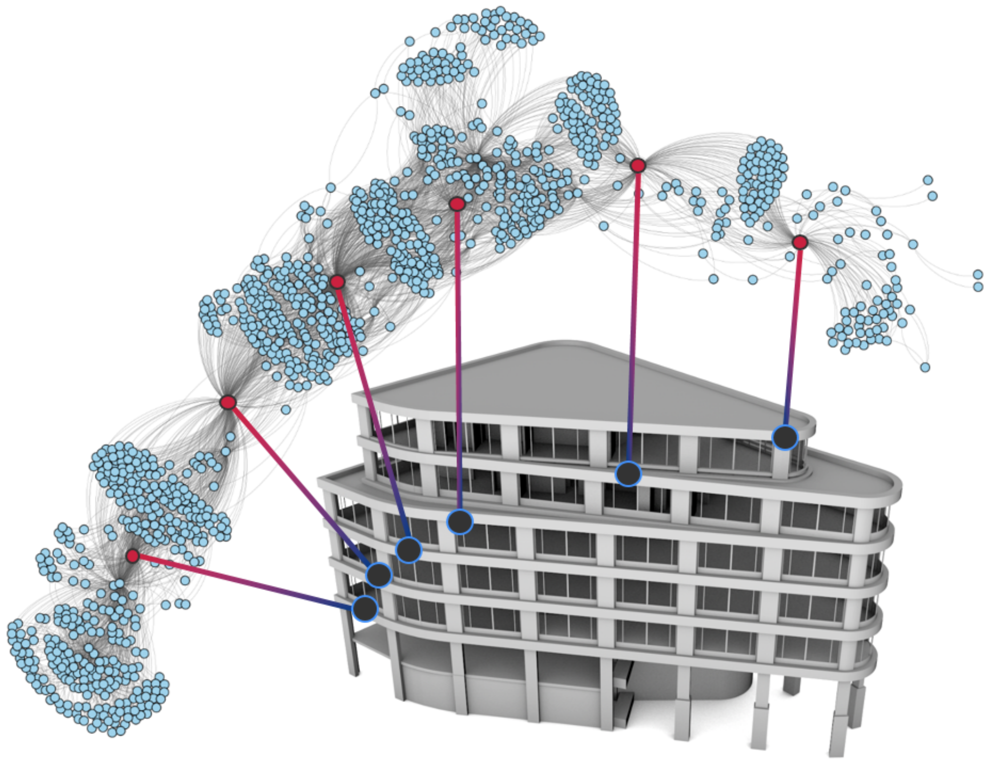


Figure 2.18: Building information model and its automatically derived precedence graph. Illustration from Braun et al. (2015a).

Marx & König (2011), König et al. (2012), as well as Braun et al. (2015b) showed that this concept is equally feasible on a larger scale in real-life BIMs. Additionally building information models can contain information about building materials, as well as building techniques. This data gives rise to the involved activities, required resources, materials, durations, and geometric needs.

2.6.4 Discrete-Event Simulation of Construction Projects

Once a representation of a construction project has been modeled, it can be used to generate a feasible schedule. Since such a schedule is not necessarily good or even optimal, optimization procedures such as those described in Chapters 3 and 4 are highly useful. As construction operations naturally consist of many individual activities that start and finish at specific points in time, they can be described as a finite number of discrete time instances, where activity states are changing. The simulation of a system, which is defined by a state that changes only at discrete time instances is called discrete-event simulation (DES). This notion was formalized by Fishman & Kiviat (1967).

Since discrete-event simulation models exactly that kind of systems it is an appropriate tool to use when simulating construction projects. In order to run a discrete-event simulation system, the state of the system has to be defined for each time step and the times, when the state changes need to be determined. In case of the construction simulation model used throughout this thesis the state of the system is defined as the states of each activity and the availability of each involved resource.

The precise definition of an entire construction project in these terms is described in Section 2.6.2. Schruben (1983) describes simulation modeling in terms of event graphs and introduced the notation shown in Figure 2.19. It shows two events j and k connected by a transition taking time t and requiring condition (i). The event graph based representation is an activities-on-the-arrows (AoA) representation as described in Section 2.6.1. However it has been shown that an activities-on-the-nodes (AoN) representation is more intuitive for the user and can be transformed to an AoA notation (Yang & Wang, 2010). This means that the events j and k correspond to the start and end times of an activity, the transition time t is the duration of the activity, and that (i) describes precedence and resource constraints.

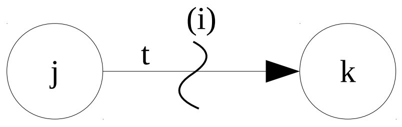


Figure 2.19: Notation of simulation modeling with event graphs (Schruben, 1983). Two states j and k are connected by an activity that requires time t and constraints (i) to be executed.

In order to perform a simulation run, the resource constraints need to be set, by setting a counter for each resource to the number of available units. The simulation clock is initialized at time $t_0 = 0$. Then a list of activities with no precedence constraints is assembled. From this list, an activity A is selected and scheduled to be executed at time

$t_0 = 0$. The required resources are allocated by reducing the respective counters by the required numbers. A new event is generated at the finish time $t_f(A)$ of activity A. The activity is now considered active. Then the list of executable (precedence and resource feasible) activities is updated and, if possible, another activity is scheduled at $t_0 = 0$, and an event is generated for the finish time. If no more executable activities are available, the simulation clock is advanced to the earliest finish time of all active activities, which defines the next discrete event. As at this event one or more activities end, the respective resources are freed and the activity is marked as finished and is no longer active. At this point a new list of precedence and resource feasible activities is generated, and an activity is selected for execution. If no more activities are executable the simulation clock is advanced. This is repeated until all activities have been scheduled. This process is illustrated in the flowchart shown in Figure 2.20. The node *Select feasible activity* allows for different selection rules that select activities from a set of currently feasible ones. The order in which the activities are scheduled forms a degree of freedom in scheduling. Based on this degree of freedom different optimization procedures can be deployed as is described in Chapters 3 and 4. Beißert et al. (2007) and Beißert (2010) introduced the constraint-based discrete-event simulation system, where the activity selection is performed randomly and the results are evaluated using Monte Carlo Methods. Other approaches use different priority rules, resulting in the parallel schedule generation scheme that is described in detail in Section 3.4.4.3.

Generally the process of simulation, either as a type of schedule generation scheme, or as constraint-based discrete-event simulation, forms an important basis for many heuristic algorithms solving resource-constrained project scheduling problems, which are introduced in Section 3.4. Furthermore specific metrics obtained through the simulation process can be used for certain scheduling procedures, such as the float time calculation presented in Section 3.4.6.

As described in Chapter 3 in terms of schedule generation schemes, this process is not necessarily capable to find the optimal schedule as defined in Section 3.4.4.1.

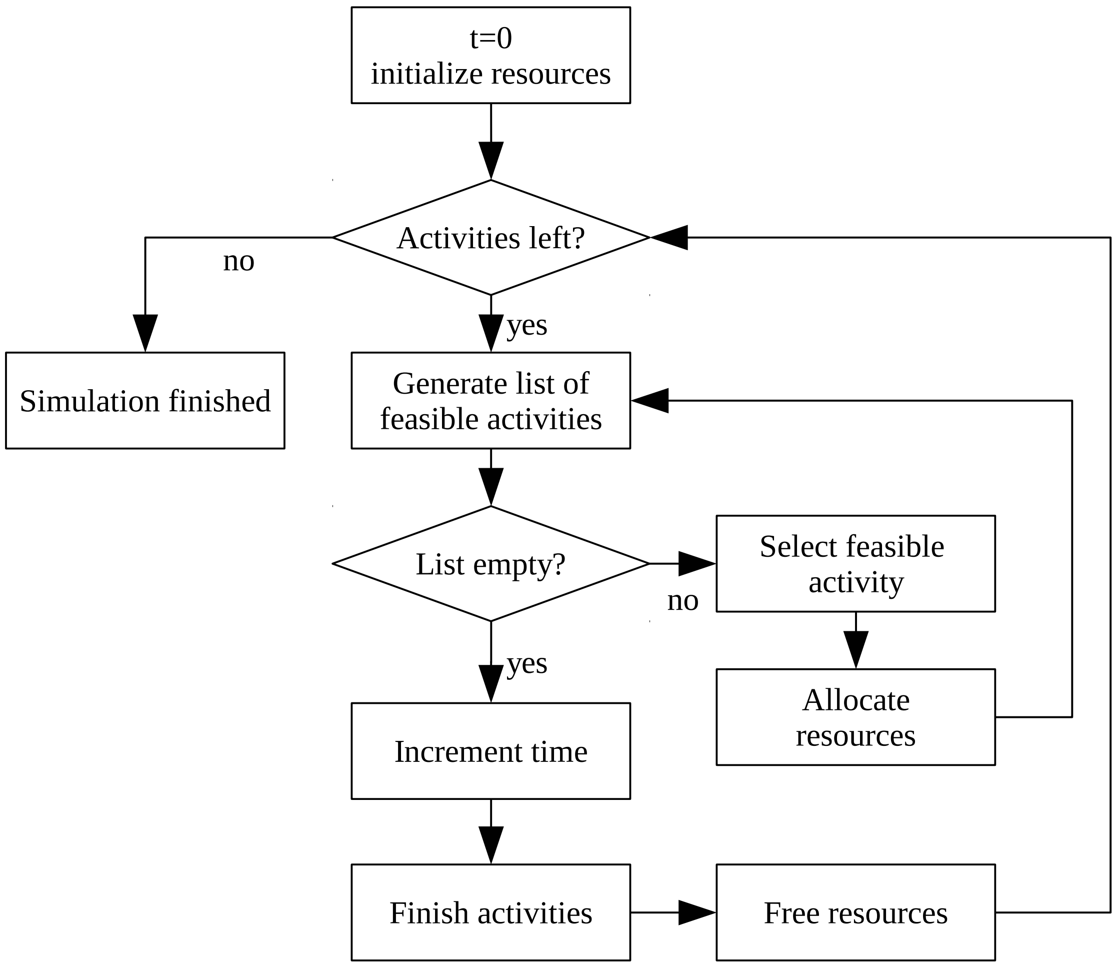


Figure 2.20: Flowchart of the discrete-event simulation process. At time $t = 0$ the system is initialized with all resources idling. As long as there are unscheduled activities the main loop continues. It creates a list of feasible activities, selects a feasible activity, executes it and allocates the respective resources. Once no more activities are feasible, time is incremented to the next finishing time, activities are finished and the resources freed.

2.7 Summary

This Chapter introduced the concept of performance factors, the modeling approaches to represent uncertainty in terms of different probability distributions, representations of costs as well as the trade-offs to be made when deciding upon the granularity of process and project models. The state-of-the-art in cyclic process modeling and non-cyclic project modeling has been laid out in terms of the required components and the different modeling approaches. Finally discrete-event-simulation approaches to perform simulations of the created models have been described.

Simulation is a very capable tool to investigate systems theoretically, based on an abstract description. While very successful in academic research, construction simulation has not yet been widely adopted by the architecture, engineering, and construction (AEC) industry for various reasons, including lack of simplicity, as well as time, costs, and skills required to set up a usable simulation model (Scherer & Ismail, 2011).

A major limitation is the accessibility of simulation systems to the construction professional, who is not necessarily a simulation expert. In the construction context, this process can be reduced to describing the activities required to complete parts of a project, or a complete project, their durations, precedence relationships, and resource demands, as described throughout this Chapter. This makes the simulation domain far more accessible, as opposed to describing the problem using the General Purpose Systems Simulation (GPSS), or even programming in general purpose programming languages.

Furthermore advances in the field of building information modeling provide novel data sources, increasing the usability of simulation systems even further. This approach however is still subject to different limitations. Due to required abstractions the model may not adequately reflect the reality on the construction site. It is up to the user to decide upon the trade-off between granularity and design and simulation effort. Furthermore the input data, such as expected activity durations may be inaccurate and lead to discrepancies. This can however be addressed by a representation of the uncertainty in the durations, costs, or underlying performance factors. Overall these limitations and trade-offs need to be known to the user of construction simulation systems in order to create sufficiently accurate, but feasible simulation models.

Once a model of a project, its individual processes, as well as a simulation approach has been chosen, there are different ways to generate schedules for the execution of the building process. As these schedules are vital for the success of a project it is desired to generate schedules that are as close as optimal with respect to the desired metrics as possible. Therefore different optimization approaches have been developed and published in literature which are presented throughout the next Chapter.

3 State-of-the-Art in Construction Schedule Optimization

"The construction professional has to be a 'jack of all trades, and master of all.' " - Halpin & Senior (2010)

A reactive simulation system needs to dynamically take changed circumstances into account and react accordingly. As the main surface to act upon is formed by the order the activities are executed in, schedule optimization algorithms are of large interest in this context.

Large scale construction sites commonly consist of thousands of individual intertwined activities that are constrained by numerous factors, such as precedence relationships, resource sparsity, and a lack of space. The problem of arranging all those activities in a way that satisfies all constraints, while additionally minimizing a given objective, is called project scheduling. In practice a construction project schedule has several important aspects that need to be kept in mind. This includes the high volatility of construction processes that require construction schedules to be robust against delays. Critical processes need to be identified, as they need special attention. Furthermore the objectives for optimizing construction schedules need to be chosen accordingly and must reflect all important aspects, such as duration, costs, and quality.

The process of project scheduling is mostly performed manually in practice, based on the experience of the planner. This becomes increasingly complex and error-prone for larger projects. While the formal project scheduling problem can be solved easily, its extension, taking resources into account, the resource constrained project scheduling problem, is NP-Hard in the strong sense (Blazewicz et al., 1983). This means that deterministic algorithms are hardly applicable for large scale problems. This Chapter therefore mainly focuses on heuristic algorithms.

First, optimization objectives and schedule metrics are introduced, followed by a description of the incorporation of uncertainty and determination of robustness into the optimization process. Then the handling of multiple optimization criteria is discussed. As an example for the proposed methods, an illustrative anchored bored pile wall project is introduced, that will be used throughout this Chapter. Then the Project Scheduling Problem and the Resource Constrained Project Scheduling Problem are introduced. Furthermore the different types of schedules and schedule generation schemes are discussed.

3.1 The Anchored Bored Pile Wall Project Example

To illustrate the effects of resource sparsity on construction scheduling a simple example of an excavation pit is introduced. A rectangular pit has to be excavated, but is surrounded by soil, threatening to cave in on two sides. In order to prevent that from happening two bored pile walls will be created. As the pit will be excavated to great depth, the bored pile walls need to be supported by a layer of anchors. The layout of the site is shown in Figure 3.1.

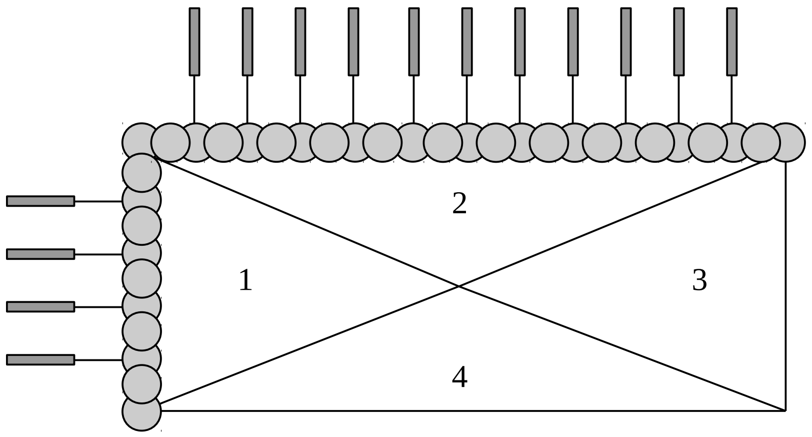


Figure 3.1: Bored pile wall project example illustration viewed from above. A rectangular pit is to be excavated. To prevent earth from caving in, two sides of the pit have to be secured using anchored bored pile walls.

It shows the two bored pile walls, which meet in the upper left of the illustration. Every second pile will be anchored into the surrounding soil to prevent the piles from falling over. The illustration shows four distinct excavation areas labeled 1 through 4. In order for the anchors to be placed, the respective bored pile wall needs to be finished, and the excavation area has to be partly excavated to the depth the anchors should be placed and in order to make room for the required machinery. After the anchors have been placed the remaining excavation of the area can proceed. After all excavation activities are finished the foundation can be produced. The degrees of freedom of this example consist of the order the excavation areas are excavated and when the bored pile walls are produced. Areas 1 and 2 depend on the bored pile walls, while 3 and 4 are independent.

3.2 Objectives of Optimization and Schedule Metrics

The most commonly used objective of the optimization of schedules is the makespan, the overall duration of the schedule. In construction projects on the other hand, the costs are often more relevant, but also more complex to determine. There are milestones to be met which are subject to fines when delayed. Furthermore machines create costs, even when not used, but the transportation of machines from and to the site can also require extensive efforts. Therefore it can be more economic to have a machine idle for a certain time, as opposed to moving it to another site for the time it is not required. The costs for the availability of machines, but also the stocking of materials should be part of cost oriented optimization.

Another important aspect is the robustness of a schedule, which describes the limited impact of the changes in durations of individual activities on the overall makespan. Additionally leveling the usage towards constant utilization of certain resources can be desired, forming the Resource Leveling Problem (RLP) (Ponz-Tienda et al., 2013).

Generally, optimization problems in the domain of construction scheduling are in most cases multi-objective problems, such that for instance resource usage and the overall makespan are objectives to be minimized.

3.2.1 Uncertainty and Robustness

Sabuncuoglu & Goren (2009) and König (2011b) describe proactive and reactive approaches to dealing with uncertainty of construction operations. Reactive approaches imply that the schedule is updated when deviations occur, while proactive approaches try to anticipate the deviations by modeling uncertainty as described in Section 2.2 and therefore maximize the robustness. Since construction processes are highly volatile, robustness may well be among the most critical measures in construction schedule optimization.

König (2011a) defines a robustness of a construction project schedule as minimizing the effects of forseeable variations of the durations of construction activities on the schedule. Herroelen & Leus (2004) define it as incorporating a degree of anticipation of variability during project execution. They furthermore distinguish quality robustness and solution robustness. Quality robustness means that deviations in activity durations have minimal impact on the quality of the schedule, which implies that the changes to the value of the objective, such as the makespan, are minimal. Solution robustness means that the changes to the schedule that may be caused by deviations are minimal.

Furthermore Chtourou & Haouari (2008) defines twelve measures of robustness of a given schedule in terms of the slack s_i, the number of direct successors N_{succ_i}, the number of resources of type k used r_{ik}, the average percentage increase in activity duration κ_i and the slack indicator α_i defined in Equation 3.1. All measures are listed in Table 3.1. The first measure $\mathcal{M}_1$ maximizes the sum of all float times, implying that a schedule with more float time is more desirable. If for instance a single activity with high uncertainty would have zero float time, this would be considered equally good than an activity with very low or no uncertainty having zero float time. Also activities with a large number of follow up activities are more desirable to have larger float times, which is expressed by $\mathcal{M}_2$. $\mathcal{M}_3$ instead takes the resource usages into account by making high float times for activities with high resource usage more desirable, as resources might be blocked by previous delayed activities and therefore would cause more potential instability. $\mathcal{M}_4$ is a combination of both previous measures. $\mathcal{M}_5$ maximizes the number of non-critical activities. A lower number of activities with zero float time is preferred. $\mathcal{M}_6$ again takes the number of successors into account, while $\mathcal{M}_7$ takes resources into account and $\mathcal{M}_8$ combines both aspects. $\mathcal{M}_9$ minimizes the total average increase in duration, given it is lower than the float time for each activity. $\mathcal{M}_{10}$, $\mathcal{M}_{11}$, and $\mathcal{M}_{12}$ expand this idea by taking successors and resources into account.

$$\alpha_i = \begin{cases} 1 & \text{if } s_i > 0 \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in \boldsymbol{J} \tag{3.1}$$

$$\begin{aligned}
\mathcal{M}_1 &= \sum_{i \in \boldsymbol{J}} s_i & \mathcal{M}_7 &= \sum_{i \in \boldsymbol{J}} (\alpha_i \sum_{k \in \boldsymbol{R}} r_{ik}) \\
\mathcal{M}_2 &= \sum_{i \in \boldsymbol{J}} s_i N_{succ_i} & \mathcal{M}_8 &= \sum_{i \in \boldsymbol{J}} (\alpha_i N_{succ_i} \sum_{k \in \boldsymbol{R}} r_{ik}) \\
\mathcal{M}_3 &= \sum_{i \in \boldsymbol{J}} (s_i \sum_{k \in \boldsymbol{R}} r_{ik}) & \mathcal{M}_9 &= \sum_{i \in \boldsymbol{J}} \min(s_i, \kappa_i t_i) \\
\mathcal{M}_4 &= \sum_{i \in \boldsymbol{J}} (s_i N_{succ_i} \sum_{k \in \boldsymbol{R}} r_{ik}) & \mathcal{M}_{10} &= \sum_{i \in \boldsymbol{J}} \min(s_i, \kappa_i t_i) \cdot N_{succ_i} \\
\mathcal{M}_5 &= \sum_{i \in \boldsymbol{J}} \alpha_i & \mathcal{M}_{11} &= \sum_{i \in \boldsymbol{J}} (\min(s_i, \kappa_i t_i) \sum_{k \in \boldsymbol{R}} r_{ik}) \\
\mathcal{M}_6 &= \sum_{i \in \boldsymbol{J}} \alpha_i N_{succ_i} & \mathcal{M}_{12} &= \sum_{i \in \boldsymbol{J}} (\min(s_i, \kappa_i t_i) N_{succ_i} \sum_{k \in \boldsymbol{R}} r_{ik})
\end{aligned}$$

Table 3.1: The 12 measures of robustness by Chtourou & Haouari (2008)

Another way of determining the robustness of the resulting schedule, if the probability distributions of the activities' durations are known or can be determined as described in Section 2.2, is Monte-Carlo simulation (Wübbeler et al., 2008). This approach is based on performing multiple simulation runs or schedule generations, while for each run, sampling new duration values from the respective distributions. When sufficiently many runs are performed, a distribution of the overall makespan forms, which gives rise to the probability for delays. For instance the probability of certain milestones being met can be calculated on this basis.

In a more general context Chassein & Goerigk (2016) describe the problem of robust optimization as a bicriterial problem in which the two criteria are formed by the average and the worst-case performance. Both of these criteria should be subject to optimization. Therefore multi-criteria optimization is an important part when aiming to take robustness into account.

3.2.2 Multi-Criteria Optimization

Creating a schedule with a maximal robustness can be a good objective in construction schedule optimization, but as a mere objective it would yield schedules which maximize slack and idle times. This would tend to create unnecessarily long and expensive schedules. On the opposite, a schedule minimizing costs or makespan will tend to be overly fragile. Furthermore the usage of additional resources can reduce the makespan, while increasing the costs, implying a trade-off between costs and makespan. It is therefore important to take multiple objectives into account. As most optimization algorithms are designed to minimize a single objective function, ways of evaluating the superiority of a certain schedule, or mapping multiple objectives to a single objective function are desired. Usually the user has to decide upon which objectives are of more or less importance.A survey of such methods was published by Marler & Arora (2004), on which the following Sections are loosely based.

3.2.2.1 Pareto Optimality

In the field of multi-critertia optimization the work of Pareto (1906) plays an important role, as he defined the terms *Pareto Optimality* and *weak Pareto Optimality*. Lemmas 1 and 2 define the terms respectively.

Lemma 1. *A point, $x^* \in \boldsymbol{X}$, is Pareto optimal iff there does not exist another point, $x \in \boldsymbol{X}$, such that $F(x) \leq F(x^*)$, and $F_i(x) < F_i(x^*)$ for at least one function (Marler & Arora, 2004).*

Pareto optimal points are on the boundary of the space of feasible solutions. A point is Pareto optimal if and only if there is no point having the same or a lower value for all objectives F, but a lower value for at least one objective F_i. The set of points satisfying that condition is called the Pareto Frontier. An example of such a frontier is illustrated in Figure 3.2. The blue crosses are 50 random points and the red line shows the Pareto frontier.

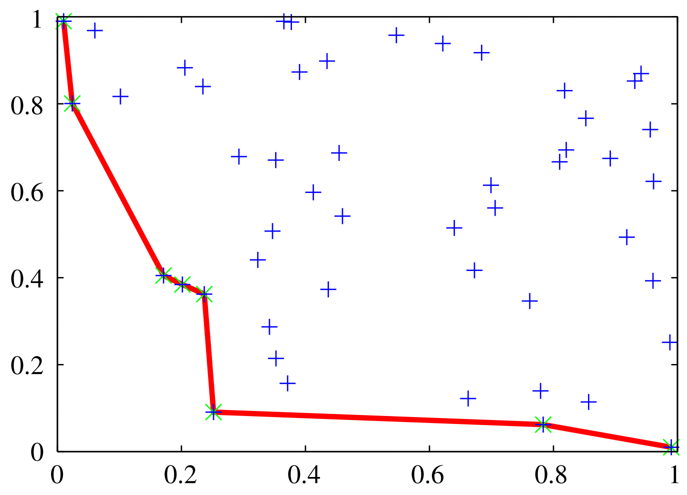


Figure 3.2: Plot of 50 random points and their Pareto frontier illustrated by the thick red line.

.

Lemma 2. *A point, $x^* \in \boldsymbol{X}$, is weakly Pareto optimal iff there does not exist another point, $x \in \boldsymbol{X}$, such that $F(x) < F(x^*)$ (Marler & Arora, 2004).*

A point is weakly Pareto optimal if and only if there is no other point that improves on all objectives. All points that are Pareto optimal are also weakly Pareto optimal.

3.2.2.2 Compromise Solution

An alternative approach to Pareto Optimality is the concept of a compromise solution (Yu & Leitmann, 1974). For this purpose an utopia point, which is a point consisting of the minimal value for each objective, is created. This is formalized in Lemma 3. Mostly the utopia point will never be reached, but a point as close as possible to the utopia point is called the compromise solution (Marler & Arora, 2004). An example of 10 random points with their utopia and compromise points is shown in Figure 3.3. As closeness is commonly calculated using the Euclidean norm, the choice of the closest point may depend on the scale of the individual objective functions. It may be difficult to adequately scale the objective functions in practice, if it is uncertain how different units, such as time and costs, are related.

Lemma 3. *A point F^o is an utopia point iff $F_i^o = \min_x F_i(x)\ \forall i$ (Yu & Leitmann, 1974).*

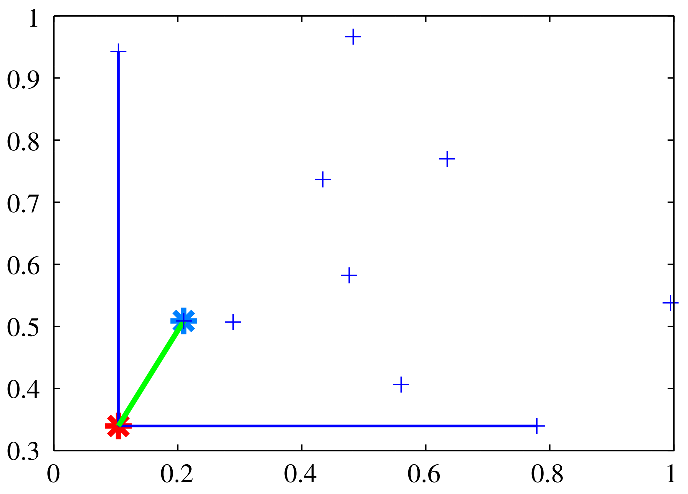


Figure 3.3: Plot of 10 random points with the light blue star at $(0.2; 0.5)$ indicating the compromise point. The thin blue lines indicate the minima for each objective, the red star in the lower left is the utopia point.

.

3.2.2.3 Weighted Global Criterion

For the weighted global criterion all objective functions are combined into a single function by a weighted exponential sum. The simplest form is given in Equation 3.2. It involves a vector of weights w which is set a-priori in order to define the importance of each objective function. The sum of all weights should equate to 1. Furthermore the parameter p is set proportional to the importance of minimizing the function with the largest difference to the utopia value F_i^o (Marler & Arora, 2004). An extension of this method was presented by Yu & Leitmann (1974) and is described in Equation 3.3. It does not require the objective function values to be strictly positive, as it makes use of the utopia point F^o in order to offset each objective function.

$$U_1 = \sum_{i=1}^{k} w_i [F_i(x)]^p, \quad F_i(x) > 0 \, \forall i \tag{3.2}$$

$$U_2 = \left(\sum_{i=1}^{k} w_i^p [F_i(x) - F_i^o]^p \right)^{\frac{1}{p}} \tag{3.3}$$

3.2.2.4 Weighted Sum

The simplest and most common approach to multi-objective optimization is the weighted sum of the individual objectives as given in Equation 3.4. Each objective is assigned a weight a-priori in order to set its importance. This approach is used for the objectives of the uncertainty extension of the PSPLIB problem library described in Section 4.5.

$$U_3 = \sum_{i=1}^{k} w_i F_i(x) \tag{3.4}$$

3.3 Project Scheduling Problem

In the Project Scheduling Problem, a set of precedence-constrained activities has to be arranged in a way - minimizing a given objective (Möhring & Schulz, 2003). A problem is defined as an acyclic directed graph, where each node represents an activity and edges represent precedence relationships. Each activity additionally takes a certain amount of time to be executed. A feasible solution to a project scheduling problem assigns start and end times to each of the activities, such that none of the precedence constraints is violated. It is desired to find a feasible solution that minimizes or maximizes some objective function, such as the makespan or the project costs.

Compared to the previously discussed approaches, the Project Scheduling Problem deals with non-cyclic activities, not taking any resources into account. While the Project Scheduling Problem is not an optimization problem it is the basis of the resource-constrained project scheduling problem introduced in Section 3.4 and many optimization procedures are based on the solution of the associated Project Scheduling Problem. Therefore the two solution procedures for the Project Scheduling Problem are introduced within this Section.

3.3.1 Precedence Constraints

In order to define precedence relationships among the activities different kinds of constraints can be used. While in Section 2.6.2.5 only one type of precedence relationship is described, there are multiple types of such constraints. Figure 3.4 shows the four basic types of precedence relationships with an illustrative feasible schedule, showing the earliest possible start time for *B*, relative to *A*, for each one on the right side. The most common and most intuitive precedence constraint is the Finish-Start (FS) constraint, requiring the first activity to be finished for the next one to start. If not specified otherwise, this type of precedence relationship is the default one used throughout this thesis. The second type is the Finish-Finish (FF) relationship, requiring the second activity to finish at earliest, simultaneously with the first activity. The Start-Finish (SF) constraint requires the second activity to finish at earliest when the first activity starts. Finally the Start-Start relationship requires the second activity to start at earliest when the first activity starts.

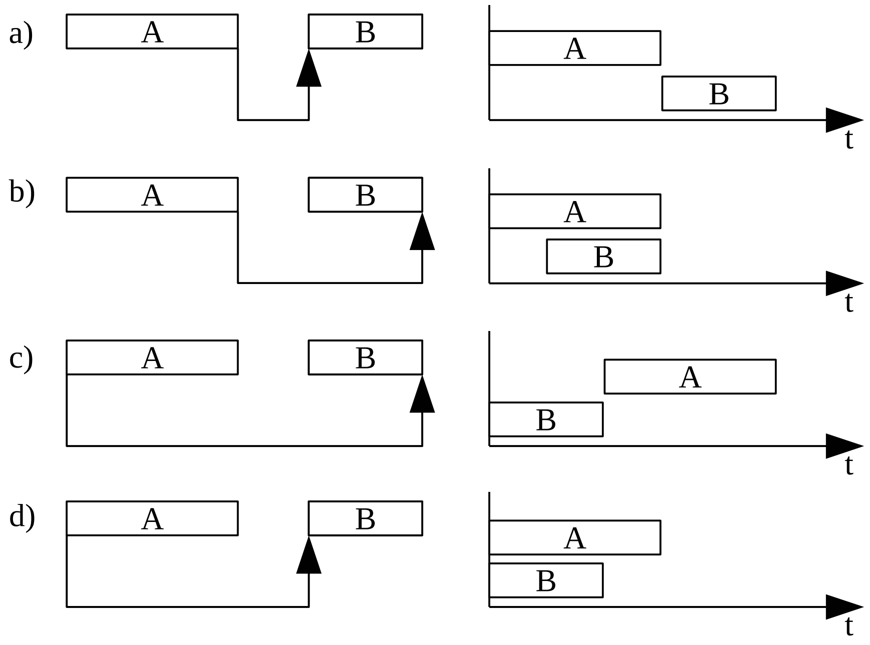


Figure 3.4: Types of precedence relationships. a) Finish-Start (FS), b) Finish-Finish (FF), c) Start-Finish (SF), d) Start-Start (SS).

3.3.2 Graphical Representation of the Bored Pile Wall Project Example

A project scheduling problem is defined by an acyclic directed graph of activities as described in Section 2.6, but without taking resources into account. Each node represents an activity and the directed edges indicate the precedence relationships. The precedence relationship graph for the example introduced in Section 3.1 is shown in Figure 3.5. Each of the involved bored pile walls creates one strand of four activities. Each strand starts by producing the piles, which is indicated by the nodes *B1* and *B2*. After the piles have been produced the first layers of excavation can be performed, represented by the nodes *E11* and *E12*. After that, the walls can be anchored by the activities *A1* and *A2*, allowing the activities *E21* and *E22* to finish the excavation. Eventually, after the areas *E3* and *E4* have been excavated, the foundation can be produced in activity *FND*. The box on the lower right scales the durations of the individual activities which are indicated by the width of the individual boxes. Activity *B1* has a duration of 4 time units, *E11* and *E12* have a duration of 5 time units, all other activities have a duration of 6 time units.

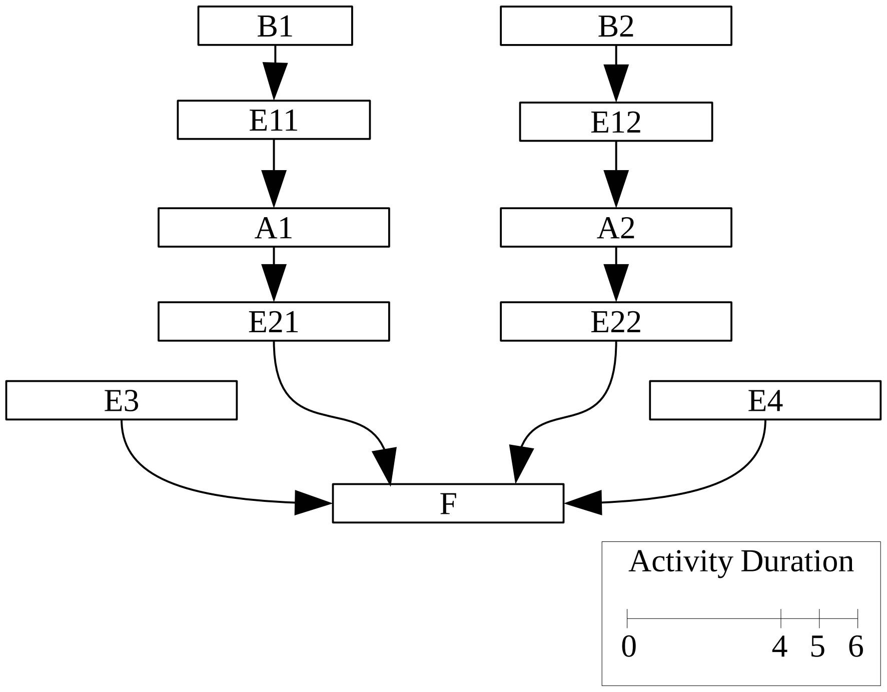


Figure 3.5: Bored pile wall project example precedence constraints. The two bored pile walls (B1 and B2) need to be finished before the first layer of excavation (E11 and E12), after which the pile walls can be anchored (A1 and A2) followed by the second layer of excavation (E21 and E22). Furthermore excavation activities E3 and E4 need to be performed. When all excavation is finished the foundation (F) can be produced.

The shortest possible schedule for this problem is shown in Figure 3.6 as a Gantt diagram. A Gantt diagram is a graphical representation of a schedule of activities over time, as defined by Gantt (1903). The horizontal axis describes the time while the vertical axis aligns the activities. Each line corresponds to exactly one activity. The arrows between the activities illustrate the satisfied precedence constraints. It is visible that both strands are executed in parallel as the precedence constraints in the example do not prevent that.

3.3.3 Critical Path Method (CPM)

In order to generate solutions for project scheduling problems Kelley & Walker (1961) described an approach, the Critical Path Method (CPM), which, while in itself not being an optimization procedure, formed a basis for many other algorithms. The critical path method is a two-pass approach consisting of a forward and a backward pass. In each pass start and finish times for each activity are calculated. During the forward pass the algorithm starts by selecting all start activities, which have no predecessors. In the case of the example shown in Figure 3.5 these are *B1*,*B2*,*E3*, and *E4*. These activities are

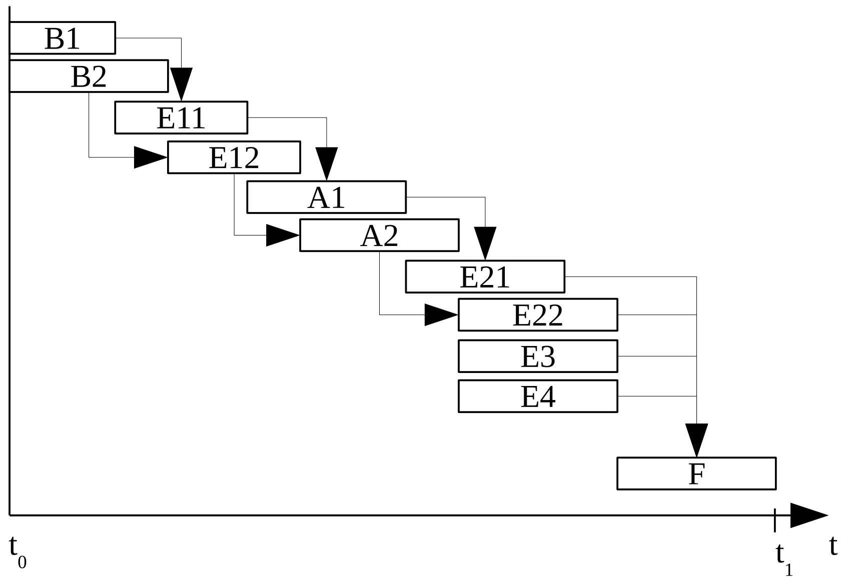


Figure 3.6: Bored pile wall project example solution.

assigned an early start time t_{ES} of 0, and the early finish time t_{EF} is set to the activity's duration added to its early start time. Those times are written on the upper left and upper right of the activity's boxes respectively. After that, all activities of which all predecessors have been treated are selected and the maximum early finish time of all predecessors is assigned as the early start time of each activity. The early finish time is assigned by adding the duration. Figure 3.7 shows the example after the forward pass has been executed.

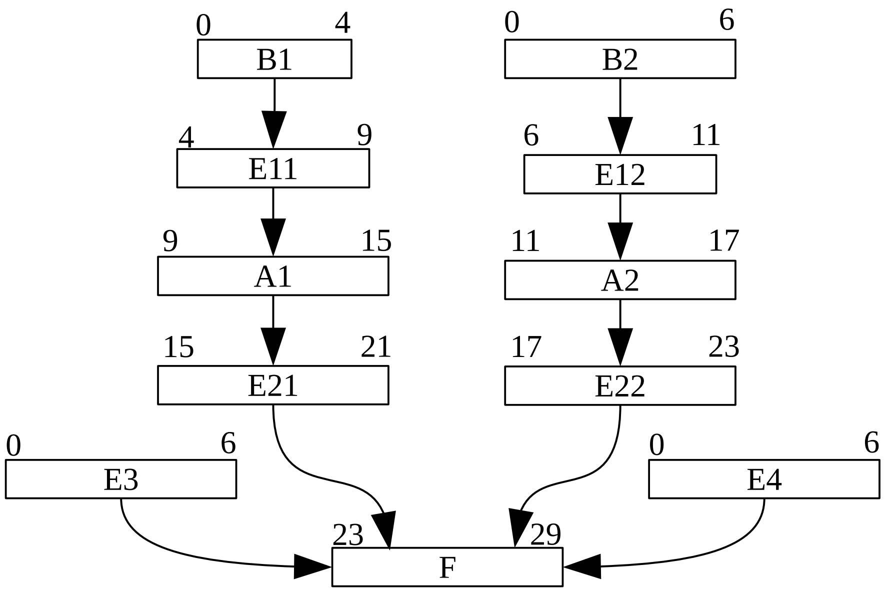


Figure 3.7: Bored pile wall project Example after the forward pass of the critical path method.

After the forward pass is completed, each activity has an early start and an early finish time assigned. The second stage of the algorithm consists of the backward pass. For this pass all precedence relationships are considered in reverse order. The latest of all early finish times is considered to be project horizon. Therefore the latest activities, that have no successors are selected initially and are assigned the latest early finish time as their late finish times t_{LF}. Their late start times t_{LS} are calculated by subtracting the respective duration from the late finish time. After that, all activities of which all successors have been treated are selected and their late finish times are assigned to be the earliest late start time of all successors. Their late start times are again calculated using the duration of the activities. The late start and late finish times are written on the lower right and lower left of the activities' boxes respectively. This results in the state illustrated in Figure 3.8.

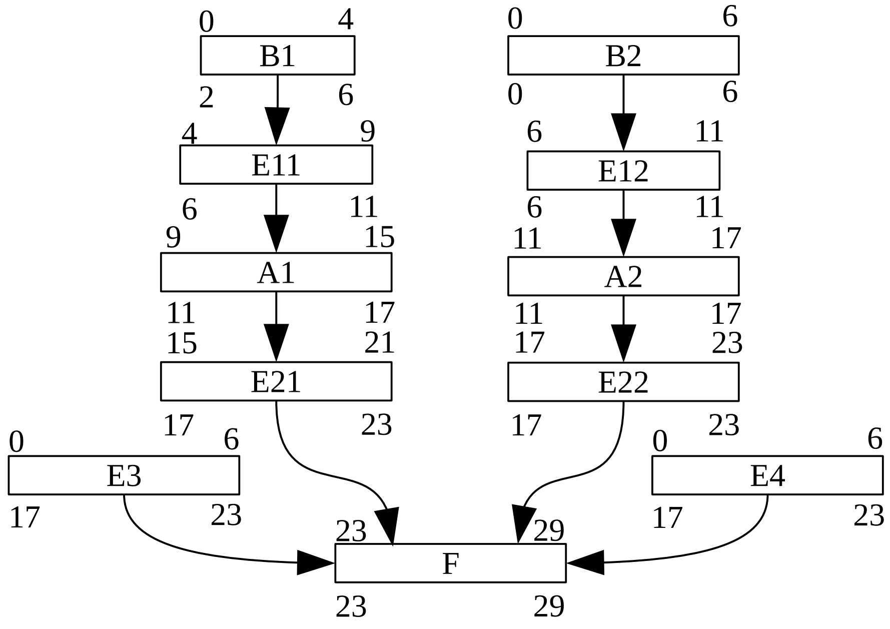


Figure 3.8: Bored pile wall project example after the backward pass of the critical path method.

Eventually the total float time, or the slack, s_i can be calculated for each activity i. This is done by subtracting the early start time and the duration from the late finish time. The resulting value represents the amount of time the activity can be shifted or delayed without changing the overall project duration. Table 3.2 shows the results of the algorithm for the example. The bottom row shows the total float times for each activity. Each activity that has a total float time of 0 is considered critical. This means that any delay in those activities propagates through the entire schedule. The set of all activities with zero total float time forms one or more critical paths. In this example the critical path is *B2* $\rightarrow$ *E12* $\rightarrow$ *A2* $\rightarrow$ *E22* $\rightarrow$ *F*.

The resulting schedule is visualized in Figure 3.9. Each activity is either blue, which indicates it is critical, or has a gray bar attached, indicating the total float time or the range

	B1	E11	A1	E21	B2	E12	A2	E22	E3	E4	F
t_{ES}	0	4	9	15	0	6	11	17	0	0	23
t_{EF}	4	9	15	21	6	11	17	23	6	6	29
t_{LS}	2	6	11	17	0	6	11	17	17	17	23
t_{LF}	6	11	17	23	6	11	17	23	23	23	29
s_i	2	2	2	2	0	0	0	0	17	17	0

Table 3.2: Results of the critical path method for the example in Figure 3.5.

the activity can be shifted in. As the excavation activities *E3* and *E4* are only required for the final foundation activity *F* and do not depend on any other activities to be finished they can be executed anytime before the foundation activity. Also, since the first bored pile wall *B1* is built much faster than the second one, as it contains fewer piles, it, and all the follow up activities *E11*, *A1*, and *E21*, that are preceding *F*, have non-zero float time, and hence are not critical.

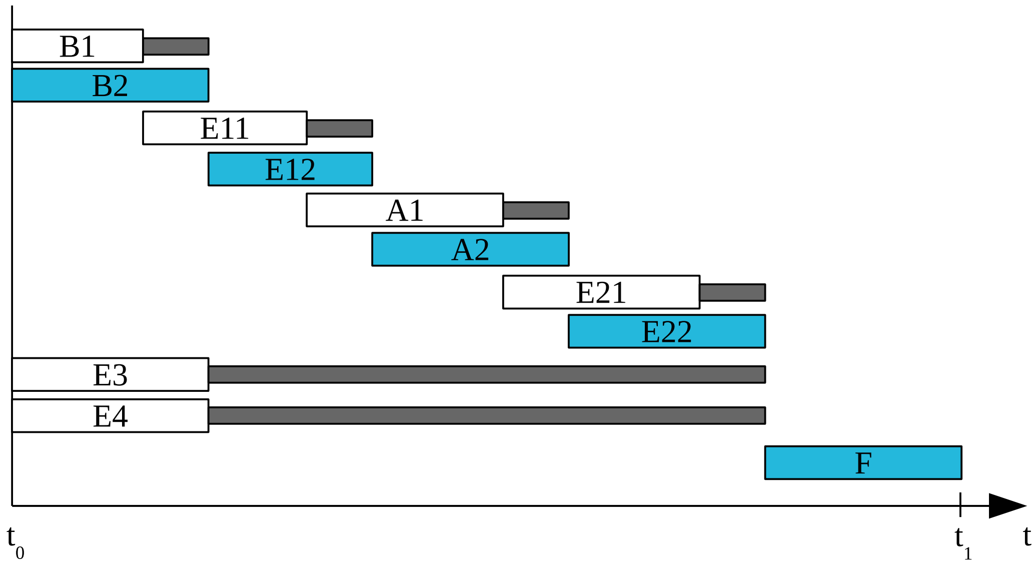


Figure 3.9: Visualization of critical path and total float times for the example in Figure 3.5. The gray bars indicate total float times, while the blue activities are critical.

The critical path method is an exact procedure that generates a feasible schedule for project scheduling problems with precedence constraints. It does however not consider resource sparsity, which is a commonly encountered constraint in real-life scheduling problems. While the project scheduling problem can be exactly solved using the described methods, it does neglect the need for resources. Most construction activities require machines, workers, or geometric spaces. These demands add strong constraints in practical scenarios, which need to be taken into account. Dori (2016) addressed the determination of float times and the critical path, taking resource constraints into account, which is discussed in detail in Section 3.4.6.

3.3.4 Program Evaluation and Review Technique (PERT)

The Program evaluation and review technique (PERT) was developed by the US Navy in order to forecast the research and development progress in large projects involving a simple representation of uncertainty (Malcolm et al., 1959). Compared to the Critical Path Method, PERT aims to find a schedule taking the expected duration of activities into account. The method is based on modeling each involved activity by a set of parameters. Those parameters include three time metrics which model the uncertainty of the activity durations, the optimistic duration t_O, the most-likely duration t_M, and the pessimistic duration t_P. From these three metrics, a fourth metric, the expected duration t_E is calculated using Equation 3.5, which is based on assuming that activity durations are beta distributed as defined in Section 2.2.1 and t_O and t_P are the limits of the feasible range of the distribution.

$$t_E = \frac{t_O + 4t_M + t_P}{6} \tag{3.5}$$

Furthermore each activity can have precedence constraints as defined in Section 3.3.1. All input data is usually represented in tabular form. Such a representation, listing precedence constraints and the four duration metrics for each activity is shown in Table 3.3.

Activity	Predecessors	t_O	t_M	t_P	t_E
B1		3	4	6	4.17
B2		5	6	8	6.17
E11	B1	4	5	6	5
E12	B2	4	5	6	5
A1	E11	5	6	9	6.33
A2	E12	5	6	9	6.33
E21	A1	4	6	9	6.17
E22	A2	4	6	9	6.17
E3		4	6	9	6.17
E4		4	6	9	6.17
F	E21,E22,E3,E4	5	6	7	6

Table 3.3: Tabular representation of the example in Figure 3.5 with added uncertainty.

From the data shown in the table, five time metrics are calculated, the early start time t_{ES}, the late start time t_{LS}, the early finish time t_{EF}, the late finish time t_{LF}, and the slack time t_S. The early finish time t_{EF} of an activity is calculated by adding the expected duration t_E to its early start time t_{ES}. The early start time is the latest early finish time t_{EF} of all its predecessors. The latest finish time t_{LF} of an activity is the smallest of all late start times t_{LS} of all its successors. Activities without successors have a late finish time t_{LF} equal to the largest early finish t_{EF} time of all activities. The late start times t_{LS} of the activities are calculated by subtracting the expected duration t_E from the late finish time t_{LF}. The slack time t_S is calculated by subtracting the early start time t_{ES} from the late start time t_{LS} of each activity. The calculation of the early start times t_{ES} can be viewed as calculating the longest predecessor path towards this activity, while the calculation of the late finish time t_{LF} can be viewed as calculating the longest path along the successors starting from the last activity. An activity with a slack time of zero is considered critical.

Activity	t_{ES}	t_{EF}	t_{LS}	t_{LF}	t_S
B1	0	4.17	2	6.17	2
B2	0	6.17	0	6.17	0
E11	4.17	9.17	6.17	11.17	2
E12	6.17	11.17	6.17	11.17	0
A1	9.17	15.5	11	17.5	2
A2	11.17	17.5	11.17	17.5	0
E21	15.5	21.67	17.5	23.67	2
E22	17.5	23.67	17.5	23.67	0
E3	0	6.17	17.5	23.67	11.33
E4	0	6.17	17.5	23.67	11.33
F	23.67	29.67	23.67	29.67	0

Table 3.4: Tabular representation of the results of applying PERT on the data from Table 3.3.

The results are either displayed in tabular form, as shown in Table 3.4, or in a so-called PERT chart, which can be represented as an Activities-on-the-Node (AoN) or an Activities-on-the-Arrows (AoA) network. For the AoN representation, nodes are commonly illustrated as is shown in Figure 3.10. An AoN representation of the anchored bored pile wall project example is shown in Figure 3.11.

t_{ES}	t_E	t_{EF}
Activity name		
t_{LS}	t_S	t_{LF}

Figure 3.10: Representation of an activity in the PERT chart notation.

It becomes visible that the path from *B2* towards *F* is critical, while all other activities have non zero slack times. PERT is an early approach to managing large projects and, by formalizing the defined metrics, as well as the PERT chart, offered new tools to project managers. Furthermore through the incorporation of uncertainty, it allowed assessment of the risk of delays, and therefore allowed to minimize costs. The calculation of the slack time also helped to identify critical activities. Resource sparsity however, was in the context of large scale military projects, not in the focus of the developers.

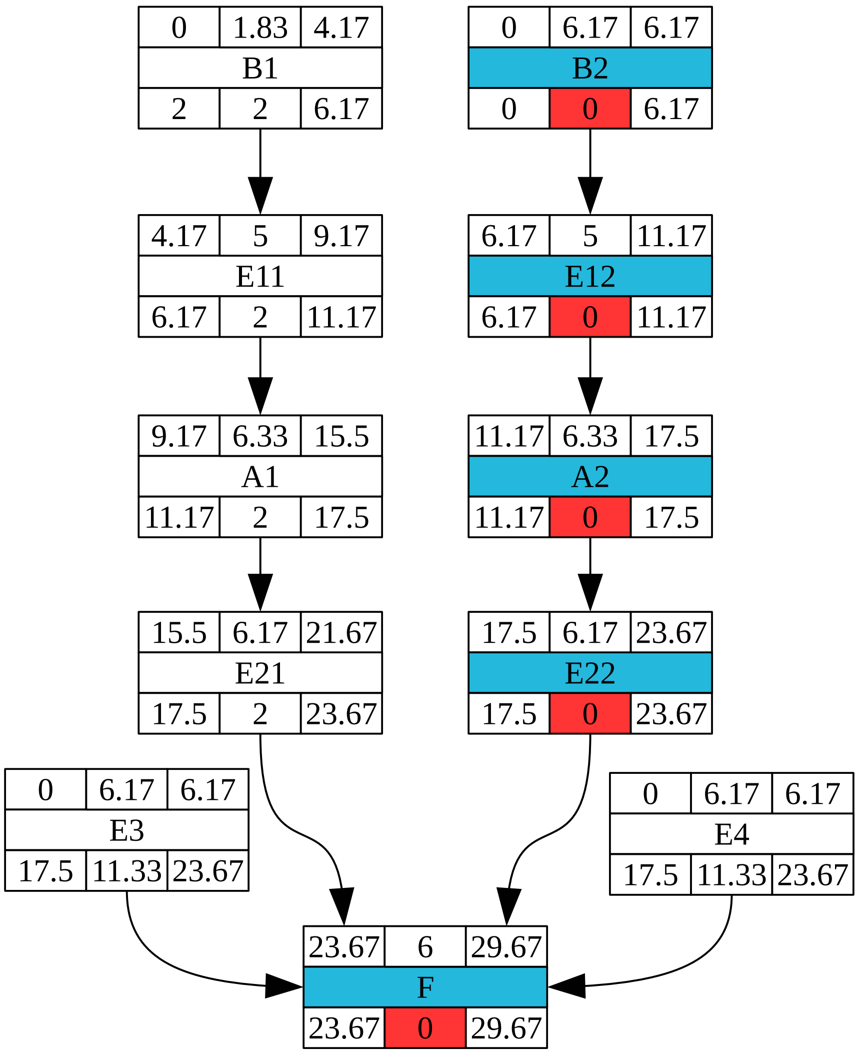


Figure 3.11: Anchored bored pile wall project example solution in PERT notation. The critical path, consisting of all activities with zero float time (*B2* → *E12* → *A2* → *E22* → *F*), is highlighted in blue. The corresponding Gantt Chart is shown in Figure 3.9.

3.4 Resource Constrained Project Scheduling Problem

Extending the project scheduling problem by resource demands, creates the Resource-Constrained Project Scheduling Problem (RCPSP). The object of the RCPSP is to schedule a given set of activities in an order that fulfills all given precedence and resource constraints while minimizing the overall makespan of the project. A formal definition of the problem is given in Section 3.4.7. The precedence constraints, same as in the project scheduling problem, may be represented as an directed acyclic graph, that defines a partial order of the activities. Each activity additionally may use a number of resources of different types. There is a limited number of instances of each resource type. As this problem is known to be NP-hard in the strong sense (Blazewicz et al., 1983), several heuristic algorithms have been developed, which are presented in Section 3.4.8. However, RCPSPs are still subject to many ongoing research endeavors.

3.4.1 Graphical Representation

There are different ways of representing the entirety of a resource constrained project scheduling problem. Due to the different possible cases, different notations are found throughout literature. When the anchored bored pile wall example from Figure 3.1 is extended by resources, a possible outcome is shown in Figure 3.12 (Davis, 1973).

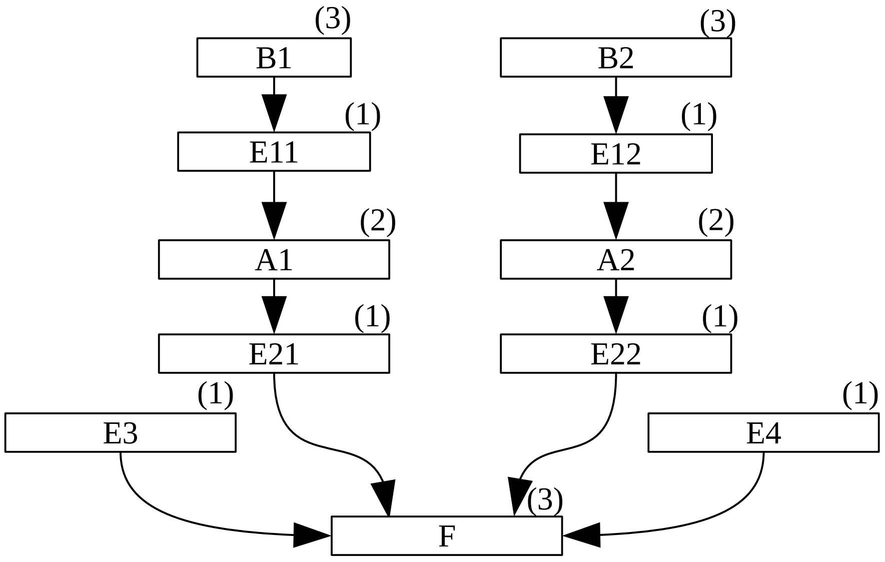


Figure 3.12: Bored pile wall project example with precedence and resource constraints.

On the upper right of each of the activities' boxes there is a vector of the number of resources of each type that is required for the activity to be executed. For simplicity only one type of resource, a worker, is considered. For the bored pile wall activities *B1* and *B2*, three workers are required. Each bored pile is created by drilling into the ground by connecting parts of a metal casing to each other, to go deeper into the ground. One worker is required for operating the drilling rig, one for operating the crane, delivering the casing parts, and one to guide the casing to connect to the previous part. The excavation activities only require one worker for operating the excavator. The anchoring process requires two workers for operating and guiding the anchoring machine. The foundation process requires three workers, adding reinforcement material and concrete.

Another, more intuitive way of representing resources graphically, is the notation shown in Figure 3.13. In this notation resources are shown as rounded rectangles and dashed arrows towards these rectangles indicate resource requirements. The numbers on the arrows indicate the number of instances required. For the anchored bored pile wall project example, the worker resource is added on the top side of the figure and each activity indicates its requirements by a dashed arrow.

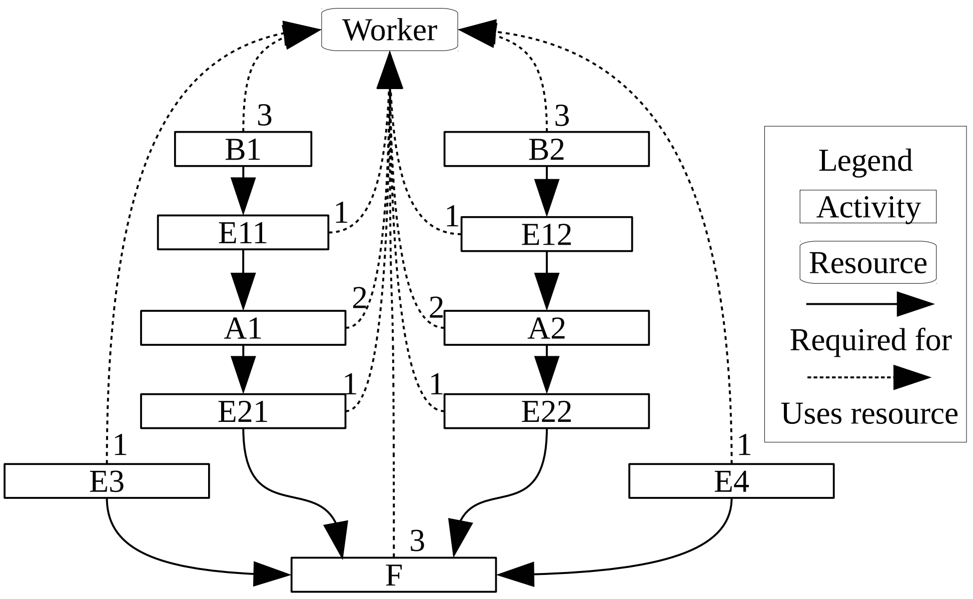


Figure 3.13: Bored pile wall project example precedence and resource constraints.

Additionally the activities' boxes can be colored to indicate the required resources, furthermore geometry and their overlaps can be indicated by a rhombus shape, and a hexagon indicates material demands. A general representation of such a problem is illustrated in Figure 3.14.

3.4.2 Problem Characteristics

In order to determine how hard a RCPSP is, prior to solving it, it is useful to analyze it. This Section presents commonly used metrics for characterizing RCPSPs. A number of indicators characterizing the structure of scheduling problems have been published in literature. Patterson (1976) defined a number of such indicators and analyzed their impact on the performance of (meta-)heuristic algorithms. The most commonly used indicator among which is the *resource constrainedness*. At the same time Cooper (1976) defined the *resource factor*, as well as the *resource strength*. The *resource constrainedness* is defined as the ratio γ_k of the average of the quantities r_{ik} of resource k required by activity i,

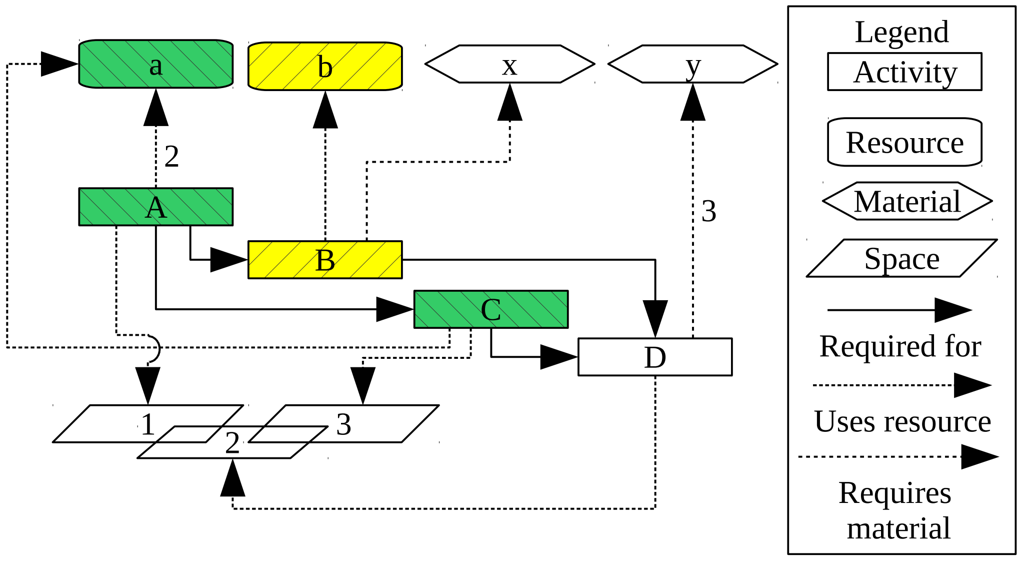


Figure 3.14: Scheme of a Resource Constrained Project Scheduling Problem (RCPSP).

requiring at least one unit, as defined in Equations 3.6 and 3.7, and the total number of units available of that resource r_k, resulting in Equation 3.8 (Patterson, 1976).

$$\rho_{ik} = \begin{cases} 1 & \text{if } r_{ik} > 0 \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in \boldsymbol{R}, \forall i \in \boldsymbol{J} \tag{3.6}$$

$$\phi_k = \frac{\sum\limits_{i \in \boldsymbol{J}} r_{ik}}{\sum\limits_{i \in \boldsymbol{J}} \rho_{ik}} \quad \forall k \in \boldsymbol{R} \tag{3.7}$$

$$\gamma_k = \frac{\phi_k}{r_k} \quad \forall k \in \boldsymbol{R} \tag{3.8}$$

The *resource factor* λ_i, as defined in Equation 3.9 by Cooper (1976), is the ratio of the average number of resources required for each activity and the total number of resource types $|\boldsymbol{R}|$.

$$\lambda_i = \frac{\sum\limits_{i \in \boldsymbol{J}} \sum\limits_{k \in \boldsymbol{R}} \rho_{ik}}{N \cdot |\boldsymbol{R}|} \tag{3.9}$$

The ratio of the number of resources available of type k, r_k and the average number required by all activities Φ_k, as defined in Equation 3.10 is called the *resource strength* ξ_k. It is calculated as is shown in Equation 3.11 (Cooper, 1976).

$$\Phi_k = \frac{\sum\limits_{i \in \boldsymbol{J}} r_{ik}}{N} \quad \forall k \in \boldsymbol{R} \tag{3.10}$$

$$\xi_k = \frac{r_k}{\Phi_k} = \frac{r_k \cdot N}{\sum\limits_{i \in \boldsymbol{J}} r_{ik}} \quad \forall k \in \boldsymbol{R} \tag{3.11}$$

For a more detailed overview of problem characteristics the reader is referred to the papers by Patterson (1976) and Cooper (1976).

3.4.3 The Anchored Bored Pile Wall Project Example

It is now assumed that only 4 workers are available, which for the schedule illustrated in Figure 3.6 results in the resource profile shown in Figure 3.15. The vertical axis indicates the number of workers required, while the horizontal axis represents the time. The red line marks the resource limit of four workers. Such a resource profile is created by stacking all activities that require the analyzed resource on top of each other. In this case the activities *E11*, *A1*, and *E21* are split into two boxes, to prevent gaps in the profile. It is visible that this resource limit is exceeded by the parallel execution of the bored pile wall activities *B1* and *B2* on the very left.

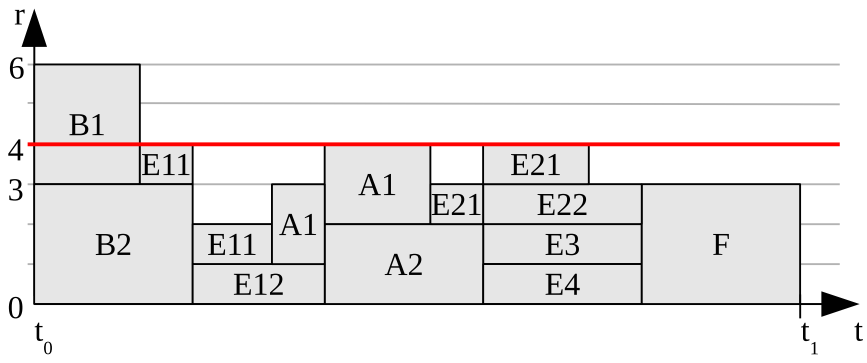


Figure 3.15: Bored pile wall project example resource usage.

Therefore, in order to not exceed the four worker resource limit, the activities *B1* and *B2* cannot be executed in parallel. A schedule preventing that is illustrated in Figure 3.16. Activity *B1* is executed first, followed by activity *B2*. As illustrated in Figure 3.17, the resource limit is not exceeded at any time throughout the schedule.

However, if instead of *B1*, the other bored pile wall *B2* would be produced first, this would result in the schedule shown in Figure 3.18, which also does not exceed the resource limit, as shown in Figure 3.19. The new schedule is more effective than the previous one, shown in Figure 3.16, as the duration is shorter. Also it is visible in the resource profile, that the idle times of workers are reduced. This illustrates how choices in the orders of activities result in different schedules with different makespans. Finding the order of activities minimizing some objective is in the focus of this Section.

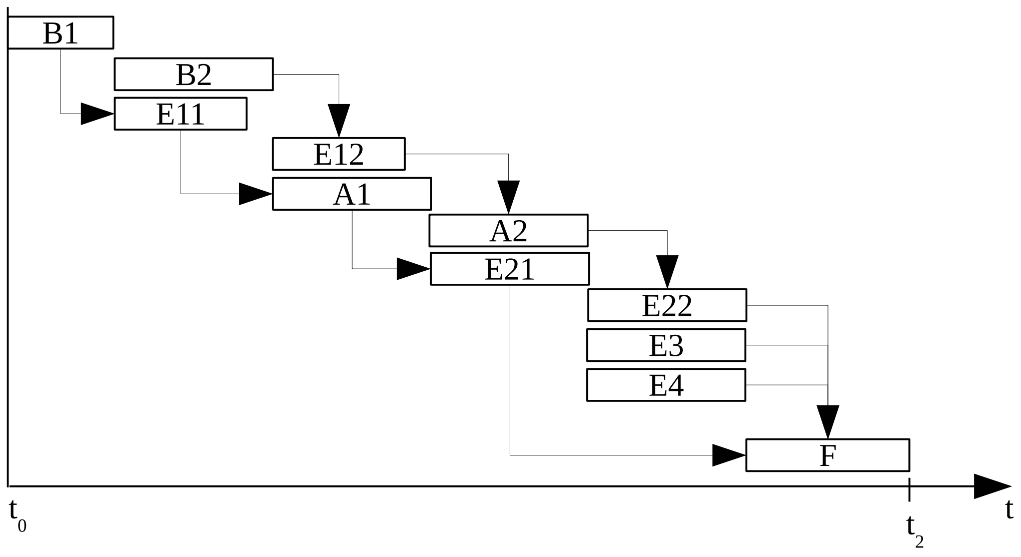


Figure 3.16: Bored pile wall project example solution with resource constraints.

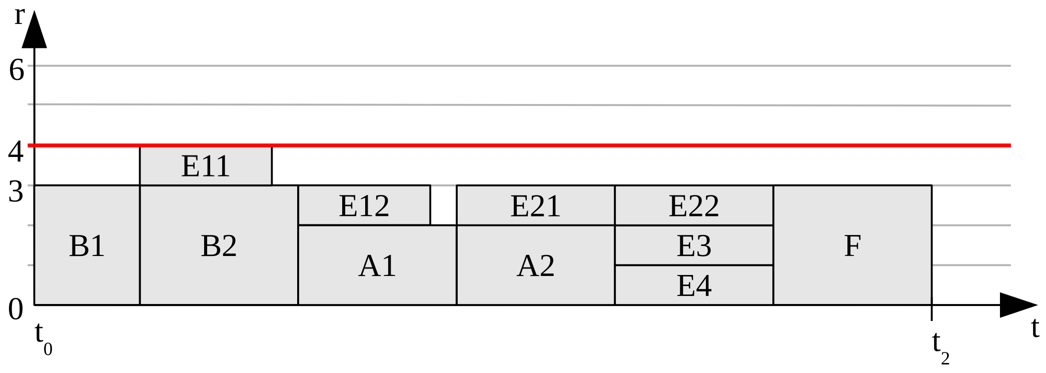


Figure 3.17: Bored pile wall project example resource usage.

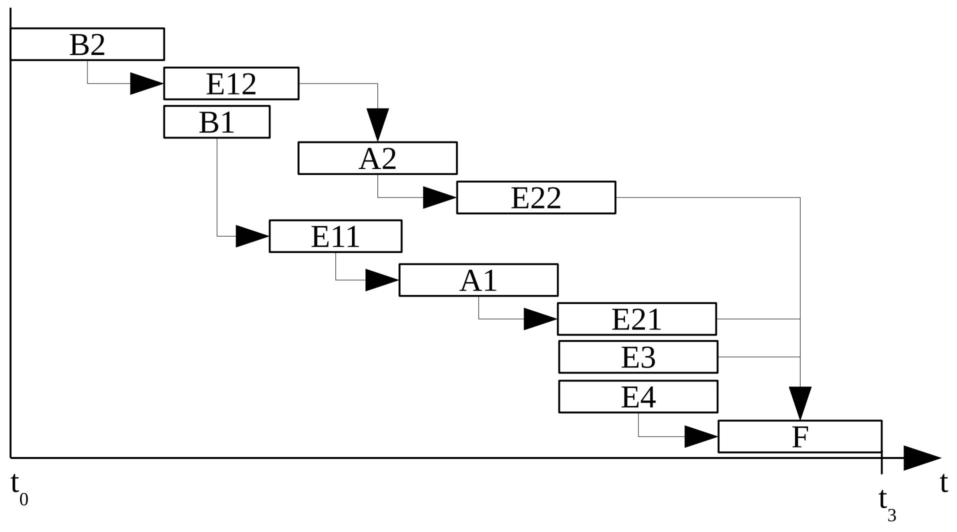


Figure 3.18: Bored pile wall project example solution with resource constraints.

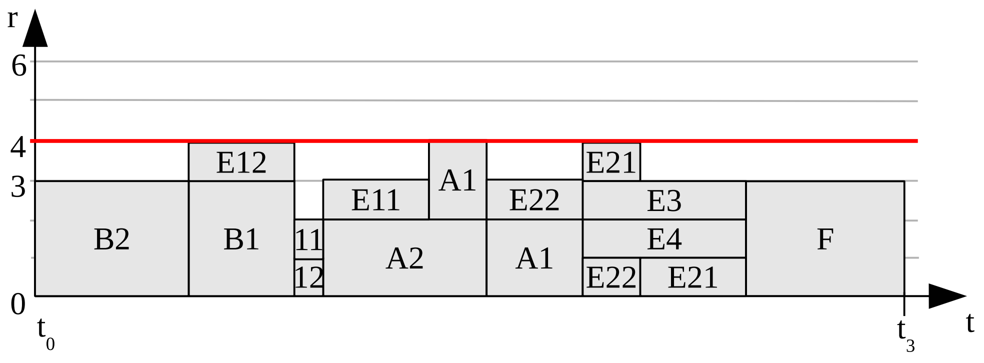


Figure 3.19: Bored pile wall project example resource usage.

3.4.4 Schedule Generation Schemes

A schedule generation scheme is a formal way of generating a feasible schedule for a given RCPSP. As many optimization procedures operate on the priorities of the individual activities, schedule generation schemes are commonly used to generate schedules from those priorities in a deterministic way. There are different schemes creating schedules with different properties. The most commonly used schemes are the serial schedule generation scheme (SSGS) and the parallel schedule generation scheme (PSGS). These schemes are described in detail in the following Sections. To overcome their specific limitations, a novel pseudo-serial schedule generation schema was developed in the frame of this thesis which is presented in Section 4.1. In order to discuss the differences between the schedules generated by the individual schemes, the different types of schedules are introduced.

3.4.4.1 Types of schedules

Sprecher et al. (1995) described four different types of schedules, feasible, semi-active, active, and non-delay schedules[2]. In order to illustrate each of the schedule types, the anchored bored pile wall project example has been modified, in order to clarify the differences. The resource limit was reduced to three workers, the duration of *E12* was reduced to four time units, and the anchoring activity *A2* requires one additional worker. To be able to understand the differences among the schedule types, two schedule operations have to be introduced, the local and the global left shift.

A local left shift continuously moves an activity to the left, to an earlier time, without at any time during the shift creating an unfeasible schedule. In Figure 3.20 shifting activity *E3* from time t_2 left towards t_1 illustrates such a local left shift. *E3* can be placed anywhere between the two time points without violating any constraints.

A global left shift can pick up any activity and move it to another position that is earlier in time. At the final position no constraint may be violated, but it is not required to only pass feasible solutions during the move. In the example moving activity *E4* from time t_4 to time t_3 illustrates a global left shift. It is feasible to execute the activity at both points in time, but it cannot be scheduled in between, as in combination with the activity *B2* it would exceed the resource limit.

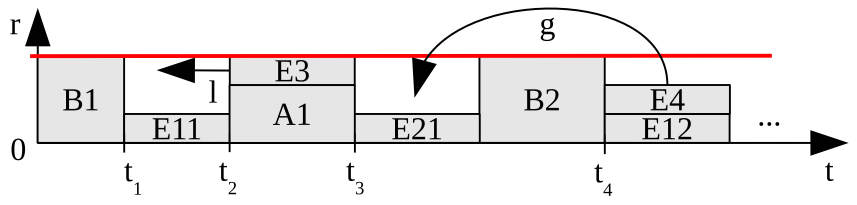


Figure 3.20: Illustration of global and local left shift operations. Left shifting E3 to t_1 is a local left shift (Labeled l), while shifting *E4* to t_3 is a global left shift (Labeled g).

[2]This Section is based on the description by Sprecher et al. (1995)

Figure 3.21 shows a feasible schedule for the problem. A feasible schedule is any schedule satisfying all precedence and resource constraints. As is shown in Figure 3.21 the schedule may contain gaps and it may therefore be possible to locally or globally left shift activities.

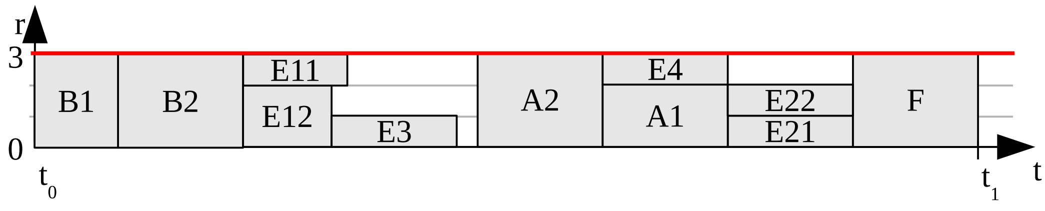


Figure 3.21: Feasible schedule for anchored bored pile wall project example.

A semi-active schedule is a schedule were none of the activities can be locally left shifted. As is illustrated in Figure 3.22 such a schedule may not contain any gaps, but it may be possible to globally left shift activities, such as activity *E4* can be moved to start right after activity *E12*. If no activity can be locally or globally left shifted , the schedule is called an active schedule. An active schedule is not necessarily an optimal schedule.

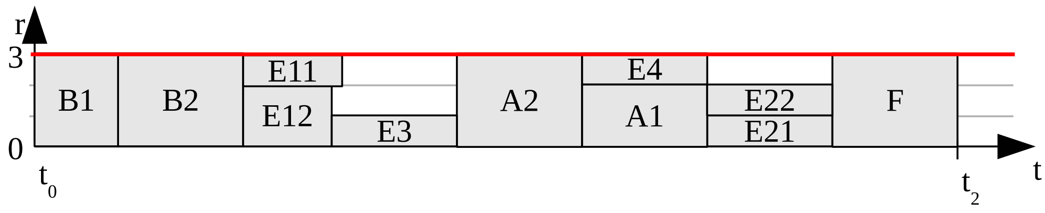


Figure 3.22: Semi-active schedule for anchored bored pile wall project example.

A non-delay schedule is a schedule where each activity is started at the earliest point where it becomes resource and precedence feasible. Such a schedule is shown in Figure 3.23. Compared to the schedule in Figure 3.22 the activity *E4* becomes precedence and resource feasible right after the activity *E12* finished and therefore has to be scheduled at that time. A way to verify whether a schedule is a non-delay schedule is to convert the schedule to a schedule for the corresponding unit-time-duration RCPSP (UTDRCPSP), where each activity has the exact same duration. For the schedule in Figure 3.23 this is shown in Figure 3.24. If the original schedule is a non-delay schedule, then the unit-time transformed schedule is an active schedule (Sprecher et al., 1995).

To show the shortcomings of non-delay schedules, Figure 3.25 shows an active schedule that is much shorter than the non-delay schedule in Figure 3.24. It is visible that there is a short gap right after activity *E12* is finished. At this point activities *E3* and *E4* are both precedence and resource feasible, and in a non-delay schedule have to be executed at that point, creating a suboptimal schedule. It is therefore important to not only consider non-delay schedules when searching for optimal schedules, as delaying an activity can

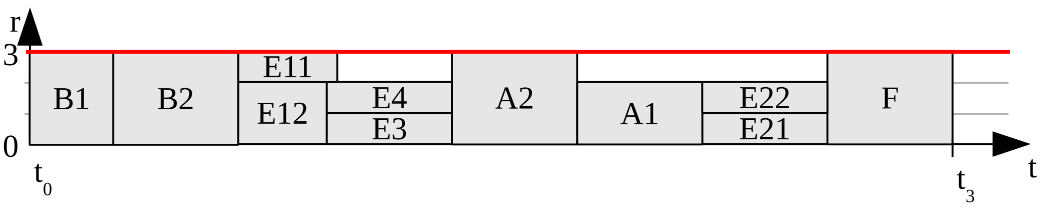


Figure 3.23: Non-delay schedule for anchored bored pile wall project example.

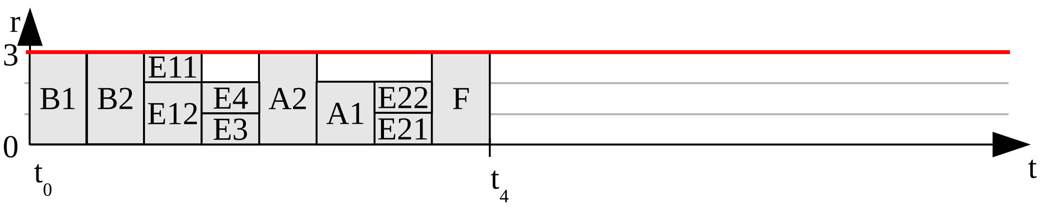


Figure 3.24: Unit time duration transformation of the schedule in Figure 3.23

yield an overall shorter schedule. Lemma 4 describes how the different schedule types are related in terms of set theory.

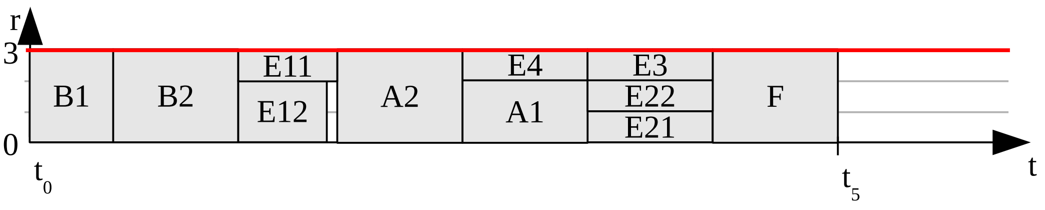


Figure 3.25: Active schedule for anchored bored pile wall project example.

Lemma 4. *Let $\boldsymbol{S}$ denote the set of all schedules, $\boldsymbol{S}_F$ the set of feasible schedules, $\boldsymbol{S}_{SA}$ the set of semi-active schedules, $\boldsymbol{S}_S$ the set of active schedules, and $\boldsymbol{S}_{ND}$ the set of non-delay schedules. Then following holds:*
$\boldsymbol{S}_{ND} \subseteq \boldsymbol{S}_A \subseteq \boldsymbol{S}_{SA} \subseteq \boldsymbol{S}_F \subseteq \boldsymbol{S}$ *(Sprecher et al., 1995).*

3.4.4.2 Serial Schedule Generation Schemes (SSGS)

For the purpose of generating schedules, different schedule generation schemes exist. To be able to explain these different scheduling schemes, priority values were assigned to each activity in the anchored bored pile wall project example as shown in Figure 3.26. The numbers on the right, inside each of the activities' box indicates the respective priority value. The number on the upper right of each box indicate the number of workers required for the activity. There is a total of three workers available. The priorities were assigned in an arbitrary way and form the basis for optimization procedures. Logically it makes sense to have the priorities respect the precedence relationships, because an activity can never be executed prior to its predecessors. The scheduling schemes select the lowest priority values first. This means that a lower priority value implies a higher priority. This choice will be motivated in Section 4.1.

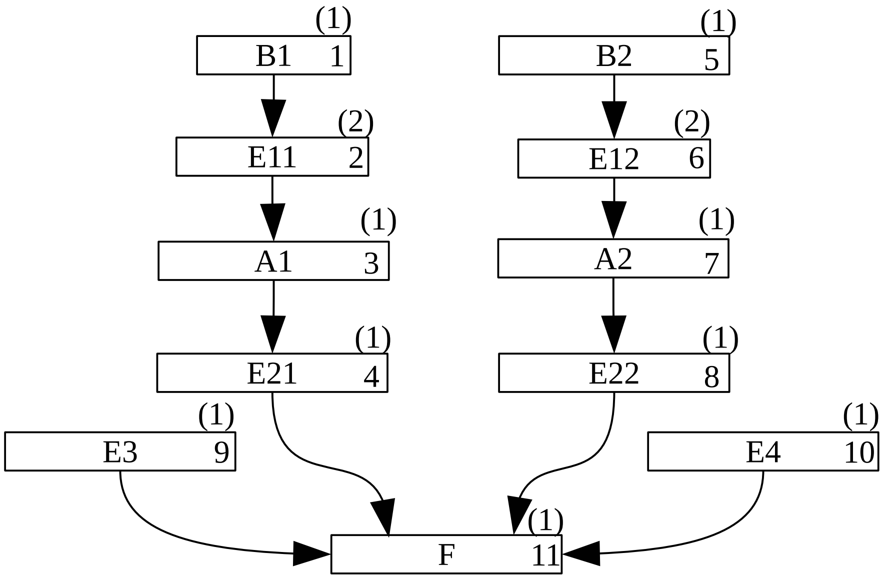


Figure 3.26: Priorities assigned to each activity of the anchored bored pile wall project example.

Most commonly the serial schedule generation scheme is used, because it can be used to represent the entire space of active schedules using priority rules (Kolisch, 1996b). A priority rule is based on a priority value assigned to each activity. Depending on the priority rule an activity is selected to be scheduled next. For simplicity all priority rules in this paper select the *lowest priority value first*. An overview of priority rules is presented in Section 3.4.8.1.

In the serial schedule generation scheme the priorities define an absolute order in which the activities are dealt with. The order of the priority values must not violate any precedence constraints. This means that no activity preceding another activity can have a higher priority value. Hence the preceding activity is always selected first. The schedule generation scheme then schedules the activities in the exact order of the priorities, respecting resource constraints. If resources are unavailable at the end time of the previous activity, it is moved forward until the activity blocking the resources has finished. The procedure is listed in Algorithm 1. The scheme is named *serial*, as only one activity is considered for scheduling at a time.

Lemma 5. *All active schedules can be generated by the serial schedule generation scheme (Kolisch, 1996b).*

Figure 3.27 shows a schedule generated for the problem with the priorities shown in Figure 3.26. The bottom part of the figure shows the corresponding resource profile.

Algorithm 1 Serial Schedule Generation Scheme

```
1: procedure SSGS
   $\mathcal{E}(g)$: all activities which can be started precedence-feasible in stage $g$
   $\mathcal{F}(j)$: maximum finish time of all predecessors of activity $j$
   $L_i$: latest start time of activity $i$
2:     for $g = 0$ to $n + 1$ do
3:         Calculate the eligible set $\mathcal{E}(g)$
4:         Select $i \in \mathcal{E}(g)$ with lowest priority value
5:         Schedule $i$ at the earliest precedence- and resource-feasible start time $t \in$
   $[\mathcal{F}(i), L_i]$
```

(Pseudo code from Merkle et al. (2002))

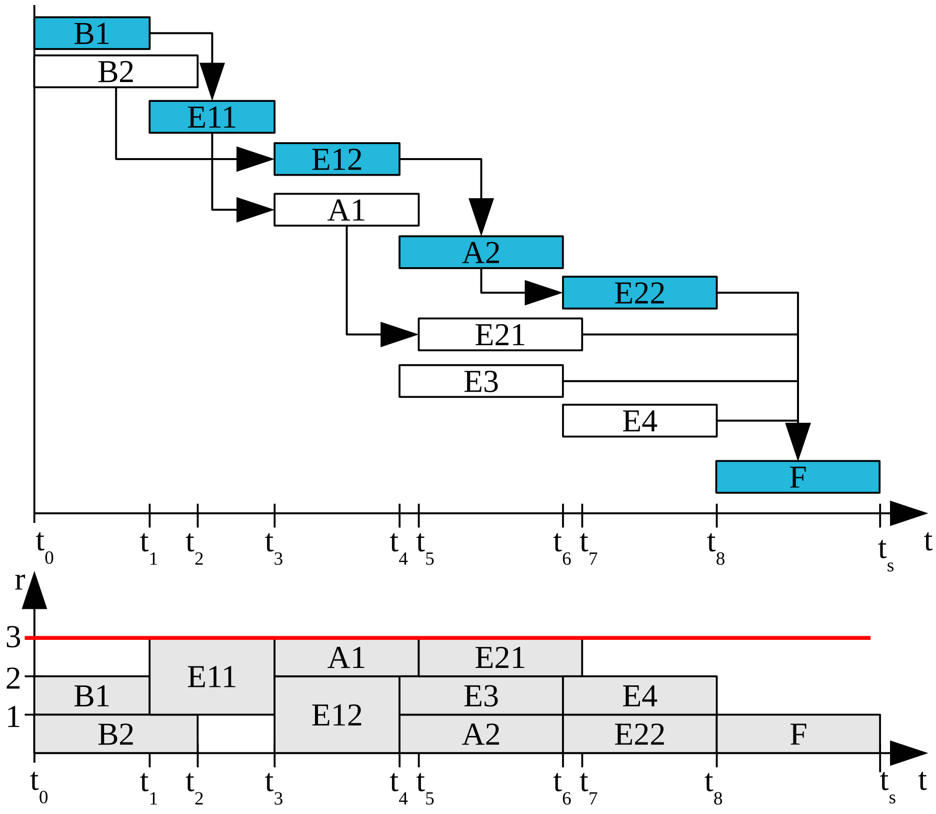


Figure 3.27: Schedule and resource profile created by the serial schedule generation scheme based on the priorities assigned in Figure 3.26. The critical path is highlighted in blue.

.

The schedule is generated by treating the activities in the exact order of the priorities as follows:

1. The activity *B1* has the lowest priority value and no predecessors. All three workers are available, therefore the activity is scheduled at time t_0.

2. *E11* has the lowest priority. it is only precedence feasible after *B1* has finished. It is scheduled right after *B1* at t_1.
3. *A1* requires *E11* to be finished. It is scheduled right after *E11* at t_3.
4. *E21* succeeds *A1* and is scheduled right after at t_5.
5. *B2* has lowest priority value of the remaining activities. Neither with *B1*, nor with *E11* does it exceed the resource limit of three workers. Therefore it can be scheduled at time t_0.
6. *E12* requires *B2* to be finished and exceeds the resource limit in combination with *E11*. It does however not exceed the limit togethet with *A1*. The earliest feasible start time is therefore directly after *E11* has finished at t_3.
7. *A2* required *E12* and is resource feasible in parallel with *A1* and *E21*. It can therefore be scheduled directly after *E12* at t_4.
8. *E22* succeeds *A2* and does not exceed the resource limit when combined with *E21*. It is scheduled directly after *A2* at t_6.
9. *E3* does not have predecessors. As it has a longer duration than *B1* it can not be scheduled at time 0. Also it is longer than *E12* and therefore does not fit parallel to that activity either. The earliest slot with one free worker is in parallel with *A2* at t_4.
10. *E4* can only be scheduled in parallel to *E22* at t_6.
11. As *F* required *E4* to be scheduled, the only possible time is in the end at t_8.
12. No more activities are left to be scheduled.

3.4.4.3 Parallel Schedule Generation Scheme (PSGS)

In contrary to the serial scheme, the parallel schedule generation scheme, also known as the Brooks Algorithm (BAG), which was first defined by Brooks & White (1965) and extended by Whitehouse (1979), generates schedules by time increments. At each discrete time instance, where a previously scheduled activity ends, a list of unscheduled activities with satisfied resource and precedence constraints is generated, similar to the constraint-based simulation approach by König & Beißert (2009), as described in Section 2.6.4. The next activity to be scheduled at this time is selected from this list by a priority rule. The scheduling procedure is listed in Algorithm 2.

Lemma 6. *All schedules generated by the parallel schedule generation scheme are non-delay schedules (Kolisch, 1996b).*

Figure 3.28 shows a schedule generated for the problem with the priorities shown in Figure 3.26. The bottom part of the figure shows the corresponding resource profile. The schedule is generated by discrete time increments and selecting currently feasible activities by choosing the activity with the lowest priority value:

Algorithm 2 Parallel Schedule Generation Scheme

```
1:  procedure PSGS
    $\boldsymbol{J}$: set of all activities
    $\mathcal{E}_r(g)$: all activities which can be started resource- and precedence-feasible in stage $g$
    $\boldsymbol{C}$: set of already scheduled activities
    $\boldsymbol{A}_g$: set of active activities
2:      $g := 0, t_g := 0, \boldsymbol{C} := \emptyset$
3:      while $|\boldsymbol{J} - \boldsymbol{C}| > 0$ do
4:          Calculate the eligible set $\mathcal{E}(t_g)$
5:          while $\mathcal{E}(t_g) \neq \emptyset$ do
6:              Select $i \in \mathcal{E}(t_g)$ with lowest priority value
7:              Schedule $i$ at $t_g$
8:              $\boldsymbol{C} := \boldsymbol{C} \cup \{i\}$
9:              recalculate $\mathcal{E}(t_g)$
10:         Calculate $\boldsymbol{A}_g$
11:         $g := g + 1$
12:         Calculate the minimal finish time $t_g$ of all activities in $\boldsymbol{A}_{g-1}$
```

(Pseudo code from Merkle et al. (2002))

1. The procedure starts at time t_0.
   At this point activities *B1, B2, E3*, and *E4* are precedence and resource feasible. *B1* has the lowest priority value and is therefore scheduled first.
   Now *B2, E3*, and *E4* are still feasible. Therefore *B2*, as it has lowest priority, is scheduled. Since *E3* and *E4* are still feasible, *E3* is scheduled next. Now no more activities are feasible as all resources are in use.

2. The next activity that finishes is *B1*. Therefore time is incremented to t_1.
   The only feasible activity at this time is *E4* which is scheduled next.

3. The next finish time is t_2.
   Now *E11* and *E12* are feasible. As *E11* has a lower priority value, it is scheduled next. No more activities are feasible at this time.

4. Both *E4* and *E11* end at time t_3.
   Activities *A1* and *E12* are feasible now. As *A1* has a lower priority value, it is scheduled next. Since *E12* is still feasible, it is also scheduled.

5. Time is incremented to the end time of *E12*, t_4.
   The only feasible activity is *A2* which is scheduled.

6. *A1* ends at time t_5.
   E21 is the only feasible activity and is therefore scheduled.

7. *A2* ends at time t_6.
   E22 is now feasible and scheduled.

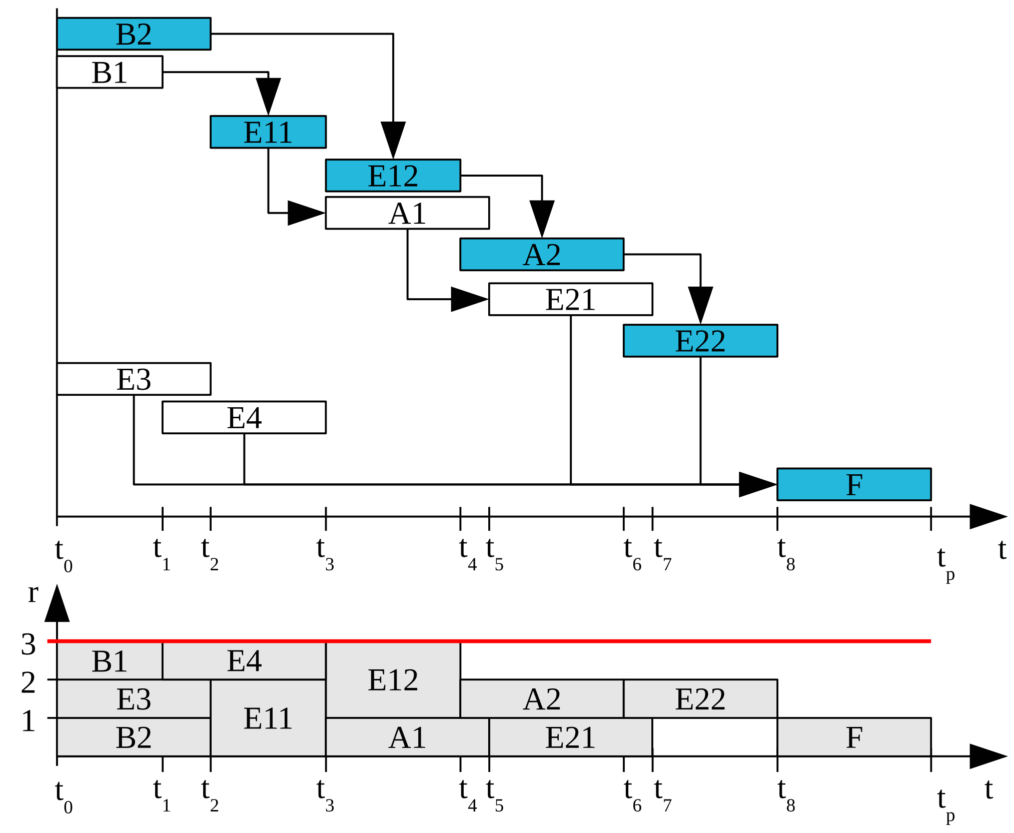


Figure 3.28: Schedule and resource profile created by the parallel schedule generation scheme based on the priorities assigned in Figure 3.26. The critical path is highlighted in blue.

.

8. *E21* finished at time t_7.
   The only remaining activity F is not yet precedence feasible.

9. *E22* finishes at time t_8.
   The last activity F is now feasible and scheduled.

10. F finishes at time t_p.
    No more activities are left.

3.4.4.4 Differences between the Schedule Generation Schemes

When comparing the results of both schedule generation schemes in Figures 3.27 and 3.28, it can be observed that the main difference lies in the gaps within the resource profile of the serial schedule between times t_0 and t_1, and t_2 and t_3. Those gaps cannot be created in the parallel scheme as activities *E3* and *E4* are used to fill the gaps. This also gives rise to the naming of the parallel scheme, as it maximizes parallel execution in a greedy way. Therefore the parallel schedule generation scheme can only create non-delay schedules, as is stated in Lemma 6. It can also be observed that both schedules form a different critical path, which is highlighted in blue in both schedules, and therefore have different makespans.

Generally the serial schedule generation scheme has higher computational demands because it needs to search for the earliest precedence- and resource-feasible start time for every activity, while the parallel scheme performs time increments. However, the run time complexity for both schemes is $\mathcal{O}(N^2 \cdot |\boldsymbol{R}|)$ (Kolisch & Hartmann, 1999; Pinson et al., 1994), since in the worst case each activity has to be started at an individual time instance. In practical cases, the runtime of the serial scheme is much higher, as is illustrated in Figure 3.29. It shows the simulation run time of both scheduling schemes for 1000 randomly generated problems consisting of up to 1000 activities.

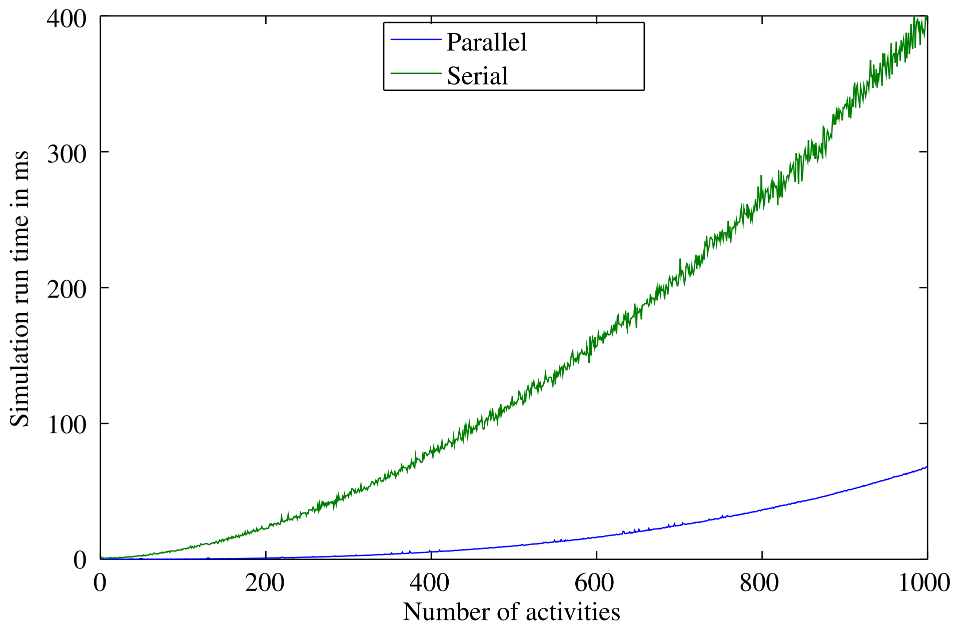


Figure 3.29: Comparison of performance of the serial and parallel schedule generation schemes with random problems with up to 1000 activities.

.

The author of this thesis has developed the new pseudo-serial schedule generation scheme, which is presented in Section 4.1. It combines the advantages of both the serial and the parallel schedule generation schemes.

3.4.5 Scheduling using Simulation

Simulation based scheduling techniques were initially applied to scheduling problems in the manufacturing industry (Jackson, 1957). The related problem is called the job shop scheduling problem, and it consists of a set of jobs and a set of machines which perform operations on jobs. Each job must go through a specific order of the machines in order to be completed. There are no precedence constraints among the operations of different jobs, operations cannot be interrupted, and each machine can only operate on one job at a time, as well as one job can only be operated on by one machine at a time. The objective is to find job sequences that minimize the makespan of all operations (Błażewicz et al., 1996). Same as the resource constrained project scheduling problem, the job shop scheduling problem is NP-hard (Lenstra & Rinnooy Kan, 1979). The main difference between the problems is, that the job shop scheduling problem involves a set of stationary machines where jobs pass through, whereas the RCPSP involves one large job consisting of many activities (operations) that need to be performed involving resources (machines). Furthermore, in the RCPSP, multiple resources can be required for a single activity. Kiran & Smith (1984) presented an overview of simulation based methods for job shop scheduling. For RCPSPs the simulation-based approaches merely use the simulation model to generate schedules based on predetermined parameters, such as priority values, that are set by a heuristic algorithm, such as those described in Section 3.4.8.

The first simulation model that was widely applied to construction operations was CYCLONE, which is based on the modeling approach presented in Section 2.5.4 (Halpin, 1977). Later STROBOSCOPE was developed, which is based on a similar modeling approach, which is presented in Section 2.5.5 (Martinez & Ioannou, 1994). The ABC approach of Shi (1999) was presented in Section 2.5.6. All those approaches focus on cyclic operations, such as earthworks, or the bored drill pine example. Using them as a model for an entire construction project causes the model to become highly complex.

Simulation based methods for RCPSPs only got into focus of research as sufficient computational power for simulating large construction projects became available. König et al. (2007) introduced the constraint-based simulation approach in order to create schedules for civil engineering projects and ship construction. The constraint-based simulation approach has strong similarity to the parallel schedule generation scheme presented in Section 3.4.4.3, but features some key differences. The main difference is that the next activity to be scheduled is selected randomly. Through multiple simulation runs in a Monte Carlo way (Section 2.2.6), well-performing schedules are generated. Additionally, incorporating soft-constraints allows to include experiences from similar projects, human strain factors, and preferences of the planners (Beißert et al., 2010). As these constraints do not strictly need to be satisfied, they are modeled as soft-constraints. The simulation concept is illustrated in Figure 3.30.

Additionally König's simulation concept involves productivity of workers and machines under different circumstances, that may influence the durations of activities, as for instance, having to perform an activity in a confined space may reduce efficiency. Furthermore the concept was extended by incorporating uncertainty in order to create schedules which maximize robustness (König, 2011a,b). Moreover the feedback of actual progress data from construction sites was enabled by Szczesny & König (2015). It involves updating a given schedule based on real-time data, and if considered necessary, create a new schedule using evolutionary algorithms in combination with König's simulation approach. Updating may be required due to extensive delays of the involved activities, which is discussed in more detail in Section 4.6.

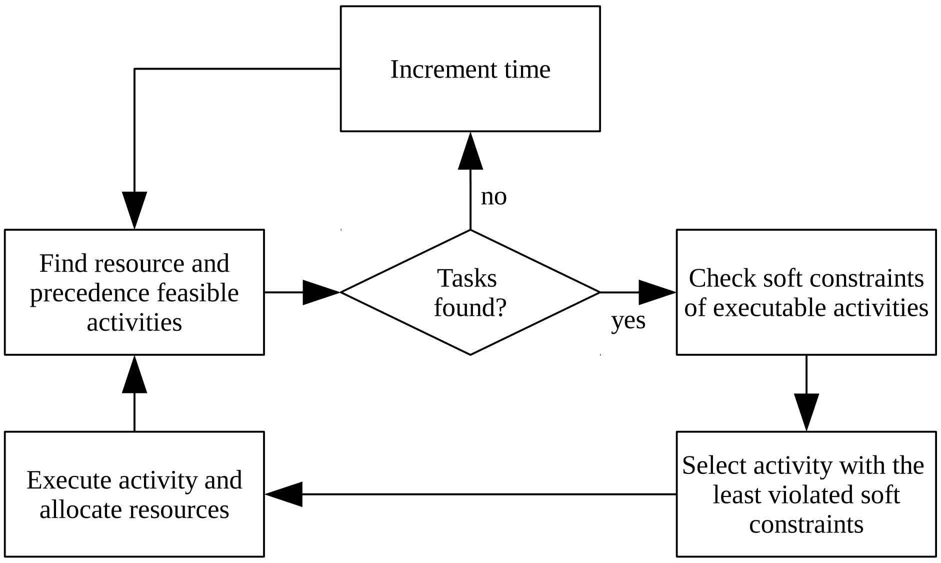

Increment time
no
Find resource and precedence feasible activities
Tasks found?
yes
Check soft constraints of executable activities
Execute activity and allocate resources
Select activity with the least violated soft constraints

Figure 3.30: Flowchart of simulation concept by Beißert et al. (2010).

3.4.6 Simulation-based Critical Path Analysis using the Dori-Algorithm

In order to know when a delay is causing a schedule update to be necessary it is useful to know both the critical path of the schedule and the float time of each activity as it is described in Section 3.3.3 about the critical path method (CPM).

The CPM however does not take resource limits into account. This limitation was addressed by Dori et al. (2012) and Dori (2016) who published a simulation based algorithm to determine both float times and the critical path of a given RCPSP schedule.

Similar to the critical path method, the Dori-Algorithm (DA) is a two pass procedure based on a forward and a backward simulation run. However, due to resource limitations a direct backward simulation run can yield a different order of activities using the same resources, and therefore result in incorrect float times. In order to fix this, additional constraints need to be introduced for the backward simulation run. These constraints are required for every set of activities that are not linked by precedence constraints, but in combination exceed the number of available resources. This is referred to as a Dori-Set (DS). A Dori-Set is a special case of a forbidden set, as defined by Schäffter (1997). It is to be ensured, that the backward simulation run does not create switches in the order of activities that cannot be scheduled simultaneously due to resource constraints.

As an example, the schedule for the anchored bored drill pine project example, generated by the serial schedule generation scheme, which is shown in Figure 3.27 is used. It is assumed that this is the result of the forward simulation run. The example involves only one type of resource and therefore the Dori-Sets can be determined easily. *B1* and *B2* are not precedence linked, but together use a total number of two resource instances. The next finishing independent activity is *E12* and the combination of those three activities exceed the number of available resources. The Dori-Algorithm now dictates an artificial sequence enforcement constraint to be added between *B1* and all direct successors of *E12*, which is only the activity *A2*. Then *E11* and *E12* form a Dori-Set and therefore a sequence enforcement constraint between *E11* and *A2* is required. Another Dori-Set is formed by *E12*, *A1*, and *E21*, which adds a sequence enforcement constraint between *E12* and *F*. All other Dori-Sets involve *F* which does not have successors. Therefore no other constraints need to be added. Figure 3.31 shows the schedule with the added constraints as thick red arrows. The result of the backward simulation run is depicted in Figure 3.32, and the resulting critical path and float times are shown in Figure 3.33. For a more in-depth description of the Dori-Algorithm the reader is referred to Dori (2016, pp 103–116).

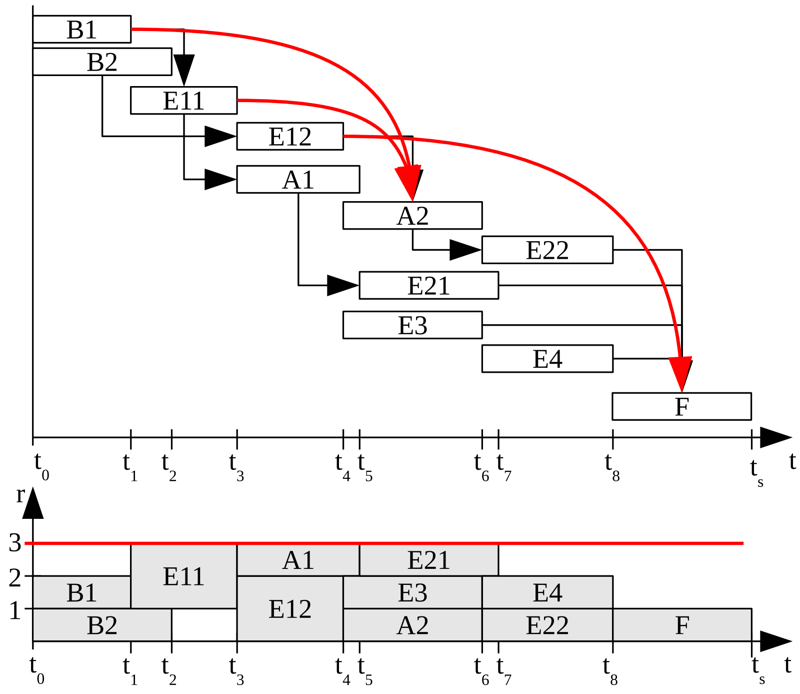

B1
B2
E11
E12
A1
A2
E22
E21
E3
E4
F
t0 t1 t2 t3 t4 t5 t6 t7 t8 ts t
r
3
2
1
B1
B2
E11
A1
E21
E12
E3
E4
A2
E22
F
t0 t1 t2 t3 t4 t5 t6 t7 t8 ts t

Figure 3.31: Schedule from Figure 3.27 with sequence enforcement constraints added by the Dori-Algorithm depicted as thick red arrows.

.

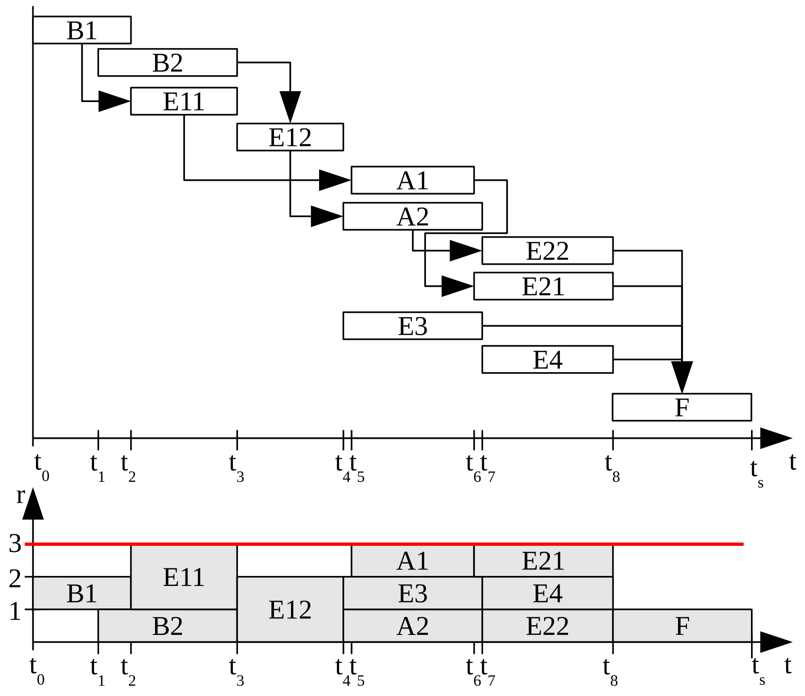

B1
B2
E11
E12
A1
A2
E22
E21
E3
E4
F
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_s t
r
3
2
1

Figure 3.32: Schedule resulting from backward simulation run according to the Dori-Algorithm based on the sequence enforcement constraints in Figure 3.31.

.

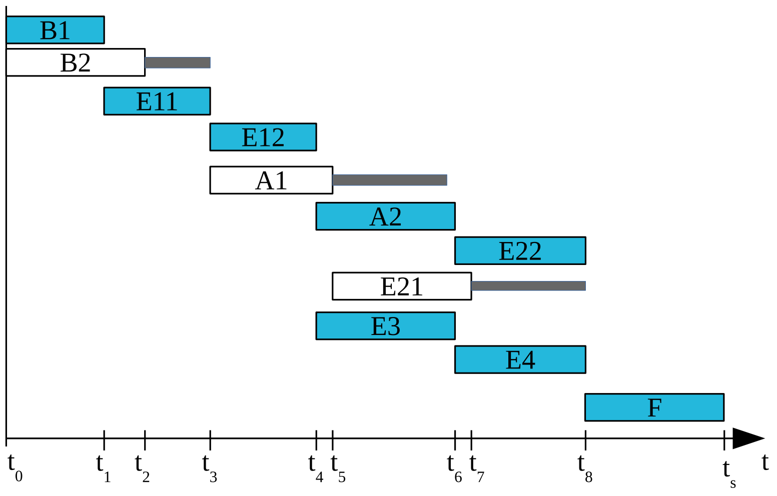

B1
B2
E11
E12
A1
A2
E22
E21
E3
E4
F
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_s t

Figure 3.33: Float times and critical path resulting from Dori-Algorithm for the schedule in Figure 3.31.

.

3.4.7 Deterministic Approaches to the Resource Constrained Project Scheduling Problem

Since the resource constrained project scheduling problem is NP-Hard in the strong sense (Blazewicz et al., 1983), deterministic approaches are hardly feasible for large real-life problems. However, for completeness, the theoretical approaches for deterministic RCPSP solutions are described within this Section[3]. Talbot & Patterson (1978) defined a conceptual linear program in order to solve the project scheduling problem without taking resource constraints into account. This linear program is shown in Equation 3.12. This definition is forms a basis for the definitions of the RCPSP and is therefore described at this point. It introduces two artificial activities called dummy activities 0 and $J+1$. Activity 0 is the start activity and it precedes all other activities while activity $J+1$ is the final activity which succeeds all other activities. Both dummy activities have a duration of 0. The objective of the linear program is to minimize the starting time F_{J+1} of the finish activity as is shown in Equation 3.12a. Equation 3.12b defines all precedence constraints $\boldsymbol{V}_{ij}$. It defines that the finish time of activity i, F_i must be smaller or equal to the finish time F_j of activity j, minus the duration d_j of activity j, given that there is a precedence constraint that defines activity j to be a successor of activity i. Equation 3.12c defines the start activity to finish at time 0, while Equation 3.12d defines the finish times to be larger or equal than 0 for the set of all activities $\boldsymbol{J}$. Equation 3.12e defines that all finish times must be smaller or equal to some predetermined project horizon, as for instance the sum of all activities' durations.

$$\begin{aligned}
\text{minimize} \quad & F_{J+1} && && (3.12a)\\
\text{subject to} \quad & F_i \leq F_j - d_j, && \forall \langle i,j\rangle \in \boldsymbol{V}_{ij} && (3.12b)\\
& F_0 = 0 && && (3.12c)\\
& F_i \geq 0, && \forall i \in \boldsymbol{J} && (3.12d)\\
& F_{J+1} \leq \bar{d} && && (3.12e)
\end{aligned}$$

Pritsker et al. (1969) however defined a different approach to solve PSPs using linear programs involving time indexing. They therefore introduced a binary variable x_{it} as defined in Equation 3.13 that is 1 if and only if activity i starts at time t. $\boldsymbol{E}_i$ is the set of possible start times of activity i. This set can be determined for each activity using the critical path method as described in Section 3.3.3. The sum defined in Equation 3.14 therefore results in the time F_i, where activity i finishes, as only one x_{it} can be 1 for each activity i, as is defined in the constraint defined in Equation 3.15b. The objective function is equal to the finishing time of the final dummy activity as defined in Equation 3.15a. Equation 3.15b defines the precedence constraints equally to Equation 3.12a after substitution according to Equation 3.14. Equation 3.15d defines that the initial dummy activity start at time 0 and Equation 3.15e defines that all x_{it} are binary variables.

[3]This Section was inspired by the description by Bauer (2009)

$$x_{it} = \begin{cases} 1 & \text{if } t = F_i \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

$$F_i = \sum_{t \in \boldsymbol{E}_i} t \cdot x_{it} \tag{3.14}$$

$$\begin{aligned}
\text{minimize} \quad & \sum_{t \in \boldsymbol{E}_{J+1}} t \cdot x_{J+1,t} && && (3.15a) \\
\text{subject to} \quad & \sum_{t \in \boldsymbol{E}_i} x_{it} = 1 && \forall i \in \boldsymbol{J} && (3.15b) \\
& \sum_{t \in \boldsymbol{E}_j} t \cdot x_{jt} - \sum_{t \in \boldsymbol{E}_i} t \cdot x_{it} \geq d_j && \forall \langle i, j \rangle \in \boldsymbol{V}_{ij} && (3.15c) \\
& x_{00} = 1 && && (3.15d) \\
& x_{it} \in \{0, 1\} && \forall i \in \boldsymbol{J}; \forall t \in \boldsymbol{E}_i && (3.15e)
\end{aligned}$$

This time indexed notation allows the definition of an additional constraint modeling resource limitations as defined in Equation 3.17d. $\boldsymbol{R}$ is the set of all resources and $\boldsymbol{T}$ is the set of all time periods. k_r is the limit of resource r and k_{ir} is the consumption of resource r for activity i *per time period*. Equation 3.16 defines k_{ir} as the overall demand u_{ir} of resource r by activity i divided by the duration d_i of activity i. $\boldsymbol{Q}_{it}$ is the set of time periods where activity i is in progress if it was started at time t. Therefore the constraint in Equation 3.17d prevents any resource limit to be exceeded at any time.

$$k_{ir} = \frac{u_{ir}}{d_i} \tag{3.16}$$

$$\begin{aligned}
\text{minimize} \quad & \sum_{t \in \boldsymbol{E}_{J+1}} t \cdot x_{J+1,t} && && (3.17a) \\
\text{subject to} \quad & \sum_{t \in \boldsymbol{E}_i} x_{it} = 1 && \forall i \in \boldsymbol{J} && (3.17b) \\
& \sum_{t \in \boldsymbol{E}_j} t \cdot x_{jt} - \sum_{t \in \boldsymbol{E}_i} t \cdot x_{it} \geq d_j && \forall \langle i, j \rangle \in \boldsymbol{V}_{ij} && (3.17c) \\
& \sum_{i \in \boldsymbol{J}} k_{ir} \sum_{\tau \in \boldsymbol{Q}_{it}} x_{i\tau} \leq k_r && \forall r \in \boldsymbol{R}; \forall t \in \boldsymbol{T} && (3.17d) \\
& x_{00} = 1 && && (3.17e) \\
& x_{it} \in \{0, 1\} && \forall i \in \boldsymbol{J}; \forall t \in \boldsymbol{E}_i && (3.17f)
\end{aligned}$$

When examining the linear program defined by Equation 3.17 several downsides become clear. For activities with very long durations the set $\boldsymbol{T}$ can become very large. This can be counteracted by setting the length of a time period to the greatest common divisor of all activities' durations. However, for instance two activities with durations 1 and 10000 would still require 10000 time periods to be defined. Each time period in $\boldsymbol{T}$ creates one constraint for each resource in $\boldsymbol{R}$ and each activity in $\boldsymbol{J}$ creates one variable x_{it} for each possible start time in E_i. This causes the linear program for larger problems to become extremely large. This illustrates why heuristic procedures are of high interest for research.

3.4.8 Heuristic Approaches to the Resource Constrained Project Scheduling Problem

As shown in the previous Section, deterministic solutions for large scale problems are hardly feasible, and therefore heuristic procedures are highly interesting for practical problems. This Section provides an overview of the state-of-the-art of heuristics for resource-constrained project scheduling problems[4].

3.4.8.1 Priority Rules

A priority rule assigns a value $v(i)$ to each activity i and states whether the minimum or maximum value is to be selected first (Kolisch & Hartmann, 1999). Additionally a way to handle ties needs to be defined. The priority rules can be used by either of the schedule generation schemes defined in Section 3.4.4 but could also be incorporated into the constraint-based discrete-event simulation approach by Beißert et al. (2007), which is described in Section 2.6.4. Özdamar & Ulusoy (1995) defines the difference between the scheduling schemes as the employment of either fixed a-priori priorities (serial) or dynamically updating priorities (parallel). In this thesis priorities are always assumed to be set a-priori, but for the serial schedule generation scheme need to be in precedence feasible order.

The priorities can either be assigned randomly and used as a basis for optimization procedures, or be calculated based on different criteria. The individual rules can be classified based on the information used to calculate the priority values $v(i)$, which can be network, time, and resources, as well as lower and upper bounds (Kolisch & Hartmann, 1999). For an overview of many different priority rules the reader is referred to the papers by Davis & Patterson (1975), Kolisch (1996a), and Hartmann & Kolisch (2000).

3.4.8.2 X-Pass Methods

X-Pass methods deploy different priority rules in combination with different schedule generation schemes on the problem. The simplest way are single pass methods which employ exactly one priority rule in combination with one schedule generation scheme. Multi-pass methods can be created by different combinations of schedule generation

[4]This Section was in part inspired by the review papers by Hartmann & Kolisch (2000); Kolisch & Hartmann (1999, 2006).

schemes and priority rules, or by forward-backward scheduling methods. Another category of x-pass methods are sampling methods which make use of randomly selected priority rules, randomly generated priority values, or priority rules based on selection probabilities. Among the most advanced versions is adaptive regret based biased random sampling (RBRS) as presented by Kolisch & Drexl (1996) and Schirmer (2000). The idea is to select the priority rule, schedule generation scheme and the sampling method based on problem characteristics, as described in Section 3.4.2, such as number of activities or resource constrainedness.

3.4.8.3 Tabu Search

Glover (1989, 1990a,b) introduced the method of tabu search which can be applied to resource constrained project scheduling methods. It is basically a steepest decent neighborhood search. Starting from an initial solution, neighboring solutions are analyzed for the best solutions. As it does not strictly require improving solutions, it is possible that the algorithm starts to cycle. To prevent this, Glover introduced a kind of memory, remembering previously encountered solutions to prevent going back to a recently visited solution. This memory can be depleting, such that only a certain number of previously visited solutions are stored in order to save memory and computational effort. An application of tabu search for resource constrained project scheduling was published by Baar et al. (1999).

3.4.8.4 Simulated Annealing

Kirkpatrick et al. (1983) introduced the algorithm called simulated annealing, combining combinatorial optimization and statistical mechanics into a new metaheuristic optimization algorithm. It is based on the process of the annealing of metals where free floating particles cool down and lock into a state of low energy. After an initial solution is generated, the neighborhood is analyzed for other solutions. If a better solution is found it is always accepted as the new current optimum. If no better solution is found, but a worse one, it is only accepted with a probability depending on the quality of the solution and the current temperature of the simulated cooling matter. The temperature T_t is lowered during the execution of the algorithm until it is cooled down, according to a cooling scheme. The probability $P(y|x)$ of accepting a new solution y, given the current solution x is defined in Equation 3.18, where $f(x)$ is the fitness of solution x.

$$P(y|x) = \begin{cases} 1 & \text{if } f(y) \leq f(x) \\ \exp(-\frac{f(y)-f(x)}{T_t}) & \text{if } f(y) > f(x) \end{cases} \tag{3.18}$$

Simulated annealing was applied to resource constrained project scheduling problems by Józefowska et al. (2001), Bouleimen & Lecocq (2003), Valls et al. (2005), and König & Beißert (2009).

3.4.8.5 Genetic Algorithms

Holland (1975) introduced a new nature inspired metaheuristic algorithm which was inspired by adaption and evolution in biology. While the simulated annealing algorithm works on improving a single solution, a genetic algorithm works on a population of solutions which are improved using a set of operators which either modify (mutate) a member of the population or mate (crossover) two members of the population. The population is usually kept at a certain number of members. When through modification that number is exceeded, the weakest members of the population die out. Weakness in this context is commonly determined by a fitness function. In the scheduling domain, the weakest solutions would be those having the longest makespan. Genetic algorithms were applied to resource constrained project scheduling problems by Hartmann (1998), Toklu (2002), Valls & Ballestín (2002), Senouci & Eldin (2004), Debels & Vanhoucke (2005), Debels et al. (2006), Valls et al. (2008), Long & Ohsato (2009), König (2011a), and Ghoddousi et al. (2013).

3.4.8.6 Ant Colony Optimization

Dorigo et al. (1996) introduced the ant colony optimization algorithm as a new metaheuristic. The idea of the algorithm is to simulate a population of agents driven by a greedy force. Through the interaction of the agents by so-called pheromones, the performance of the algorithm was improved significantly over the performance of a single agent. The algorithm was adapted to resource-constrained project scheduling problems by Merkle et al. (2002). They used the serial schedule generation scheme and pheromones reflect the expected efficiency of putting certain activities in specific places using the latest start time priority rule, where activities are ordered by their latest start time according to the critical path method described in Section 3.3.3.

3.4.8.7 Local Search Procedures

Local search is a procedure that starts from random points in the search space and tries to find improved solutions by local modifications of the solutions. In the case of scheduling this is usually done by modifying the priority values. These methods include procedures such as the previously introduced simulated annealing and tabu search methods. Simulated annealing is a stochastic local search procedure as defined by Hoos & Stützle (2004). Fleszar & Hindi (2004) use a variable version of neighborhood search applied to RCPSPs based on the serial schedule generation scheme. Valls et al. (2003) uses a topological order representation of the schedules encountered, combining local search with tabu search, while also using the serial schedule generation scheme.

3.4.8.8 Swap-based Local Search

Another possibility for local search algorithms is the use of swaps between two activities, that are precedence feasible, modifying an existing schedule (Bügler & Borrmann, 2014; Bügler et al., 2013; Dori, 2016). A swap is commonly performed by interchanging the priorities of two activities. A special kind of swap, a cyclical shift, was used in the simulated annealing algorithm by Bouleimen & Lecocq (2003). The effect of a normal swap differs depending on the schedule generation scheme that is used. The schedule generation schemes were introduced in Section 3.4.4. In order to illustrate the effects of swapping activities a simplified version of the bored pile wall project example is introduced in Figure 3.34. It features two bored pile walls *B1* and *B2*, both requiring a drilling rig, which is colored in yellow. Furthermore two excavation activities *E1* and *E2* need to be scheduled, requiring an excavator, which is colored in green. Finally a foundation can be build, which is not assigned any resources in this example. There is only one drilling rig and one excavator available, hence neither *B1* and *B2*, nor *E1* and *E2* can be executed in parallel.

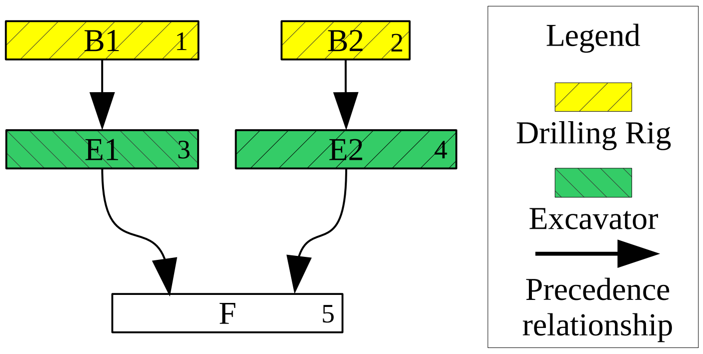


Figure 3.34: Simplified bored pile wall project example with priorities on the right inside of each activity's box.

Figure 3.35 shows an example schedule for the simplified bored pile wall project example in Figure 3.34. The schedule could have been create by either of the two schedule generation schemes based on the priorities defined a-priori.

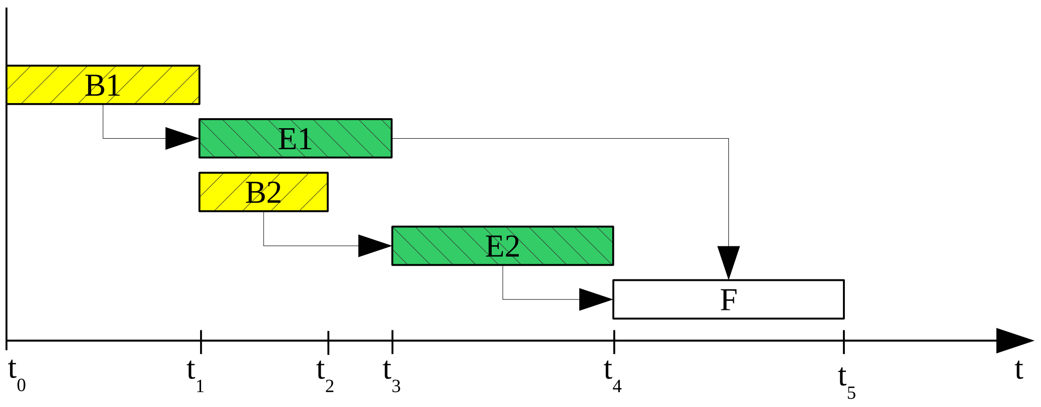


Figure 3.35: Example schedule for simplified bored pile wall project example in Figure 3.34.

Figure 3.36 shows the schedule after *E1* and *E2* have been swapped. This swap however only has an effect using the serial schedule generation scheme. When activity *B1* ends, *E1* immediately becomes resource and precedence feasible and therefore is scheduled at this very time instance. In the serial scheme however, *E2* is scheduled at the earliest resource and precedence feasible time, right after *B2* finished, leaving insufficient room for *E1* and therefore pushing it to start after *E2*. This creates a longer schedule as the initial one.

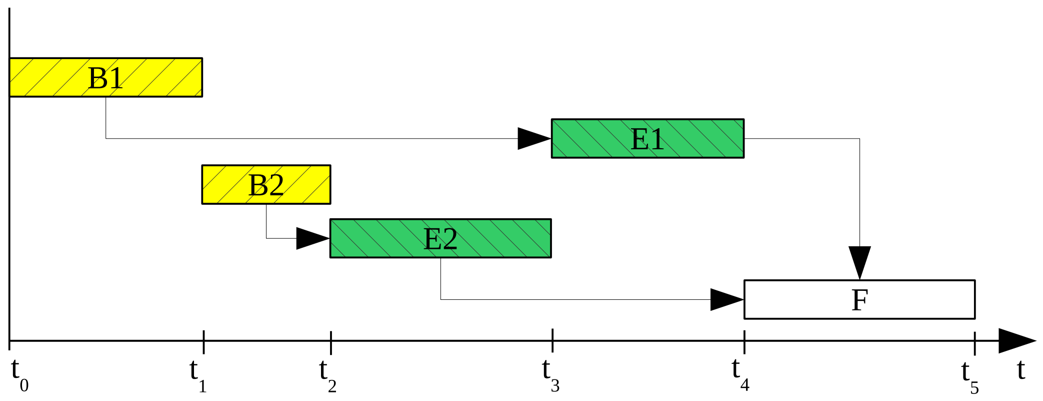


Figure 3.36: Example schedule from Figure 3.35 after swapping *E1* and *E2* using the serial schedule generation scheme.

When an additional swap is performed, switching the priorities of *B1* and *B2*, this results in the schedule shown in Figure 3.37. This schedule would, in this priority configuration be equal for both scheduling schemes. The resulting schedule is shorter than the initial one and is actually an optimal solution for the problem.

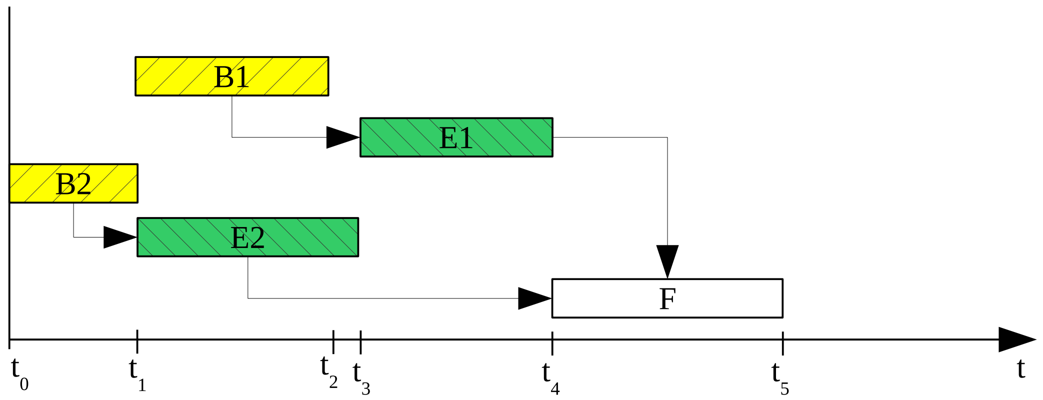


Figure 3.37: Schedule from Figure 3.35 after additionally swapping *B1* and *B2*.

When swapping *B1* and *B2* first, however, using the serial scheduling scheme, a longer schedule is created, similar to the one in Figure 3.36. This is shown in Figure 3.38.

When on the other hand the parallel scheme is used for the exact same swap, the schedule collapses into the optimal solution right away, as is shown in Figure 3.39. This is due to the property of the parallel schedule generation scheme stating that it only creates

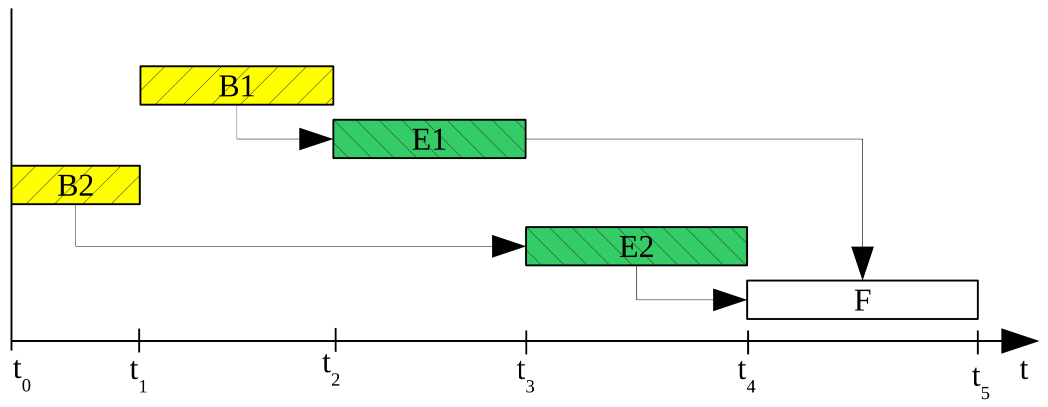


Figure 3.38: Example schedule from Figure 3.34 after swapping *B1* and *B2* using the serial schedule generation scheme.

non-delay schedules and therefore minimizes idle times in a greedy manner. This however is not always optimal as is shown in the following example.

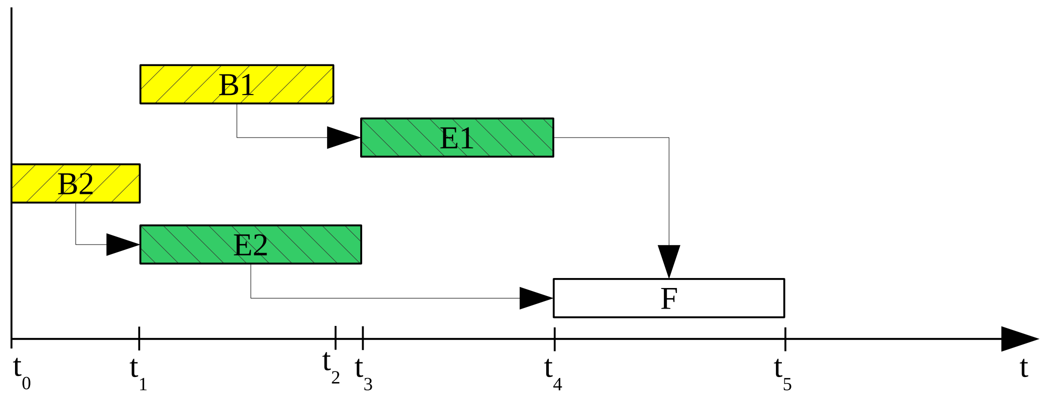


Figure 3.39: Example schedule from Figure 3.34 after swapping *B1* and *B2* using the parallel schedule generation scheme.

When viewing another modification of the example, where instead of the second bored pile wall, a sheet pile wall is created, and therefore a piledriver is required for the activity *P1*. The modified problem, with activity priorities, is shown in Figure 3.40.

A schedule, resulting from the priorities in Figure 3.40, is shown in Figure 3.41. This schedule would result from both scheduling schemes, given the assigned priorities.

However, wehen *E1* and *E2* are swapped, this does not have any effect on the result of the parallel schedule generation scheme, as *E2* is the only precedence and resource feasible activity at t_1, when *P1* is finished. In the serial schedule generation scheme on the other hand, *E1* will be scheduled at time t_2, right after *B1* has finished, and in the result create a much shorter schedule, as shown in Figure 3.42. Due to the unnecessary idle time of the excavator between times t_1 and t_2, this is not a non-delay schedule and therefore can in no way be generated by the parallel schedule generation scheme.

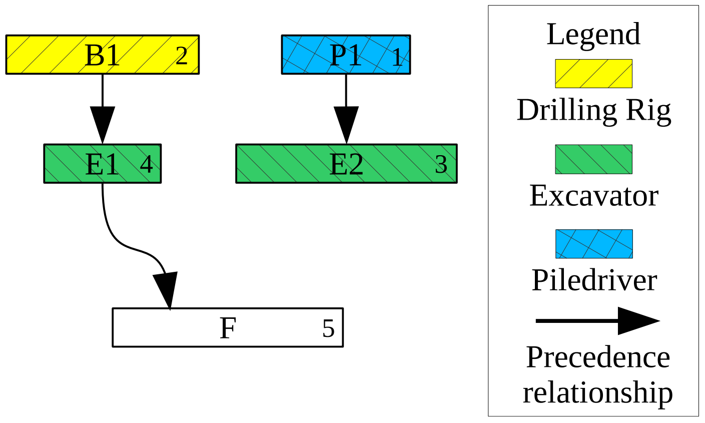


Figure 3.40: Simplified bored pile wall and sheet pile wall project example with priorities on the right inside of each activity's box.

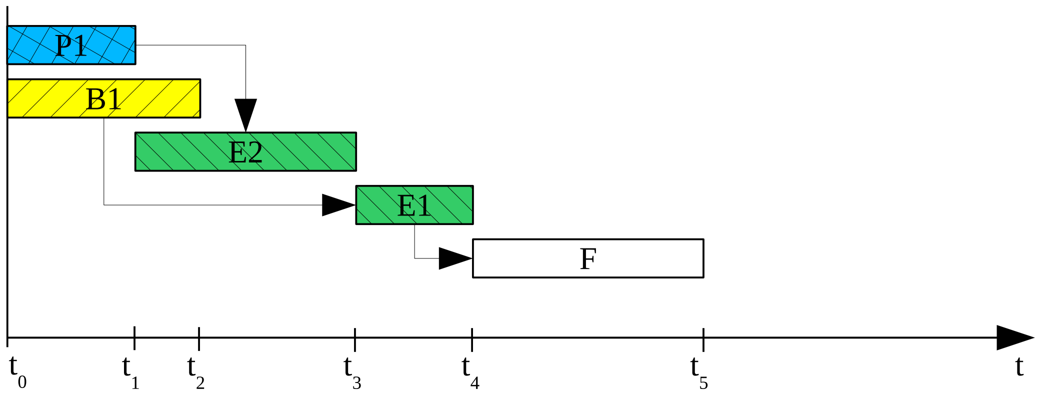


Figure 3.41: Schedule for the example in Figure 3.40.

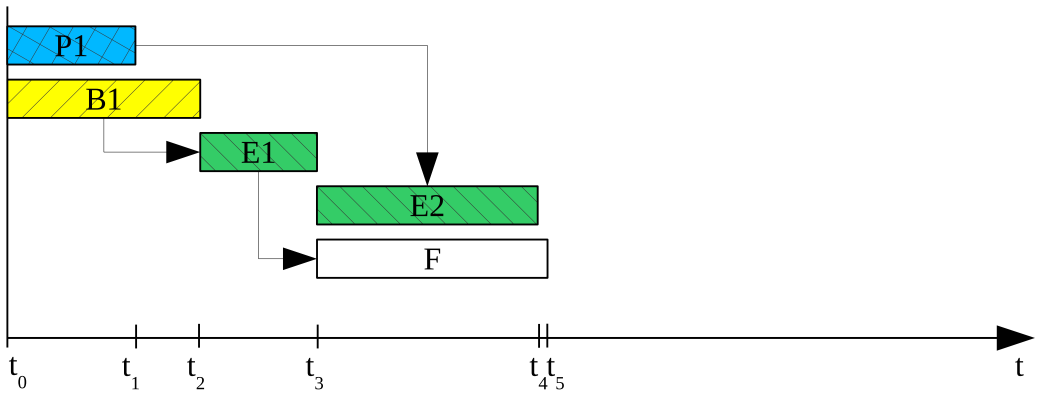


Figure 3.42: Schedule for the example in Figure 3.40 after swapping *E1* and *E2*, when using the serial schedule generation scheme.

Lemma 7. *Every active schedule can be reached from any other schedule by applying a* *series of swaps using the serial schedule generation scheme.*

Lemma 8. *Every non-delay schedule can be reached from any other schedule by applying a series of swaps using the parallel schedule generation scheme.*

Dori (2016) defined the notion of *reasonable swaps*, as swaps between activities which share the usage of a common resource, and cause an actual change in the resulting schedule. Whether a swap results in a different schedule depends on the scheduling scheme that is used. As the work of Dori (2016) builds up on the constraint-based simulation by König et al. (2007), the definition of *reasonable swaps* is based on the parallel schedule generation scheme as defined in Section 3.4.4.3.

The parallel schedule generation scheme allows the collection of such swaps during schedule generation with minimal effort. Every time during the schedule generation process, the eligible set $\mathcal{E}(g)$ is updated, after an activity was scheduled. When due to the scheduled activity, activities previously in the eligible set, were removed, this must be caused by resources shared by the executed activity and the removed one, as the precedence feasibility cannot have changed at this point. Therefore any swap between the scheduled activity and any subsequently removed activity is *reasonable* according to the definition by Dori (2016).

The notion of *reasonable swaps* can be used as a basis for several metaheuristic algorithms. Dori (2016) calculated the total possible number of swaps Ψ as defined in Equation 3.19 based on the number of activities N. This results in a total possible number of resulting schedules $|\mathcal{S}|$ as defined in Equation 3.20.

$$\Psi = \frac{N^2 - N}{2} \tag{3.19}$$

$$|\mathcal{S}| = \Psi! \cdot \sum_{i=0}^{\Psi} \frac{1}{i!} \tag{3.20}$$

It is obvious that the worst-case number of possible schedules is enormous. For a problem with $N = 10$ activities, a maximum of $\Psi = 45$ swaps is possible. This results in a maximum number of schedules $|\mathcal{S}| = 3.2517 \cdot 10^{56}$. In order to not have to explore all those possible schedules, Dori (2016) introduces two techniques for intelligently exploring only parts of the search space by utilizing *reasonable swaps*. First a simulated annealing algorithm, similar to the one proposed by Bouleimen & Lecocq (2003), which is described in Section 3.4.8.4, but instead of cyclical shifts, it is using *reasonable swaps*. Another approach is the creation of search trees by swaps as is illustrated in Figure 3.43. Each node in this tree represents one schedule by a vector of priorities. A swap is performed by interchanging priorities within this vector. As the set of reasonable swaps changes for each schedule, it has to be recalculated for each node. Dori (2016) proposed a greedy-like heuristic search on such a tree, by exploring the search tree by only investigating nodes improving the current solution, but having a certain fixed tolerance towards worse solutions in order to overcome local optima. This approach is similar to the simulated annealing approach, but it is accepting solutions in a non-stochastic manner, by a fixed threshold. Furthermore it searches along multiple branches in a breadth-first manner, which additionally facilitates parallel execution of the algorithm.

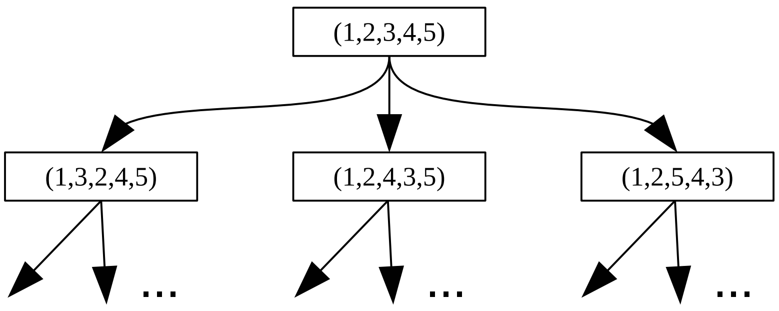


Figure 3.43: Tree created from swaps for a problem with 5 activities. Each node contains a vector of priorities. One pair of priorities is swapped along each arc.

In summary Dori (2016) has shown that swap-based local search algorithms are an effective heuristic search procedure for RCPSPs. A downside is the humongous size of the resulting search trees, making the search within the tree very tedious. This issue was addressed by the author of this thesis when developing the iterative swap-based limited depth search algorithm, that is presented in Section 4.3.

3.4.8.9 Overview of Heuristic Procedures for RCPSPs

This Section provides a tabular overview of the different published heuristic algorithms. The algorithms are classified into the previously described categories and sorted by publication date. Additionally the table includes many review papers, that provide comparisons between the different published algorithms.

Algorithm class	References
Genetic Algorithms	Chan et al. (1996), Hartmann (1998), Alcaraz & Maroto (2001), Toklu (2002), Valls & Ballestín (2002), Kochetov & Stolyar (2003), Senouci & Eldin (2004), Valls et al. (2004), Kim et al. (2005),Debels & Vanhoucke (2005), Debels et al. (2006), Jaśkowski & Sobotka (2006), Gonçalves et al. (2008), Valls et al. (2008), Alcaraz & Maroto (2009), Long & Ohsato (2009), Agarwal et al. (2011), Ballestín et al. (2011), König (2011a), Ghoddousi et al. (2013)
Hybrid Algorithms	Drexl (1991), Demeulemeester & Herroelen (1997), Mika et al. (2005), Rivera (2013)
Particle Swarm Optimization	Zhang et al. (2005a), Zhang et al. (2006), Lu et al. (2008), Chen et al. (2010), Koulinas et al. (2014)
Priority Rules	Thesen (1976), Kurtulus & Davis (1982), Kolisch (1996a), Klein (2000), Browning & Yassine (2010)
Simulated Annealing	Boctor (1996), Józefowska et al. (2001), Bouleimen & Lecocq (2003), König & Beißert (2009), Bettemir & Sonmez (2012)
Tabu Search	Thomas & Salhi (1998), Baar et al. (1999), Waligóra (2009)
Tree Search	Simpson & Patterson (1996), Bügler et al. (2013), Bügler & Borrmann (2014), Dori (2016)
X-Pass Methods	Bell & Han (1991), Kolisch & Drexl (1996), Schirmer (2000), Chtourou & Haouari (2008)
Others	Bell & Park (1990), Li & Willis (1992), Leon & Balakrishnan (1995), Salewski et al. (1997), Schirmer (2001), Tormos & Lova (2001), Calhoun et al. (2002), Merkle et al. (2002), Valls et al. (2003), Fleszar & Hindi (2004), Debels et al. (2006), Alvarez-Valdes et al. (2008), Goncharov (2011), Horenburg et al. (2012), Aziz et al. (2014), Jedrzejowicz & Ratajczak-Ropel (2014), Mejía et al. (2016)
Reviews	Wiest (1967), Davis & Patterson (1975), Cooper (1976), Elsayed (1982), Boctor (1990), Boctor (1993), Kolisch (1996b), Kolisch & Hartmann (1999), Böttcher & Drexl (1999), Hartmann & Kolisch (2000), Kolisch & Hartmann (2006), Bauer (2009), Van Peteghem & Vanhoucke (2014)

Table 3.5: List of heuristic optimization procedures for the resource constrained project scheduling problem.

3.4.9 Benchmarks

In order to evaluate the efficiency of an approach to solve resource-constrained project scheduling problems, different problem sets have been published. Davis & Patterson (1975) and Talbot & Patterson (1978) published the first such sets, consisting of 83 and 50 test problems. Later Patterson (1984) created a very popular set consisting of 110 problems of sizes varying between 7 and 50 activities, that was used throughout many publications. Kolisch & Sprecher (1997) later published the most commonly used problem sets, the Project Scheduling Problem LIBrary (PSPLIB) (Kolisch & Sprecher, 1996). It is structured by activity count into four sets with $30, 60, 90$ and 120 problems respectively. The $30, 60$ and 90 activity sets each contain 480 problem instances and the 120 activity set contains 600 problems. A majority of papers published in the area of RCPSPs since then used the PSPLIB in order to compare the performance of algorithms. For the 30 activity problems all optimal solutions are known. For all other sets the best heuristic solutions that have yet been found are stored in the library, as well as lower and upper bounds for the optimal makespan. For the 60 activity problems, most of the bounds are closed, such that the optimal makespan is known. The PSPLIB problem sets have been used in nearly all recent publications on algorithms for the RCPSP, such that it forms a solid basis for a comparison of new developments with the state-of-the-art. The author of this thesis has developed an extension of the PSPLIB which is presented in detail in Section 4.5.

3.5 Summary

Construction schedule optimization forms a vital part of the work presented in this thesis because proper reaction to changing circumstances requires optimization procedures to analyze the space of feasible schedules for good solutions. This Chapter described in detail, the state-of-the-art in construction schedule optimization. First the possible objectives and schedule metrics were introduced. Additionally the introduction of uncertainty and robustness into the problems was discussed. As the objectives, including robustness, are often antithetic or create trade-off problems, the concept of multi-criteria optimization was introduced. Through the project scheduling problem, which does not take resources into account, the resource constrained project scheduling problem was introduced. Based on an illustrative example, the different schedule generation schemes were discussed. Problem characteristics were compared and simulation-based approaches, such as the constraint-based discrete-event simulation method and the Dori algorithm were introduced. A detailed overview of heuristic approaches from literature and the introduction of the commonly used benchmarks were discussed. This Chapter illustrated that there is a very large basis of methods for solving RCPSPs in literature, but also pointed out some research voids that are addressed in the following Chapter of this thesis.

4 Improved Methods for Construction Scheduling

"In preparing for battle, I have always found that plans are useless but planning is indispensable." - Dwight D. Eisenhower

The main contributions presented in this Chapter consist of a new schedule generation scheme for resource constrained project scheduling problems, the adaption of the water wave optimization algorithm by Zheng (2015) to this problem, and a swap-based iterative limited depth search algorithm. The new algorithms offer new tools for planners to generate near-optimal schedules. Additionally the methods can be used to automate the process of updating schedules based on newly obtained data in a reactive manner. Furthermore the author developed a uncertainty extension of the project scheduling problem library PSPLIB. This extension allows scheduling algorithms to be compared on problems involving uncertain performance factors or activity durations. Such problems are common when modeling real-life construction sites and uncertainty is often generated as a product of data acquisition when maintaining a reactive simulation model. This Chapter discusses each of these contributions in the respective Sections.

4.1 Pseudo-Serial Schedule Generation Scheme

Section 3.4.4 introduced the two most commonly used serial and parallel schedule generation schemes. Schedule generation schemes form an essential part of many heuristic algorithms for resource constrained project scheduling problems. While the serial schedule generation scheme can generate all active schedules, as described in Section 3.4.4.1, it is much slower than the parrallel schedule generation scheme.

As Lemma 4 describes, the set of active schedules always contains an optimal schedule for a given RCPSP. However, in order to define a schedule to be generated by the serial schedule generation scheme, each activity needs to have a priority value that respects the precedence relationships, as the activities are processed in the exact order defined by the priorities. This increases the difficulty of systematically searching the space of possible solutions, as only a small subset of priority vectors can be used. Zhang et al. (2005a) uses a method to adjust the vectors to feasible ones prior to schedule generation by sorting the priority values in order to not violate the precedence constraints according to the work by Chan et al. (1996). This however populates the search space with numerous duplicate solutions for different input vectors and reduces the continuity of the search space.

The parallel schedule generation scheme does not suffer from this discrepancy, but does on the other hand not allow the generation of all active schedules, but only non-delay schedules. Those schedules are formed by time increments, selecting feasible activities by assigned priority values. Each activity is then started at the earliest possible time. Because the activities, to be executed next, are selected from only the feasible activities, as dictated by both the precedence, as well as the resource constraints, the priorities do not have to reflect the precedence relationships. It has been shown by Kolisch (1996b) that less than 60% of the problems in the PSPLIB j30 problem set have optimal solutions that are non-delay schedules, which can be generated by the parallel schedule generation scheme, as defined in Section 3.4.4.3.

When using schedule generation schemes as a basis for heuristic algorithms, it is desired to have a parameter space that is continuous and contains the optimal solution. This allows the algorithms to operate by applying arbitrary priority values to activities. Neither the serial, nor the parallel schedule generation scheme fulfill both criteria. For this purpose a new schedule generation scheme was developed. The pseudo-serial schedule generation scheme, yielding a complete and continuous representation of all active schedules.

As the name suggests, the set of non-delay schedules do not include schedules where an activity is executed later than the earliest point in time where the activity becomes resource- and precedence-feasible. Hence, no activity can be delayed beyond this point. The pseudo-serial schedule generation scheme is based on the parallel schedule generation scheme, extended by the concept of introducing delays. Delays are introduced into the space of possibilities by first scaling the search space down to a range of priorities in the interval $[0, 1]$. This does not have any effect on the resulting schedules as any given set of priority values can be scaled to fit into that interval without changing the relative order. Delays are then represented as integer values added to those priorities. When selecting the activity with the lowest priority value $\mathcal{P}_j$, its fractional part $\mathcal{R}_j$ is used instead. An activity being delayed once, implies that it is ignored when it occurs as the next activity to be scheduled for the first time. In this representation such an activity would be given an interval in the range $[1, 2]$. After it was ignored once, the priority value is reduced to the range $[0, 1]$ by subtracting 1. When the same activity is selected for scheduling the next time, it will no longer be ignored.

The entire procedure is summarized in Algorithm 3. Figure 4.1 illustrates the resulting search space for a problem with two activities. The bottom left rectangle, where both values $V(A)$ and $V(B)$ are in the $[0, 1]$ interval contains all non-delay schedules that can be generated by the parallel schedule generation scheme. The areas extending outwards in the top and left directions contain schedules where at least one activity has been delayed. The search space is generally bound to positive values.

Figure 4.2 shows an example problem that was introduced in Section 3.4.8.8 involving five activities and three resources. As discussed in that Section, the only schedule that can be generated by the parallel schedule generation scheme for this problem is the one illustrated in Figure 4.3. Activity E2 can executed as soon as activity P1 has finished. As

Algorithm 3 Pseudo-Serial Schedule Generation Scheme

```
 1: procedure PSSGS
    $\boldsymbol{J}$: set of all activities
    $\mathcal{E}_r(g)$: all activities which can be started resource- and precedence-feasible in stage $g$
    $\boldsymbol{C}$: set of already scheduled activities
    $\boldsymbol{A}_g$: set of active activities
    $\boldsymbol{D}_g$: set of delayed activities
    $\mathcal{P}_j$: priority value of activity $j$
    $\mathcal{R}_j$: Fractional part of the priority value $\mathcal{P}_j$
 2:     $g := 0, t_g := 0, \mathcal{C} := \emptyset$
 3:     while $|\boldsymbol{J} - \boldsymbol{C}| > 0$ do
 4:         Calculate the eligible set $\mathcal{E}(t_g)$
 5:         $\boldsymbol{D}_g = \emptyset$
 6:         while $\mathcal{E}(t_g) \neq \emptyset$ do
 7:             Select $j \in \mathcal{E}(t_g)$ with lowest priority value $\mathcal{R}_j$
 8:             if $\mathcal{P}_j < 1$ then
 9:                 Schedule $j$ at $t_g$
10:                 $\boldsymbol{C} := \boldsymbol{C} \cup \{j\}$
11:                 recalculate $\mathcal{E}(t_g)$
12:                 $\mathcal{E}(t_g) := \mathcal{E}(t_g) \setminus \boldsymbol{D}_g$
13:             else
14:                 $\mathcal{P}_j := \mathcal{P}_j - 1$
15:                 $\boldsymbol{D}_g := \boldsymbol{D}_g \cup \{j\}$
16:                 $\mathcal{E}(t_g) := \mathcal{E}(t_g) \setminus \{j\}$
17:         Calculate $\boldsymbol{A}_g$
18:         $g := g + 1$
19:         if $|\boldsymbol{A}_{g-1}| > 0$ then
20:             Calculate the minimal finish time $t_g$ of all activities in $\boldsymbol{A}_{g-1}$
21:         else
22:             $t_g = t_{g-1}$
```

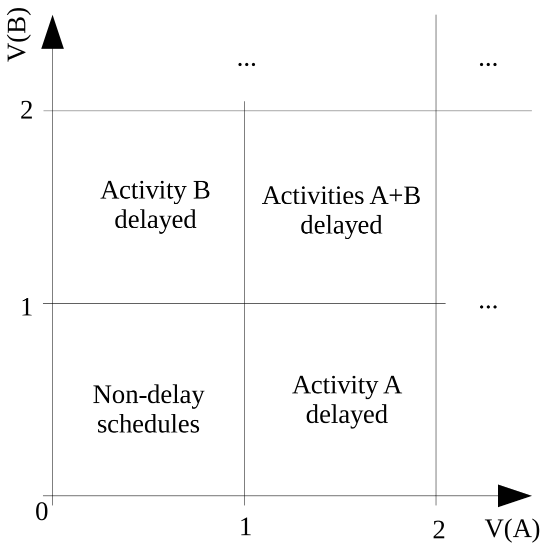


Figure 4.1: Search Space generated by the pseudo-serial schedule generation scheme for a two activity problem. Each axis corresponds the the priority of one activity. The fractional parts are priority values, the integer parts are delays.

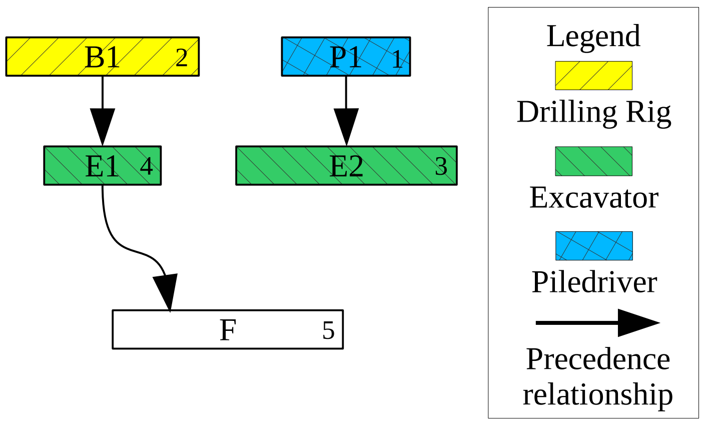


Figure 4.2: The example problem from Section 3.4.8.8. Five activities using three different resource types.

there is no other activity that can be started at that time, using the same resource, activity E2 must be scheduled at that point by the parallel schedule generation scheme. However the schedule illustrated in Figure 4.4 is the true optimal solution for the problem. In this schedule activity E2 is not scheduled at the earliest possible point t_1, but later. It is delayed beyond E2. This schedule can be generated by the serial schedule generation, giving E1 a lower priority value than E2. It can however also be generated by the pseudo-serial schedule generation scheme by delaying E2 once, given the priority value of E1 is lower. This implies that $2 > V(E2) > V(E1) + 1$.

If the priority value of E2 would be lower than the one for E1, the delay would yield

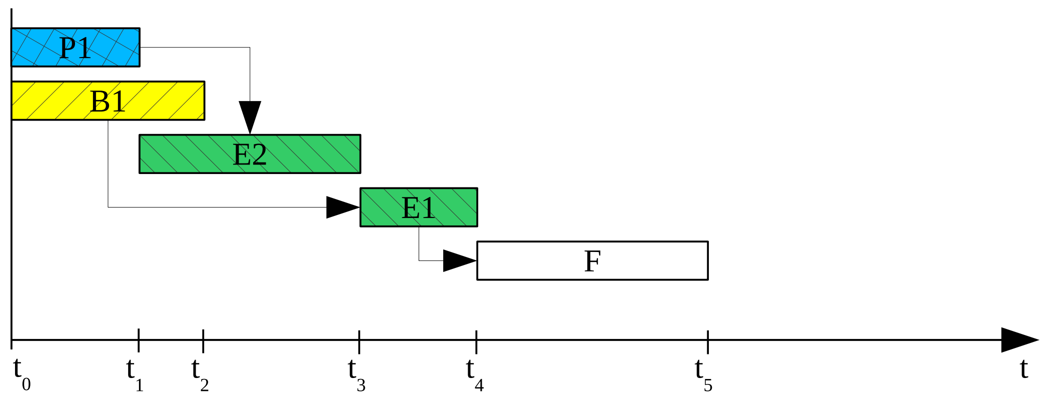


Figure 4.3: The only schedule that can be generated by the parallel schedule generation scheme for the example in Figure 4.2.

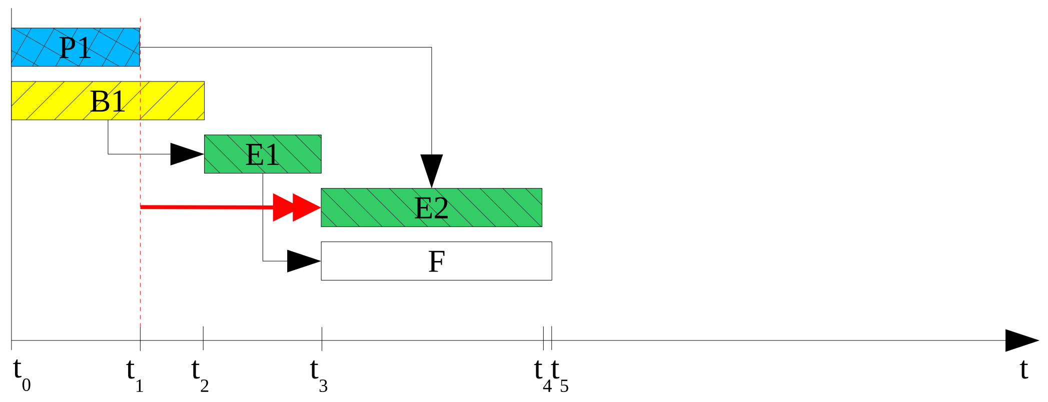


Figure 4.4: The optimal solution for the problem in Figure 4.2. Compared to the solution in Figure 4.3 this is not a non-delay schedule, but requires activity E2 to be delayed once, which is illustrated by a red double arrow.

the schedule shown in Figure 4.5. This schedule could not be generated by either the serial or the parallel schedule generation scheme as it is a delayed schedule, but E2 is not executed at it's earliest resource and precedence feasible time, given all activities with lower priority values were already scheduled. If E2 would be delayed a second time, this would again yield the schedule in Figure 4.4. If however E1 had a lower priority value than E2, but E2 would be delayed twice, this would result in the schedule illustrated in Figure 4.6, delaying activity E2 beyond activity F. Both the schedules in Figure 4.5 and 4.6 are neither active nor non-delay schedules and can under no circumstances be optimal.

Since both schedules are contained within the search space, and both schedules do allow local left shifts of activities, it can be stated, according to the theory presented in Section 3.4.4.1, that the pseudo-serial schedule generation scheme can generate a superset of the set of semi-active schedules as defined by Sprecher et al. (1995). It can however not generate the full set of feasible schedules as the time t_g can never be incremented beyond the finish time of the last activity. Therefore the algorithm can never generate a gap in

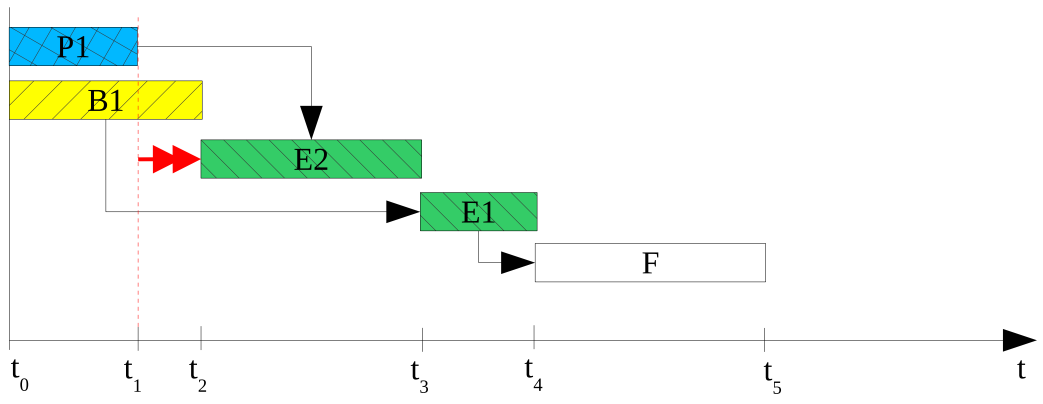


Figure 4.5: A solution for the problem in Figure 4.2 that delays activity E2 while E2 has a lower priority value than E1. This schedule cannot be generated by either of the serial and parallel schedule generation schemes. It can however be generated by the pseudo-serial schedule generation scheme.

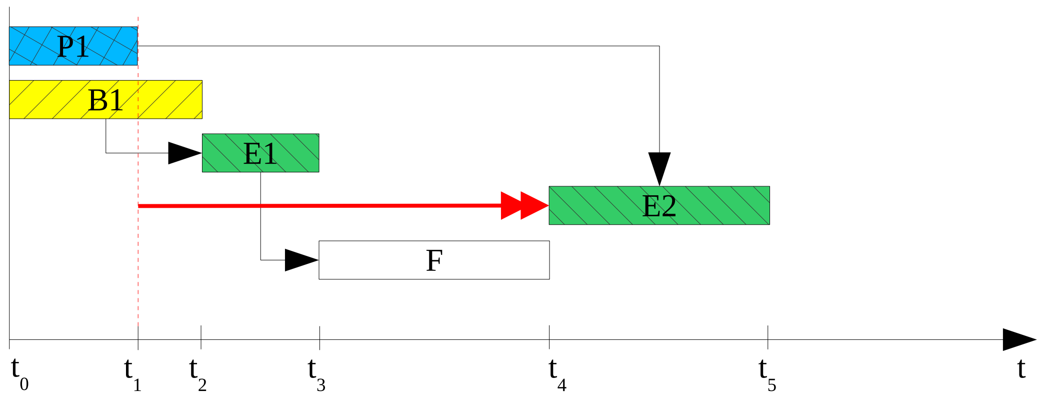


Figure 4.6: A solution for the problem in Figure 4.2 that delays activity E2 twice, while E1 has a lower priority value than E2. This schedule cannot be generated by either of the serial and parallel schedule generation schemes. It can however be generated by the pseudo-serial schedule generation scheme.

which no activity is active at all. The resulting set of schedules could accordingly be named the set of pseudo-active schedules. This extends the Lemma defined by Sprecher et al. (1995) to Lemma 9.

Lemma 9. *Let $\boldsymbol{S}$ denote the set of all schedules, $\boldsymbol{S}_F$ the set of feasible schedules, $\boldsymbol{S}_{SA}$ the set of semi-active schedules, $\boldsymbol{S}_S$ the set of active schedules, $\boldsymbol{S}_{PA}$ the set of pseudo-active schedules, and $\boldsymbol{S}_{ND}$ the set of non-delay schedules. Then following holds:*
$\boldsymbol{S}_{ND} \subseteq \boldsymbol{S}_A \subseteq \boldsymbol{S}_{SA} \subseteq \boldsymbol{S}_{PA} \subseteq \boldsymbol{S}_F \subseteq \boldsymbol{S}$.

Figure 4.7 shows the convergence of the Water Wave Optimization (WWO) algorithm discussed in Section 4.2 applied on the three discussed schedule generation schemes. The graph was created using a population of 10 over 1000 generations. As a data set the

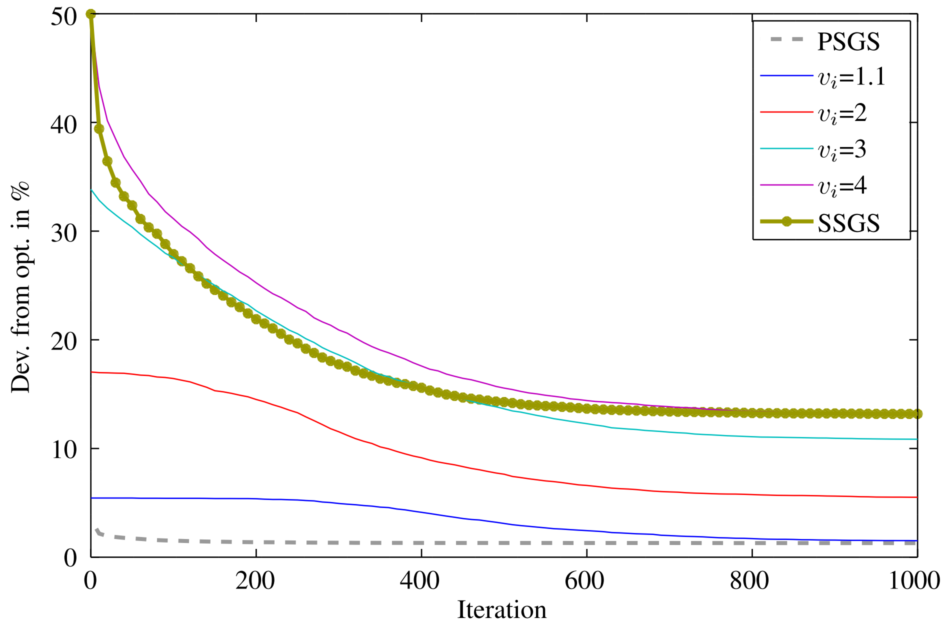


Figure 4.7: Convergence of serial (SSGS), parallel (PSGS) and pseudo-serial schedule generation schemes on the PSPLIB 30 activity benchmark set. The Water Wave Optimization (WWO) algorithm was run using all 3 scheduling schemes. In case of the pseudo-serial schedule generation scheme, the initial population v_i was varied, illustrating the transition between the serial and parallel schedule generation schemes.

PSPLIB $j30$ benchmark set was used. The lines show the average deviation in percent from the known optima of all 480 benchmark problems in the set. It is visible that the pseudo-serial schedule generation scheme can be used to model the transition between parallel and serial schedule generation scheme by adjusting the initial population. The parameter v_i defines from which area of the search space the initial population is sampled. A value of 1 implies that the population starts with only non-delay schedules. As a large part of the optimal solutions in the benchmark set are non-delay schedules, the parallel schedule generation scheme performs best on average. It is visualized by the grey dashed line. It can however be stated that for 76 of the 480 problems the solution found by the pseudo-serial schedule generation scheme was superior. The initial population was varied between $v_i = 1.1$ and $v_i = 4$. For the first case, the initial population consists of almost non-delay schedules and therefore doesn't deviate much from the parallel scheduling scheme's results. The further the parameter is increased however, the more the results approach the performance of the serial schedule generation scheme, where the initial population will always be sampled from the set of active schedules.

Schedule generation schemes generally form vital tools for heuristic algorithms solving resource constrained project scheduling problems. Compared to the commonly used serial and parallel schedule generation schemes, the pseudo-serial adaption creates the transition among both of those schemes, allowing to adjust the characteristics for the desired purpose. Restricting the search space to the $[0, 1]$ range results in the parallel schedule generation scheme, while extending it into the positive direction causes it to converge to the serial schedule generation scheme. It can however be possible to generate a problem where the number of delays for a certain activity, required to reach the optimum is so high, that the convergence of the algorithm decreases strongly, because it extends the search space to a superset of the set of active schedules. Nevertheless does the algorithm provide a valuable addition to the available schedule generation schemes and can aid improved results for many heuristic procedures. The runtime complexity of the pseudo-serial schedule generation scheme is $\mathcal{O}(N^2 \cdot |\boldsymbol{R}|)$, hence equal to that of the other schemes as presented in Section 3.4.4.

4.2 Water Wave Optimization Applied to RCPSPs

A new metaheuristic optimization algorithm named Water Wave Optimization (WWO) was introduced by Zheng (2015). The initial benchmark of the algorithm involved a train scheduling problem. The algorithm was then adapted by the author of this thesis to solve resource constrained project scheduling problems using the pseudo-serial schedule generation scheme introduced in Section 4.1 since it was already applied to a similar scheduling problem and delivered promising results.

The algorithm consists of three components, which were inspired by water wave theory, propagation, breaking, and refraction. These components model a number p of *punctual* waves within the search space, consisting of n dimensions, one for each activity. Each wave is initialized with a random position within the search space, a height h_{max} and a wavelength of $\lambda = 0.5$. In each generation of the algorithm, each wave x is propagated, creating a new wave x' based on equation 4.1, where rand$(-1, 1)$ samples a number from the uniform distribution in the range $(-1, 1)$ and λ is the wavelength. $L(d)$ is the length of the dimension d of the search space.

$$x'(d) = x(d) + \text{rand}(-1, 1) \cdot \lambda L(d) \tag{4.1}$$

After each generation the wavelength of each wave x is updated as defined by Equation 4.2. f_{min} and f_{max} are the extreme fitness values of the current population of waves and $f(x)$ is the fitness value of the current wave x. α is the wavelength reduction coefficient and ϵ is a very small number to avoid division-by-zero. This procedure simulates the movement of waves throughout the search space.

$$\lambda' = \lambda \cdot \alpha^{-(f(x)-f_{min}+\epsilon)/(f_{max}-f_{min}+\epsilon)} \tag{4.2}$$

If the height of a wave decreases to zero it is refracted based on Equation 4.3. $N(\mu, \sigma)$ samples a value from the normal distribution with mean μ and standard deviation σ. x^* is the best solution found so far. This component avoids the search to propagate towards areas of low fitness.

$$x'(d) = N\left(\frac{x^*(d) + x(d)}{2}, \frac{|x^*(d) - x(d)|}{2}\right) \tag{4.3}$$

After refraction the height of the wave is set back to h_{max} and the wavelength is updated as defined in Equation 4.4.

$$\lambda' = \lambda \frac{f(x)}{f(x')} \tag{4.4}$$

Whenever a wave x reaches a new optimal solution x^* the wave breaks up into multiple waves moving in multiple directions. A value k is randomly selected from the range $(k; k_{max})$, where k_{max} is a predefined maximum number of breaking waves. At each randomly selected dimension d it generates solitary waves x' based on Equation 4.5. If none of the solitary waves is fitter than x^*, x^* remains as the new x', otherwise the fittest of the solitary waves becomes the new x^* and x'. This component causes the vicinity of new extreme points to be evaluated for fitter values.

$$x'(d) = x(d) + N(0,1) \cdot \beta L(d) \tag{4.5}$$

The algorithm is listed as pseudo code in Algorithm 4.

Algorithm 4 Water Wave Optimization

```
 1: procedure WWO
 2:     Randomly initialize a population $P$ of $n_p$ waves
 3:     while Stop criterion is not satisfied do
 4:         for each $x \in P$ do
 5:             Propagate $x$ to a new $x'$ based on Eq. 4.1
 6:             if $f(x') > f(x)$ then
 7:                 if $f(x') > f(x^*)$ then
 8:                     Break $x'$ based on Eq. 4.5
 9:                     Update $x^*$ with $x'$
10:                 Replace $x$ with $x'$
11:             else
12:                 $x.h := x.h - 1$
13:                 if x.h=0 then
14:                     Refract $x$ to $x'$ based on Eq. 4.3 and 4.4
15:             Update wavelengths based on Eq. 4.2
16:     return $x^*$
```

(Pseudo code from Zheng (2015))

For a more detailed description of the algorithm and its implementation, the reader is referred to the original paper by Zheng (2015).

For the application of water wave optimization to resource constrained project scheduling problems, the pseudo-serial scheduling scheme, defined in Section 4.1 was used. Experiments showed that the algorithm performs best when the initial population is sampled from the vicinity of the non-delay region, as shown in Figure 4.1, only. A comparison of different initial populations is illustrated in Figure 4.7. A parameter surface for the population size and the number of generations is illustrated in Figure 4.8. The algorithm was thoroughly evaluated on the problems of the project scheduling problem library PSPLIB by Kolisch & Sprecher (1997) and outperforms many other published algorithms.

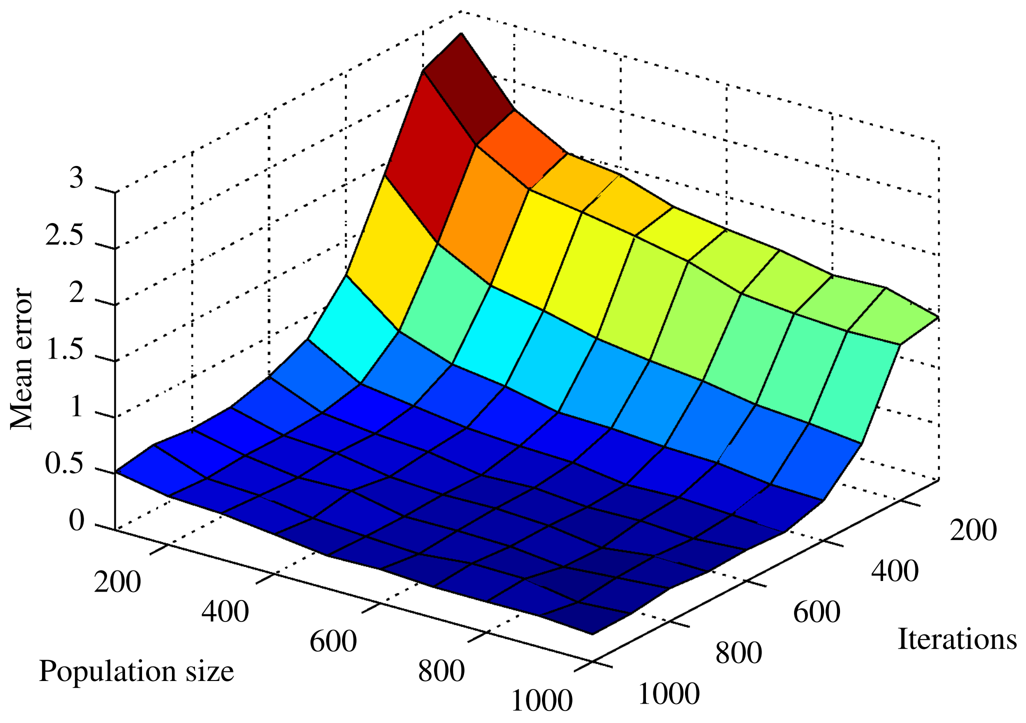


Figure 4.8: Surface of parameter values. Horizontal axes show population size n_p and number of iterations i. Vertical axis shows the mean error on the PSPLIB $j30$ problem set.

4.3 Iterative Swap-based Limited Depth Search

As described in Section 3.4.8.8 activity swaps can be used to construct search trees based on an initial schedule. Figure 4.9 shows such an initial schedule consisting of five activities using three different resource types. Figure 4.10 and 4.11 show that swapping activities B and D, or activities A and C, both result in a longer schedule, whereas applying both activity swaps results in a short schedule, as shown in Figure 4.12. This illustrates that non-beneficial swaps can be required to arrive at an optimal solution.

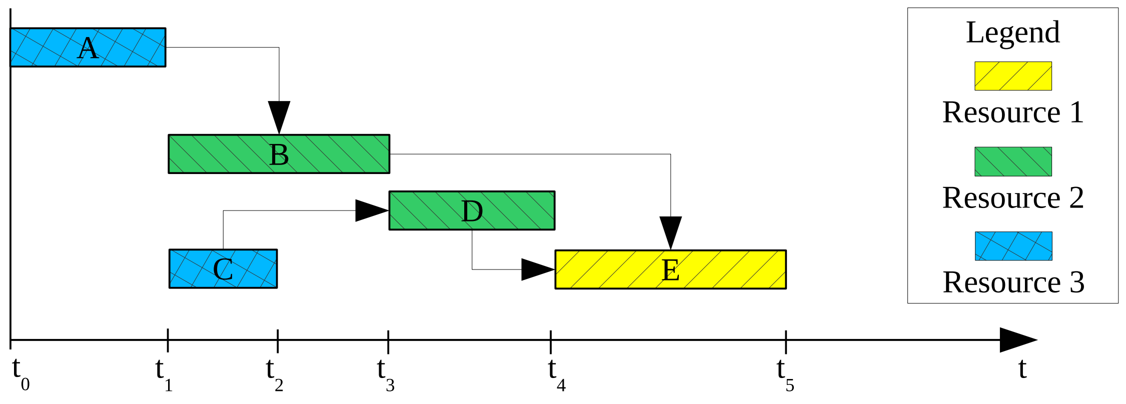


Figure 4.9: Example Schedule to illustrate local optimality. Rectangles represent activities, fill colors represent resources, and arrows represent precedence constraints.

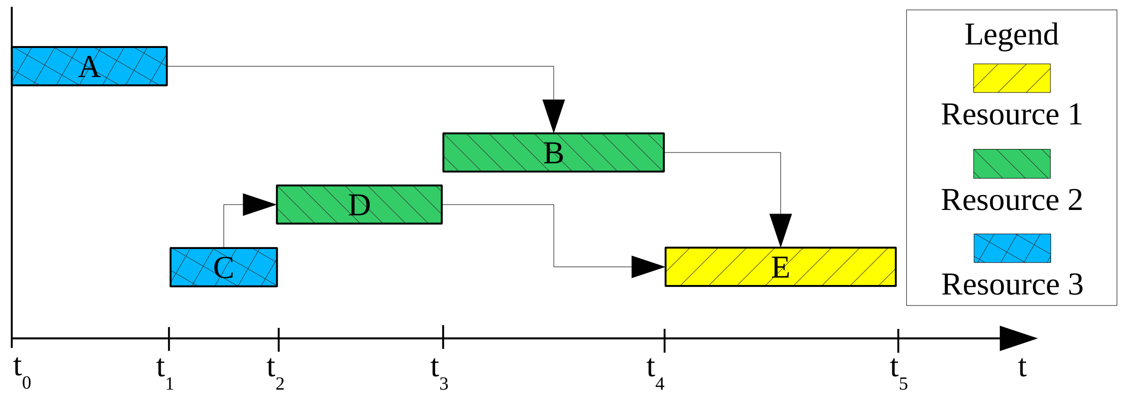


Figure 4.10: The schedule from Figure 4.9 after swapping activities B and D. The swap increases the makespan of the schedule.

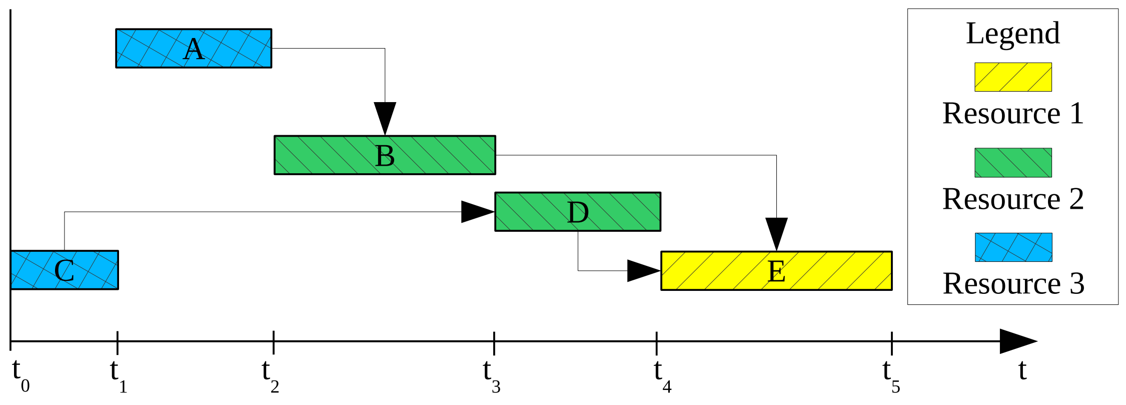


Figure 4.11: The schedule from Figure 4.9 after swapping activities A and C. The swap increases the makespan of the schedule.

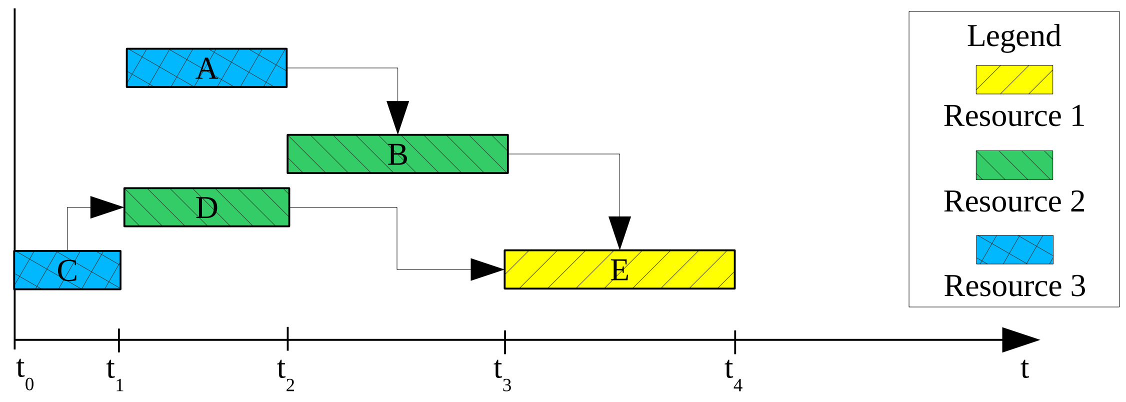


Figure 4.12: The schedule from Figure 4.9 after swapping activities A and C, as well as B and D. Applying both swaps decreases the makespan of the schedule.

The swap based local search algorithm (Dori, 2016) introduced in Section 3.4.8.8 uses the concept of reasonable swaps and the constraint-based simulation, which is similar to the parallel schedule generation scheme. For the iterative swap-based limited depth search algorithm however, the pseudo-serial schedule generation scheme, which was introduced in Section 4.1 is used implicitly. As described in Section 3.4.8.8 the parallel schedule generation scheme can be modified to collect possible swaps during the schedule generation process. The pseudo-serial schedule generation scheme can additionally collect possible delays in its process. Algorithm 5 shows the pseudo code for this process. In contrast to Algorithm 3 in Section 4.1 lines 13-15 and lines 22-24 have been added. Lines 13-15 check whether, by scheduling one activity, other activities became unfeasible, hence, whether the set of eligible activities $\mathcal{E}_r(g)$ reduced in size by more than 1. If that is the case, each of the newly unfeasible activities can be executed instead of the activity j, which was just scheduled. Hence all combinations with each of the newly unfeasible activities x create a possible swap with the scheduled activity (j, x). Additionally lines 22-24 check whether there is more than one active activity in the set of active activities for the next stage. If that is the case, all activities that have been executed in the current stage can be delayed. It can however be possible that not all of them can be delayed simultaneously, as there has to remain at least one active activity at each stage.

Figure 4.13 shows a tree created from the swaps and delays generated by Algorithm 5. The root node on the top, with index 1, shows the initial schedule from Figure 4.9. Node 2 is generated by swapping activities A and C, while node 4 is generated by swapping activities B and D. Nodes 3 and 5 are generated by delaying either activity B or C. Node 6 can either be reached by swapping B and D after node 2, or by swapping A and C after node 4. Hence it would create a duplicate node in the search tree.

In order to be greedy with computational effort it is desired to avoid duplicate nodes as described by Glover (1990b) in his work on tabu search. It does however require vast amounts of memory to keep a history of all encountered schedules during the search process. The C-sequence method and the reverse-elimination method introduced for this kind of problem in Glover (1990a) however cannot be applied to reliably detect duplicate schedules, although swaps and delays can be viewed as paired attribute moves. This is

Algorithm 5 Pseudo-Serial Schedule Generation Scheme with Swap Collection

```
 1: procedure PSSGSSC
    $\boldsymbol{J}$: set of all activities
    $\mathcal{E}_r(g)$: all activities which can be started resource- and precedence-feasible in stage $g$
    $\boldsymbol{C}$: set of already scheduled activities
    $\boldsymbol{A}_g$: set of active activities
    $\boldsymbol{D}_g$: set of delayed activities
    $\boldsymbol{U}_s$: set of possible swaps
    $\boldsymbol{U}_d$: set of possible delays
    $\mathcal{P}_j$: priority value of activity $j$
    $\mathcal{R}_j$: Fractional part of the priority value $\mathcal{P}_j$
 2:     $g := 0, t_g := 0, \mathcal{C} := \emptyset$
 3:     while $|\boldsymbol{J} - \boldsymbol{C}| > 0$ do
 4:         Calculate the eligible set $\mathcal{E}(t_g)$
 5:         $\boldsymbol{D}_g = \emptyset$
 6:         while $\mathcal{E}(t_g) \neq \emptyset$ do
 7:             Select $j \in \mathcal{E}(t_g)$ with lowest priority value $\mathcal{R}_j$
 8:             if $\mathcal{P}_j < 1$ then
 9:                 Schedule $j$ at $t_g$
10:                 $\boldsymbol{C} := \boldsymbol{C} \cup \{j\}$
11:                 calculate $\mathcal{E}'(t_g)$
12:                 $\mathcal{E}'(t_g) := \mathcal{E}'(t_g) \setminus \boldsymbol{D}_g$
13:                 if $|\mathcal{E}'(t_g)| < |\mathcal{E}(t_g)| - 1$ then
14:                     for each $x \in (\mathcal{E}(t_g) \setminus \mathcal{E}'(t_g) \setminus \{j\})$ do
15:                         $\boldsymbol{U}_s = \boldsymbol{U}_s \cup \{(j, x)\}$
16:                 $\mathcal{E}(t_g) = \mathcal{E}'(t_g)$
17:             else
18:                 $\mathcal{P}_j := \mathcal{P}_j - 1$
19:                 $\boldsymbol{D}_g := \boldsymbol{D}_g \cup \{j\}$
20:                 $\mathcal{E}(t_g) := \mathcal{E}(t_g) \setminus \{j\}$
21:         Calculate $\boldsymbol{A}_g$
22:         if $|\boldsymbol{A}_g| > 1$ then
23:             for each $x \in (\boldsymbol{A}_g \setminus \boldsymbol{A}_{g-1})$ do
24:                 $\boldsymbol{U}_d = \boldsymbol{U}_d \cup \{x\}$
25:         $g := g + 1$
26:         if $|\boldsymbol{A}_{g-1}| > 0$ then
27:             Calculate the minimal finish time $t_g$ of all activities in $\boldsymbol{A}_{g-1}$
28:         else
29:             $t_g = t_{g-1}$
```

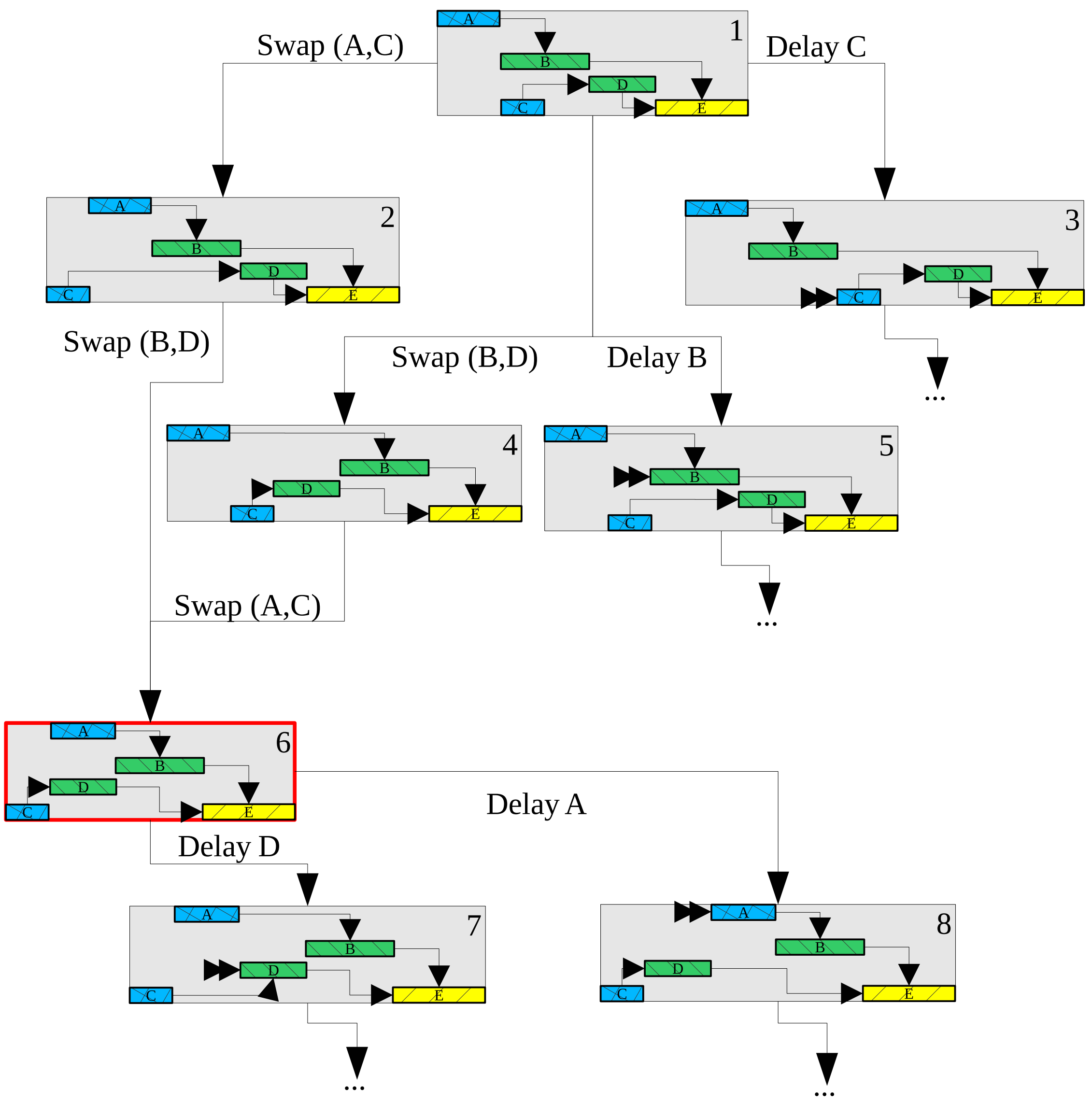


Figure 4.13: Search tree of schedules based on swaps and delays generated using Algorithm 5. The numbers in the upper right corner of each rectangle indicates the node index. The top schedule is the initial schedule from Figure 4.9. The schedule is changed by either a swap or a delay at each edge. Double arrowheads indicate delays. The red rectangle indicates the optimal solution for the problem. As the depiction contains one undirected cycle along nodes 1,2,4, and 6, it is not strictly a tree. However, node 6 would practically only be reached through either node 2 or 4.

caused by the fact that the same schedule can be reached along different paths. It is possible to reach the same schedule along different paths consisting of entirely different operations. The schedule in Figure 4.14 can for instance be generated from the schedule in Figure 4.9 by either delaying activity B, assuming the priority value for D is higher than that for B, or by swapping activities B and D.

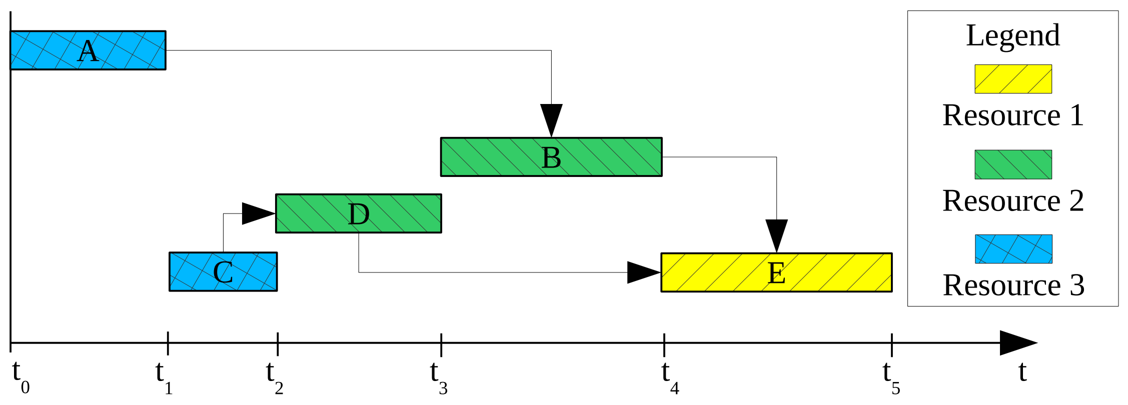


Figure 4.14: The schedule from Figure 4.9 after either delaying activity B, assuming the priority value for D is higher than that for B, or after swapping activities B and D.

The idea of the iterative swap-based limited depth tree search algorithm is based on the ascertainment that the number of swap or delay operations required to escape a local minimum can be assumed to be low. The number of steps to be taken trying to find a new optimum is the main parameter of the algorithm. The depth of search per iteration defines to how many levels the search tree is built up. The best solution found within this limited depth search tree forms a root node for the next iteration. The algorithm repeats this process until the root node of the current iteration remains the best known solution. Compared to search procedures with a fixed tolerance for accepting worse solutions, such as the tolerance search by Dori (2016), this algorithm can escape even deep local optima, given it can be escaped with a small enough number of steps. This makes sense because applying a swap early within a schedule can create an entirely different schedule. Experiments showed that the algorithm compares well to different published algorithms on the problems of the project scheduling problem library PSPLIB by Kolisch & Sprecher (1997).

4.4 Evaluation on the Project Scheduling Problem Library (PSPLIB)

This thesis introduced two new algorithms for the resource constrained project scheduling problem (RCPSP), which were presented in Sections 4.2 and 4.3. For comparing the efficiency of proposed optimization approaches, the project scheduling problem library PSPLIB by Kolisch & Sprecher (1996) has become a de facto standard. The PSPLIB consists of 4 problem sets, $j30$, $j60$, $j90$, and $j120$ with 30, 60, 90, and 120 activities per problem respectively. Each of the sets contains 480 problems, except the $j120$ set, which contains 600 problems. For each of the $j30$ problems, the optimal solutions are known, while for the others, the best known solutions are published. Almost every published algorithm for the RCPSP uses the PSPLIB to benchmark the algorithm. Therefore experiments have been performed on each of the problem sets, ranking the presented algorithms.

The detailed benchmark results are presented in 4 tables, one for each of the PSPLIB problem sets. The rows labeled *RL* shows the results of the agent-based reinforcement learning algorithm by Jedrzejowicz & Ratajczak-Ropel (2014), while the rows labeled *PSO* show the particle swarm optimization results published by Zhang et al. (2005a), and *PSO2* refers to the particle swarm optimization algorithm by Chen et al. (2010). The rows labeled *GA* list the results of the genetic algorithm presented by Alcaraz & Maroto (2001). *EA* is the evolutionary algorithm by Valls et al. (2001) and *GH* is the greedy heuristic by Goncharov (2011). *HSS* is the hybrid scatter search algorithm by Debels et al. (2006). Rows labeled **WWO PS** were obtained by the water wave optimization algorithm, which was presented in Section 4.2, operating on the pseudo-serial scheme, which was presented in Section 4.1. The rows labeled **ILDTS** were obtained by using the iterative limited depth tree search algorithm presented in Section 4.3. For each of the competing algorithms the best performing parameter set has been selected out of the published results. For details on the listed parameters, the reader is referred to the original papers. The columns labeled r_{opt} give the ratio of optimal solutions found for the $j30$ problem set, while the columns labeled r_{best} give the ratio of best known solutions found for the other sets. The $\bar{x}$ columns give the mean relative error from the optimal or best known solutions. The columns t give the mean processing time per solution and the n_f columns give the number of fitness function evaluations or schedule generations performed. Since the processing times were obtained by the individual authors on different machines, they only yield a rough comparison of the run time efficiency, but due to missing n_f values for some experiments, the values were still included in the tables. The t values for the *WWO* results were measured on an *"Intel(R) Xeon(R) CPU E5-2630 @ 2.30GHz"* machine running on a single thread per experiment. All tables are sorted in increasing order of the mean relative error $\bar{x}$. For the *WWO* rows, n_p is the population size used and i is the number of iterations of the algorithm that were performed. All other parameter values were set as recommended in the paper by Zheng (2015). $\alpha = 1.0026$, β is linearly decreasing from 0.25 to 0.001, the maximum number of breaking waves $k_{max} = \min(12, n_a/2)$ where n_a is the number of activities. The maximum wave height $h_{max} = 6$. Each table only lists the best results for each algorithms, an overview of different parameter settings can be found in Appendix B.1.

Table 4.1 shows the benchmark results for the $j30$ problem set. The hybrid scatter search (HSS) algorithm by Debels et al. (2006) performed best on this problem set, but also the two algorithms developed by the author compare well to other algorithms published in literature. While the other algorithms perform better on the data, the run times t of the agent-based reinforcement learning algorithm by Jedrzejowicz & Ratajczak-Ropel (2014) are much higher. Same holds for the evolutionary algorithm by Valls et al. (2001). The genetic algorithm by Alcaraz & Maroto (2001) and the particle swarm optimization algorithm by Zhang et al. (2005a) converged faster, as they only needed half the number of fitness function evaluations n_f in order to achieve lower error rates. The particle swarm optimization algorithm by Chen et al. (2010) did not publish the number of fitness function evaluations n_f used, but the mean deviation was higher than most tested configurations of the presented algorithm. When performing $i = 1000$ iterations on the

Algorithm	Parameters	r_{opt}	$\bar{x}$	t	n_f
HSS	$n = 500000$	99.38%	0.01%	$7,160ms$	$500,000$
RL	$s = RL123$	n/a	0.02%	$34,960ms$	n/a
WWO PS	$n_p = 10000, i = 1000$	97.71%	0.04%	$156,677ms$	$10,061,675$
EA		93.54%	0.10%	$590ms$	n/a
ILDTS	$d = 3$	93.33%	0.14%	n/a	n/a
GA	$o_c = TP\_FBC$	n/a	0.24%	n/a	$5,000$
PSO	$n_i = 125$	96.40%	0.42%	n/a	$5,000$
PSO2		n/a	0.54%	n/a	n/a
Random	$n = 500000$	65.42%	1.16%	$49,967ms$	$500,000$

Table 4.1: Benchmark results for $j30$ problem set ordered by the mean deviation $\bar{x}$ from the known optimal solutions. The column r_{opt} lists the ratio of optimal solutions found. n_f is the number of fitness function evaluations required per problem. t is the mean processing time.

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	93.13%	0.07%	$19,610ms$	$500,000$
EA		86.46%	0.17%	$1,870ms$	n/a
RL	$s = RL123$	n/a	0.51%	$62,300ms$	n/a
GA	$o_c = TP\_FBC$	n/a	0.59%	n/a	$5,000$
WWO PS	$n_p = 1000, i = 1000$	73.75%	0.87%	$69,396ms$	$1,009,426$
ILDTS	$d = 3$	64.38%	1.31%	n/a	n/a
Random	$n = 500000$	57.29%	2.14%	$218,751ms$	$500,000$
GH		68.33%	2.37%	n/a	n/a
PSO2		n/a	3.71%	n/a	n/a

Table 4.2: Benchmark results for $j60$ problem set ordered by the mean deviation $\bar{x}$ from the best known solutions. The column r_{best} lists the ratio of best known solutions found. n_f is the number of fitness function evaluations required per problem. t is the mean processing time.

a population size of $n_p = 10000$, using the *WWO PS* algorithm, the error is $\bar{x} = 0.04\%$, while in 97.71% of the problems the global optimum was found. This outperforms the results of the *PSO*, *EA*, and *GA* algorithms. For comparison with the *PSO* and *GA* algorithms two experiments with each $n_f \sim 5000$ schedules generated have been conducted yielding a minimum mean error of $\bar{x} = 1.23\%$ which is slightly worse than the compared algorithms. Generally it was observed that more than $i = 1000$ iterations only have a small effect on the outcome, while the population size plays a more important role. This was illustrated by a performance surface in Figure 4.8 in Section 4.2, which shows the mean error over the space of the parameters population size and iterations on the $j30$ problem set. It shows that a certain number of iterations is required, after which a larger population yields better improvements. The **ILDTS** algorithm performs slightly worse than the **WWO PS** algorithm, finding 93.33% of the global optima with a mean error of

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	91.04%	0.07%	$38,870ms$	$500,000$
EA		88.75%	0.16%	$4,900ms$	n/a
RL	$s = RL123$	n/a	0.98%	$77,290ms$	n/a
WWO PS	$n_p = 1000, i = 1000$	71.67%	1.61%	$133,577ms$	$1,011,275$
ILDTS	$d = 3$	65.42%	1.62%	n/a	n/a
Random	$n = 500000$	57.71%	2.64%	$353,538ms$	$500,000$
PSO2		n/a	3.79%	n/a	n/a

Table 4.3: Benchmark results for $j90$ problem set ordered by the mean deviation $\bar{x}$ from the best known solutions. The column r_{best} lists the ratio of best known solutions found. n_f is the number of fitness function evaluations required per problem. t is the mean processing time.

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	72.33%	0.28%	$69,570ms$	$500,000$
EA		50.17%	0.75%	$35,000ms$	n/a
GA	$o_c = TP\_FBC$	n/a	1.36%	n/a	$5,000$
RL	$s = RL123$	n/a	2.91%	$202,410ms$	n/a
WWO PS	$n_p = 1000, i = 1000$	25.33%	5.48%	$259,471ms$	$1,020,095$
ILDTS	$d = 2$	11.83%	6.18%	n/a	n/a
Random	$n = 500000$	14.17%	7.41%	$482,920ms$	$500,000$
PSO2		n/a	9.38%	n/a	n/a

Table 4.4: Benchmark results for $j120$ problem set ordered by the mean deviation $\bar{x}$ from the best known solutions. The column r_{best} lists the ratio of best known solutions found. n_f is the number of fitness function evaluations required per problem. t is the mean processing time.

$\bar{x} = 0.14\%$ using a search depth of $d = 3$.

Table 4.2 lists the results of running the different algorithms on the $j60$ problem set of the PSPLIB. In its best configuration with a population of $n_p = 1000$ waves and $i = 1000$ generations, the **WWO PS** algorithm finds 73.75% of the best known solutions and deviates by 0.87% on average. While outperforming the published results for the greedy heuristic by Goncharov (2011) and the particle optimization algorithm by Chen et al. (2010), the other algorithms perform slightly better on the benchmark data. Due to missing data on the number of fitness function evaluations n_f a fair comparison of the evolutionary and reinforcement algorithms was not possible. The presented **WWO PS** algorithm can be configured to run much faster than the other algorithms, while the drawback on the quality results is very small. The **ILDTS** algorithm, given a depth of $d = 3$ found 64.38% of the best known solutions with an average deviation of 1.31% and outperforms the *GH* and *PSO2* algorithms.

For the $j90$ benchmark set only results for four of the other algorithms were published, which are compared in Table 4.3. With a ratio of 71.67% of the best known solutions found and a mean deviation of 1.61% the **WWO PS** algorithm delivers good results which outperform the particle swarm optimization algorithm by Chen et al. (2010), but could neither match the reinforcement learning algorithm by Jedrzejowicz & Ratajczak-Ropel (2014) nor the evolutionary algorithm by Valls et al. (2001) in quality of the results. With little drawback though, the presented algorithm can be configured to run much faster than the other algorithms. Unfortunately the number of fitness function evaluations n_f required to obtain the results was not published for either of the other algorithms. The **ILDTS** managed to find 65.42% of the best known solutions and deviated by 1.62% on average.

Table 4.4 compares the results of the presented algorithms to the other algorithms on the most complex $j120$ problem set containing 600 problems with 120 activities each. While the genetic algorithm (*GA*) and the hybrid scatter search (*HSS*) gave outstanding results, the other algorithms published very high run times for their results. In the best configuration that was tested, $n_p = 1000, i = 1000$ the **WWO PS** algorithm was only able to outperform the particle swarm optimization algorithm by Chen et al. (2010), but with a mean error of $\bar{x} = 5.48\%$ still produced very good results. The **ILDTS** algorithm falls back, with only 11.83% of the best known solutions found, and an avarage deviation of 7.41%, given a search depth of $d = 2$. Experiments with larger search depths were not feasible due to the immense number of possible swaps within the schedules.

While outperformed by some of the algorithms in literature, especially the hybrid scatter search algorithm by Debels et al. (2006), the presented algorithms provide a valuable addition to the available arsenal of RCPSP heuristics. Especially the pseudo-serial schedule generation scheme offers the possibility to be used with a wide variety of other metaheuristic algorithms.

4.5 Uncertainty Extension for the Project Scheduling Problem Library PSPLIB

Construction processes rarely have a constant duration that is known a-priori. Instead their durations are estimated based on experience and previously collected sample data. From this sample data probability distributions can be estimated. The Project Scheduling Problem Library PSPLIB by Kolisch & Sprecher (1997) only contains problems with fixed durations. In order to determine the practical capabilities of an developed algorithm however, its performance on problems involving uncertainty is of high interest. Bershtein et al. (2015) extended some of the PSPLIB problems by using a 6-point piecewise linear membership function as defined by Rommelfanger (1990). They did however not publish the used distributions, such that they can be used for comparison studies. Another approach by Fu et al. (2012) added normal distributed delays with mean 0 and standard deviation 1 to each activity of the PSPLIB problems.

However, as described in Section 2.2.1, the beta distribution is the most suitable representation of uncertain durations in construction operations. Therefore beta distribution parameters were created for each problem in the PSPLIB. For each activity of each problem in the single mode sets $j30$, $j60$, $j90$, and $j120$ a maximum delay $\hat{d}_j$, and the α_j and β_j parameters for the beta distribution have been pseudo-randomly created[5]. The extended problem sets are named $j30u$, $j60u$, $j90u$, and $j120u$ respectively. The delays are uniformly sampled integers from the interval $(1, 5)$, and the parameters α and β are uniformly sampled integers from the interval $(1, 10)$. In order to get a sample for the duration of activity j $\bar{d}_j$, Equation 4.6 can be used, where $B(\alpha_j, \beta_j)$ takes a sample from the beta distribution with parameters α_j and β_j. Since many implementations are based on integer durations, the sampled duratios are rounded to integers. This has the advantage of the delays having strict bounds which creates the advantage that the known optima for each of the PSPLIB's problems form a lower bound for each of the uncertainty adaptions, which does not hold for the normal distributed approach by Fu et al. (2012).

$$\bar{d}_j = d_j + [\hat{d}_j \cdot B(\alpha_j, \beta_j)] \tag{4.6}$$

As an example, each activity's duration was plotted for the $j30u$ problem with instance 1 and parameter 1 in Figure 4.15. The resulting possible schedules are plotted in Figure 4.16. The figure shows the Monte-Carlo results for $100,000$ runs. Each plot shows the likelihood of an activity being active at the respective time. The precedence constraints of the activities are illustrated in Figure 4.17. Figure 4.18 shows the distribution of the overall makespan over $1,000,000$ runs.

In order to compare different algorithms on this extended benchmark set, a number of eight objective functions have been designed, which are listed in Table 4.5. Each objective is based on performing a certain number of Monte Carlo runs of a resulting schedule. $\mathcal{O}_1$ defines the first objective function of the minimum makespan of all the $\mathcal{K}$ Monte Carlo runs. $\mathcal{O}_2$ defines the second objective as the mean makespan, while $\mathcal{O}_3$ defines the worst

[5]The generated values have been published in the git repository hosted at https://github.com/tumcms/upsplib to be used for comparison studies.

encountered makespan. $\mathcal{O}_4$ defines the standard deviation of the set of the Monte-Carlo makespans. $\mathcal{O}_5$ defines the objective as the sum of the mean makespan and the standard deviation. $\mathcal{O}_6$ defines the objective as the mean makespan added to twice the standard deviation. $\mathcal{O}_7$ is the worst encountered makespan added to the standard deviation, while $\mathcal{O}_8$ adds twice the standard deviation. $\mathcal{O}_5$ till $\mathcal{O}_8$ are multi-criteria objectives using a weighted sum as presented in Section 3.2.2.4.

$\mathcal{O}_1 = \min\limits_{k \in \mathcal{K}}(m_k)$	$\mathcal{O}_2 = \sum\limits_{k \in \mathcal{K}} \frac{m_k}{\lvert\mathcal{K}\rvert}$
$\mathcal{O}_3 = \max\limits_{k \in \mathcal{K}}(m_k)$	$\mathcal{O}_4 = \sqrt{\sum\limits_{k \in \mathcal{K}} \frac{m_k^2}{\lvert\mathcal{K}\rvert} - (\sum\limits_{k \in \mathcal{K}} \frac{m_k}{\lvert\mathcal{K}\rvert})^2}$
$\mathcal{O}_5 = \mathcal{O}_2 + \mathcal{O}_4$	$\mathcal{O}_6 = \mathcal{O}_2 + 2 \cdot \mathcal{O}_4$
$\mathcal{O}_7 = \mathcal{O}_3 + \mathcal{O}_4$	$\mathcal{O}_8 = \mathcal{O}_3 + 2 \cdot \mathcal{O}_4$

Table 4.5: The eight objective functions of the UPSPLIB problem library.

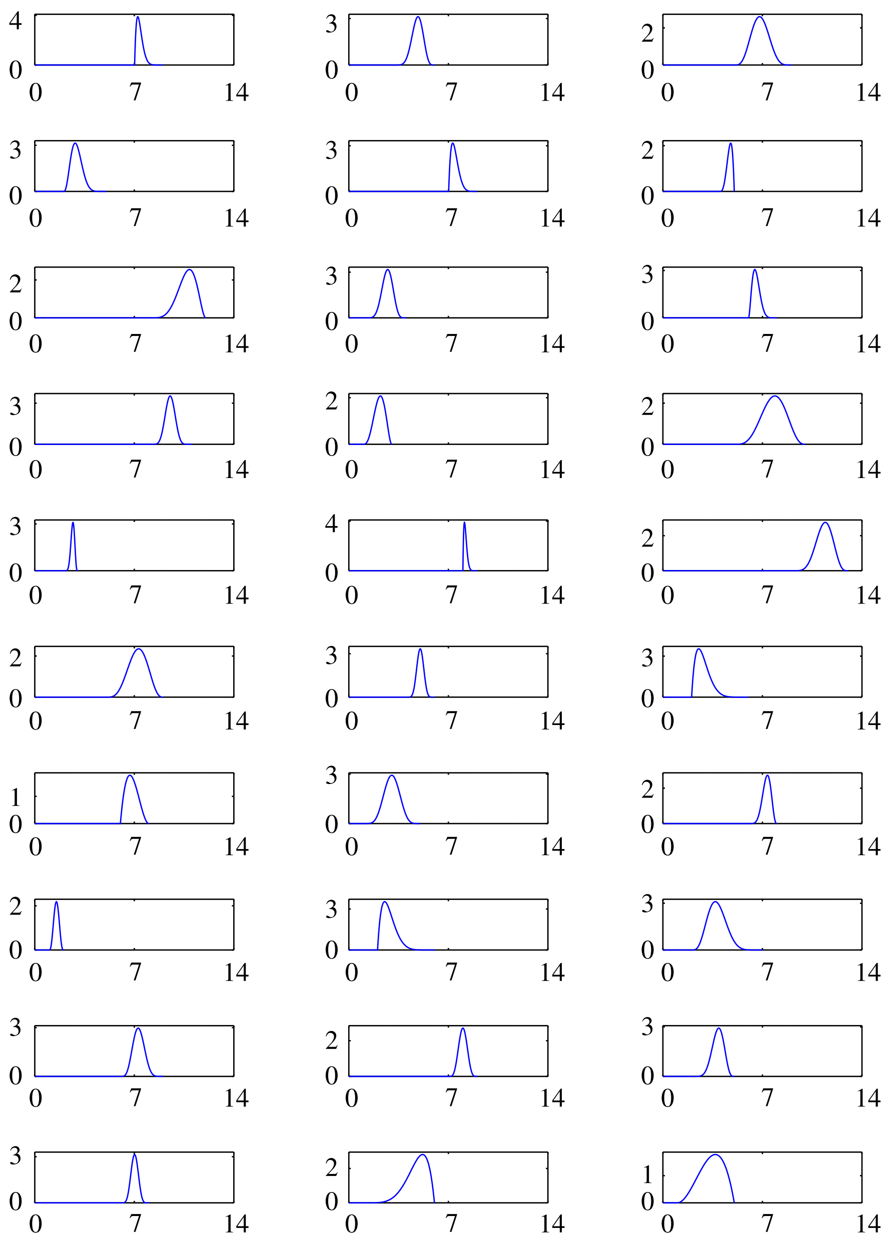


Figure 4.15: Uncertain durations of the 30 individual activities for the $j30u$ problem with instance 1 and parameter 1. Each activity is delayed using a beta distributed random variable. The horizontal axes show the duration, the vertical axes the probability. Activities are arranged from left to right, then top to bottom.

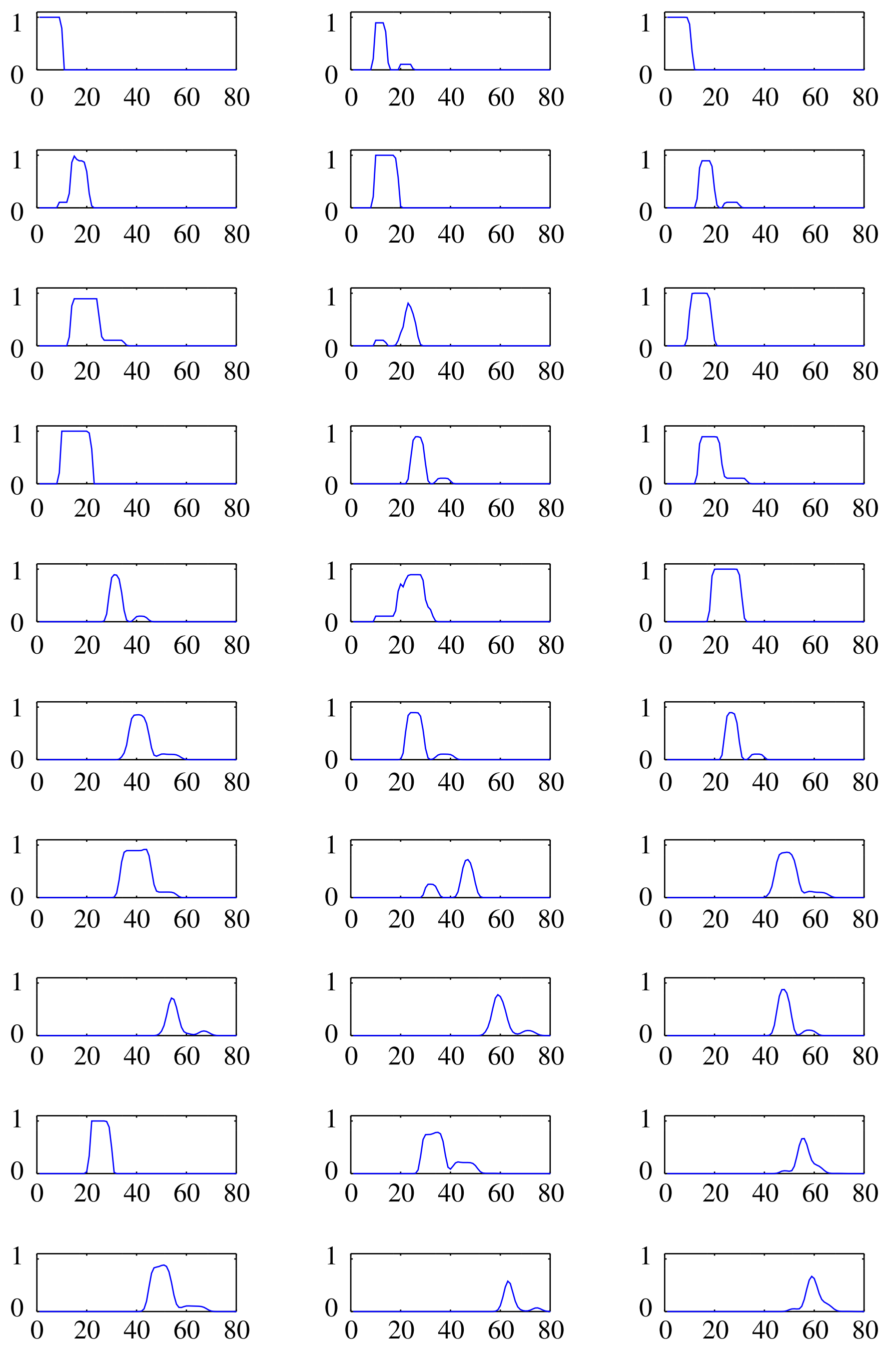


Figure 4.16: Schedule for the $j30u$ problem with instance 1 and parameter 1, as shown in Figure 4.15, using the default activity order with the parallel schedule generation scheme. Each plot corresponds to one activity and shows the probability of an activity being active at a certain time. The plots are based on $100,000$ runs. Activities are arranged from left to right, then top to bottom.

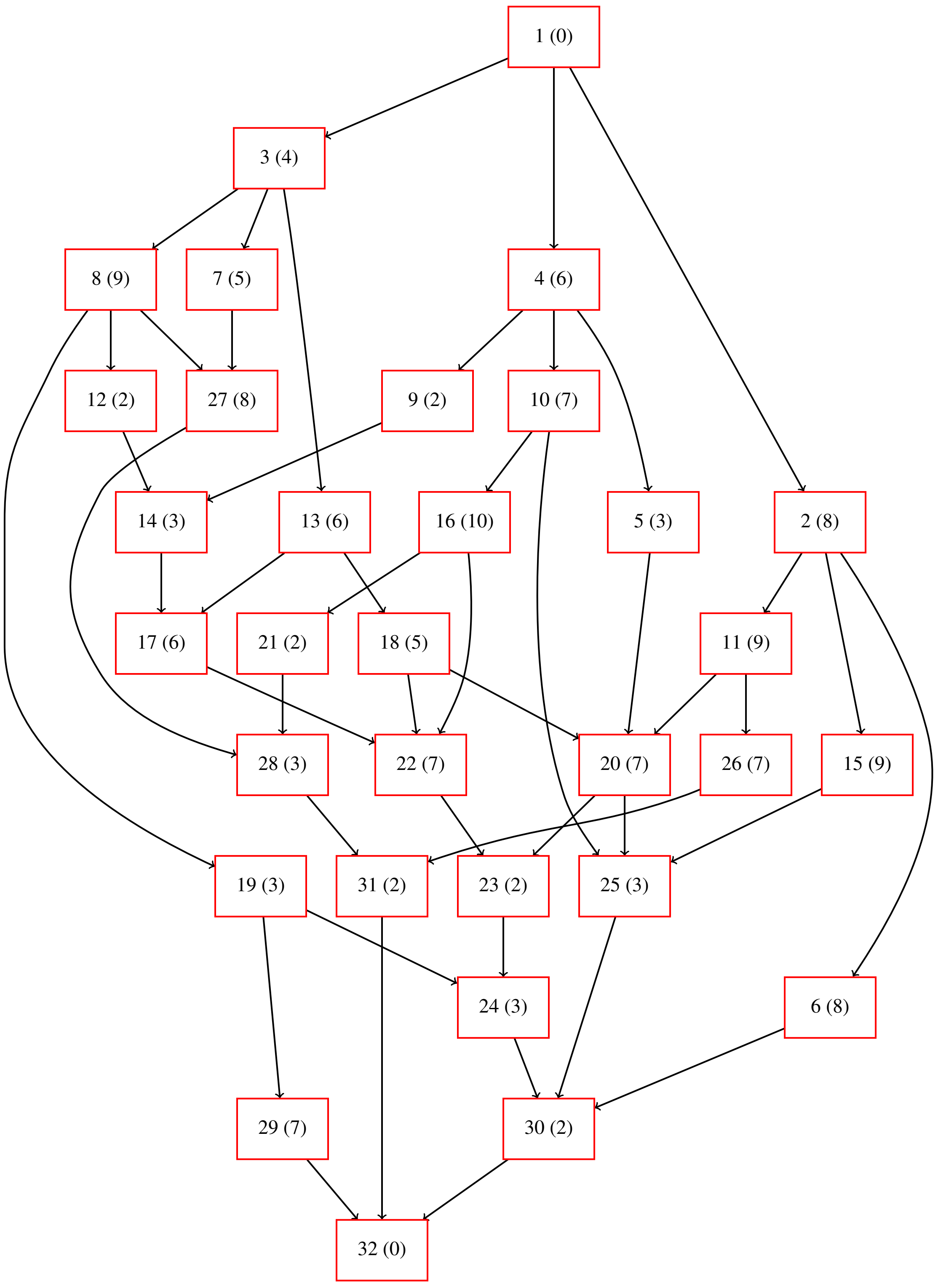


Figure 4.17: Dependency graph of the 30 activities for the $j30u$ problem with instance 1 and parameter 1. The numbers in parentheses are the minimum durations. Activities 1 and 32 are added source and sink activities with duration 0.

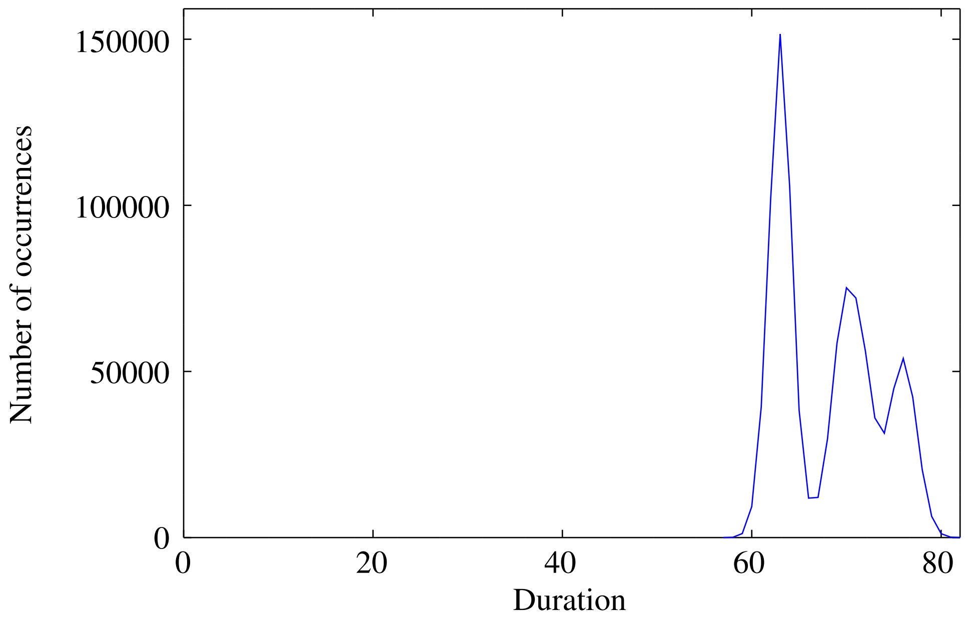


Figure 4.18: Uncertain duration of a schedule for the $j30u$ problem with instance 1 and parameter 1, as shown in Figure 4.15, using the default activity order with the parallel schedule generation scheme. The vertical axis shows the number of occurrences out of $1,000,000$ runs.

This Section contains three tables for the $j30u$ problem set. Each table has one column for each of the objective functions defined previously. Furthermore there are only four rows for the first four objective functions, as the other functions are combinations of the first four. The columns correspond to the optimization objectives that were used, and the rows correspond to the output values. Each table was populated using simulation results of the best solutions found, according to the respective objective functions, using the water wave optimization algorithm with a population of $p = 10$ waves and $n = 1000$ iterations. For each objective function evaluation, 10 Monte Carlo runs were carried out. The final solution for each of the optimization runs was evaluated $10,000$ times in a Monte Carlo way.

Table 4.6 shows the minimal deviations from the optimal solutions of the corresponding $j30$ PSPLIB problem for the first three rows, while the last row shows the standard deviation. The first row shows the minimum deviation of the best simulation runs for each solution. The second row shows the mean deviation and the third row shows the maximum deviation. Table 4.7 shows the same results, but the mean values of the $10,000$ runs, while Table 4.8 shows the worst solutions encountered during the Monte Carlo runs. Looking at Table 4.7 showing the mean results several things can be noticed. The lowest minimum makespan is achieved when using the minimum makespan $\mathcal{O}_1$ or the mean makespan $\mathcal{O}_2$ as objective function. The lowest maximum makespan is achieved when using the maximum makespan $\mathcal{O}_3$ as an objective function. But similar results are achieved when minimizing the sum of the mean makespan and the standard deviation ($\mathcal{O}_5$) or maximum makespan and the standard deviation ($\mathcal{O}_7$). Using the standard deviation $\mathcal{O}_4$ as an objective function generally returned poor results. This can be traced back to the standard deviation creating a very discontinuous search space. When looking at the worst case Table 4.8 is is visible that the objective function that minimizes both the worst case makespan, as well as the worst case standard deviation is the sum of the mean makespan and the standard deviation ($\mathcal{O}_5$). Additionally this objective function also acchieves the lowest worst case and mean best case values in Table 4.6. More detailed results are presented in Appendix B.2 and the precise solutions have been published to github[6].

[6]https://github.com/tumcms/upsplib

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.05	1.02	1.04	1.04	1.03	1.03	1.04	1.04
$\mathcal{O}_2$ (mean)	1.11	1.11	1.11	1.13	1.11	1.11	1.12	1.11
$\mathcal{O}_3$ (max)	1.16	1.16	1.16	1.21	1.16	1.18	1.20	1.18
$\mathcal{O}_4$ (stDev)	0.89	0.89	1.02	1.03	0.96	0.98	0.95	0.92

Table 4.6: Best solutions of water wave optimization on the uncertainty-extended PSPLIB $j30u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.21	1.21	1.22	1.54	1.22	1.22	1.22	1.22
$\mathcal{O}_2$ (mean)	1.33	1.32	1.32	1.68	1.32	1.32	1.32	1.32
$\mathcal{O}_3$ (max)	1.49	1.47	1.46	1.85	1.46	1.47	1.46	1.47
$\mathcal{O}_4$ (stDev)	2.82	2.41	2.09	2.47	2.11	2.17	2.09	2.09

Table 4.7: Mean results of water wave optimization on the uncertainty-extended PSPLIB $j30u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.46	1.46	1.49	2.36	1.48	1.53	1.56	1.49
$\mathcal{O}_2$ (mean)	1.68	1.59	1.60	2.62	1.62	1.67	1.66	1.62
$\mathcal{O}_3$ (max)	1.91	1.97	2.11	2.77	1.88	2.09	2.14	2.06
$\mathcal{O}_4$ (stDev)	12.79	9.44	7.04	8.30	5.72	9.87	8.01	6.24

Table 4.8: Worst solutions of water wave optimization on the uncertainty-extended PSPLIB $j30u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

This Section illustrated that the uncertainty extension of the PSPLIB problem library enriches the commonly used benchmark problems by adding uncertain delays to each problem and therefore allowing to evaluate an algorithms performance for RCPSPs with uncertain activity durations. The added objective functions allow the heuristic optimization of different statistical characteristics of the problems. The inclusion of the schedule durations' standard deviation generates a more chaotic search space, creating an added challenge for heuristic optimization procedures.

4.6 Data Fusion and Reactive Scheduling

Once changes in the circumstances of a construction project have been detected, or delays have occurred, a replanning process might be necessary in order to account for the changes. The critical path of the schedule and the float times of the activities, as described in Section 3.4.6 play an important role in that process. If a process is delayed beyond its float time, or the process is critical, a replanning process has to occur. Any activity that has already finished can be neglected in that process. Same holds for any precedence constraint linked to the removed activities. If any of the removed activities did take an unanticipated amount of time, the performance factors, and possibly the probability distributions, of the involved resources might need to be updated, given there are future processes that are affected by them. Activities that are currently in progress need to be shortened by the amount of time the process has been in execution. If the percentile progress of an activity that is currently in execution is known, a performance factor update might change the duration and probability distributions of that activity. Once all those changes have been incorporated into the project model, a new optimization run can be performed, yielding an updated schedule, taking the newly arisen circumstances into account.

4.7 Summary

This Chapter introduced a new schedule generation scheme, the pseudo-serial schedule generation scheme, that combines advantages of both the commonly used schemes, the serial and parallel schedule generation schemes. It has been shown that the new scheme can generate all active schedules, as does the serial scheme, but is computationally more efficient and can be used to collect possible swaps and delays during schedule generation, as does the parallel schedule generation scheme. Furthermore this Chapter introduced two novel scheduling algorithms. The Water Wave Optimization (WWO) algorithm by Zheng (2015) was applied to RCPSPs and the iterative swap-based limited depth search has been developed. Both methods have been shown to compete with other algorithms from literature, however the WWO algorithm proved to be much stronger and computationally more efficient in finding good solutions to the problems of the PSPLIB. Finally an extension of the PSPLIB has been developed, adding uncertainty using beta-distributed delays to each activity in each problem. All values have been published to allow comparisons to be performed by other researchers. The extension allows scheduling algorithms to be evaluated on problems with uncertain performance factors and activity durations. It has been shown that the presented algorithms can handle the extended problems and therefore can be applied to real-life problems involving data acquisition with uncertainty in the context of a reactive scheduling system.

5 State-of-the-Art in Data Acquisition on Construction Sites

"Not everything that counts can be counted and not everything that can be counted counts." - Cameron (1963)

Large construction projects are built over long periods of time, sometimes years pass before the project is finished. As such projects consist of many individual activities it can be difficult to maintain an overview over the state of all of them. Additionally construction projects are commonly subject to unforeseen events that affect the building process. Therefore copious research was dedicated to data acquisition techniques for construction activities. For the reactive scheduling approach, data feedback is a vital ingredient. The acquired progress information can be used to update the model of the construction project by updating performance factors and activity durations. This data is important to keep the model of the construction project synchronized and therefore allow for reactive planning and prognosis.

5.1 Protocol-based Tracking

Nowadays it is still common practice to manually log construction progress on paper or in spreadsheats. Kagioglou et al. (2000) formalized the generic design and construction process protocol (GDCPP) which is a framework model, describing processes on a construction site in a formal manner, such that each involved party's interests are included. Furthermore it facilitates computer-driven processing and optimization of the project structure. Such a formal process description can also be used to precisely log the progress of each involved activity. While in many cases the use of manual logging of progress information is common, Saidi et al. (2002) discusses the value of using handheld devices such as tablets in order to aid with this process. This process was later improved by special purpose android applications such as "BIM Track", which was presented by Marzouk & Zaher (2015). This application allows the end user to input status and progress information, such as dates, completion ratio, and actual cost, by using a building information model. Generally though, human-driven approaches can be error-prone and cause costs by requiring manpower on site.

5.2 Machine data

Many machines nowadays are equipped with sensors to evaluate their performance. For instance, modern drilling rigs record the drill depths, pressure levels, and torque at certain time intervals (Obergrießer et al., 2009). The data recorded by those sensors can be used to collect information about the activities that were performed using that machine (Kargul et al., 2015). A patent for such a system was filed by Likins et al. (1999). It uses the sensors installed within a drilling rig to record data on the performance of pile installation systems. Based on the retrieved values, such as drill depth, speed, pressures and velocity the individual piles, that were produced, can be distinguished and linked to respective time windows. Viljamaa & Peltomaa (2014) describes a more holistic construction progress control system also taking excavator and truck mass realization data into account. The use of machine data has a fairly limited focus as it is only useful when machines are deeply involved into the activities to be observed and the sensor data can be used to derive information on the activity performed as well as its status. For instance the rotation and load of a tower crane does not deliver sufficient information to infer what kind of activity it is lifting material for.

5.3 Laser Scanning

A commonly used method to acquire three dimensional data of construction sites is laser scanning. Cheok & Stone (1999) and Cheok et al. (2000) proposed the use of laser scanning in order obtain measurements from construction sites without interfering with the ongoing activities. They compared the measurements with a 3D model of the project to notice deviations. Kern (2002) proposed the use of laser scanner technology to calculate precise volume information from the resulting 3D point clouds of earthwork operations. Hashash et al. (2005) and Su et al. (2006) advanced the approach towards obtaining geotechnical measurements during urban excavation operations. This approach allows to obtain progress information of the excavation activities. Shih (2003) and Shih & Wang (2004) used laser scanners to compare as-built with as-planned information based on the resulting point clouds. This allows the tracking of objects on site and to determine whether a part was already placed at its destination position. Tang et al. (2010) reviews methods for detecting objects in building information models (BIM) within laser-scanned point clouds based on different geometrical models. Those methods were successfully applied for dimensional compliance control by Bosché (2010). In contrary Dore & Murphy (2014) present an approach to create building models from laser scans of façades. The wide range of possible uses of laser scanned point clouds make this technology an attractive choice. The laser scanner however is expensive, needs trained personnel, and multiple scans may be required to capture all relevant locations. Through stitching of the resulting point clouds, small errors can be created which make it troublesome to locate small objects on site (Matiukas & Miniotas, 2011).

5.4 Photogrammetry

Ullman (1979) and Webb & Aggarwal (1982) developed an approach, called structure-from-motion (SFM), that allows to create three dimensional point clouds from photographs. Later this idea was developed further into a system called VisualSFM by Wu (2011). In combination with the patch-based multiview stereo (PMVS) algorithm published by Furukawa & Ponce (2010) this method allows to generate very high quality colored point clouds from unordered photographs of an object.

In order to generate such point clouds, a method for finding common features in images is required. For this purpose the Scale Invariant Feature Transform (SIFT) (Lowe, 1999) and the Speeded Up Robust Features (SURF) (Bay et al., 2006) algorithms are most commonly used. Both algorithms create descriptions of local geometric features within images that are invariant to scale and rotation. Furthermore they are robust against changes in illumination, noise and slight deformation. Each feature can be compared to other features in order to identify similar points in different images. Figure 5.1 a) and b) shows two illustrative images of a mountain site. Figure 5.1 c) and d) shows how features in the images can be mapped to each other. The blue dashed lines show the identified features. This technique is commonly used for panorama stitching, but also finds use in the SFM approach.

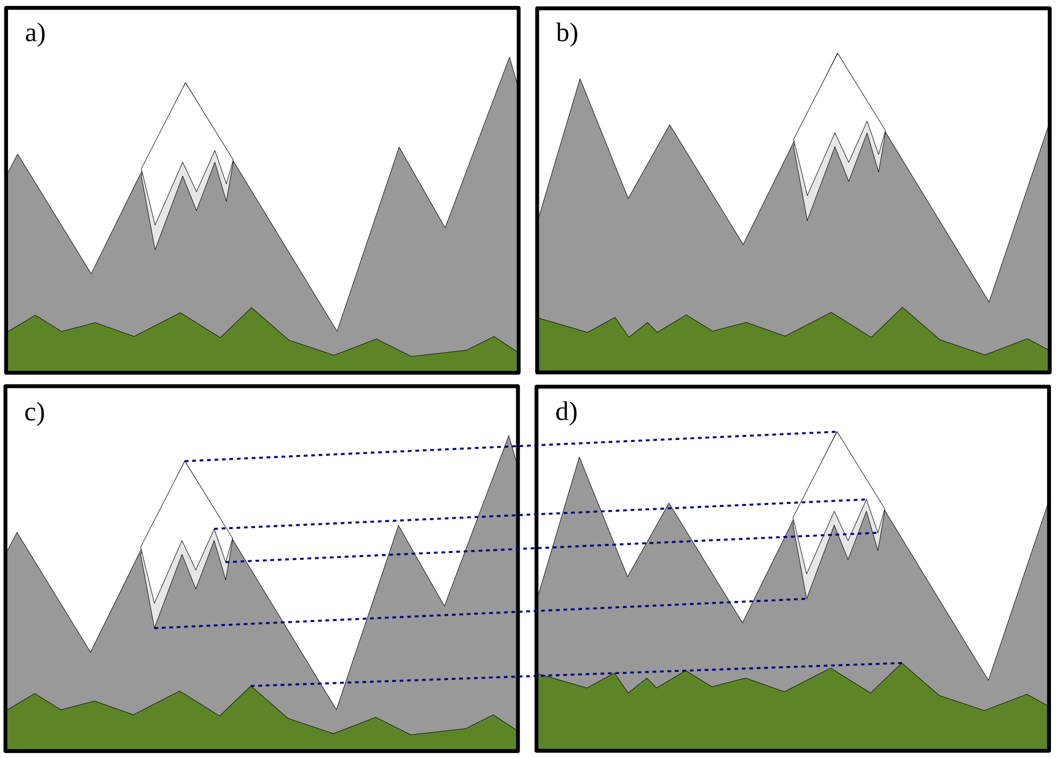


Figure 5.1: Matching of image features. a) and b) show two input images. The blue dashed lines connect matching features in c) and d).

The phrase structure-from-motion implies that 3D information can be extracted from 2D images by motion of the camera. More precisely, many such images have to be taken and their features have to be matched among them. Figure 5.2 illustrates that a feature located in a single 2D image can not generally be located in 3D space (See Figure 5.2). In order to make use of the motion of the camera in between multiple shots it is required to locate the same feature points within the images. In theory this requires the features of every image to be compared to the features of every other image, which requires $\frac{n^2}{2}$ image comparisons for n images. For large objects where many images are required, this can take substantial amounts of time. In order to reduce this effort, Silpa-Anan & Hartley (2008) have developed an optimized matching algorithm based on the k-d tree algorithm by Beis & Lowe (1997) and Lowe (2004).

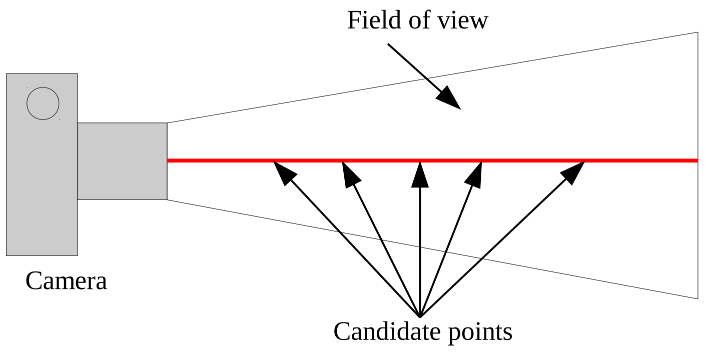


Figure 5.2: The location of a feature in a single two dimensional image does not reveal its location in three dimensional space.

If the same feature has been located in two different images while the positions where the images were taken, and the parameters, as well as the orientation of the camera were known, it is possible to locate the feature in 3D space. This is illustrated in Figure 5.3.

The problem becomes more complex if the camera positions are unknown. Depending on whether the camera parameters are known and whether it is known that the same camera setup was used for all recorded images, an initial image pair needs between six and eight feature correspondences and each added feature requires between two and three correspondences (Astrom et al., 1999; Hartley & Zisserman, 2002). Figure 5.4 illustrates two feature correspondences in three images.

Small deviations in the detection of features and the quality of the recorded images cause the features to not match up precisely. For this purpose the bundle adjustment algorithm is used to find the optimal geometry minimizing those errors among the analyzed images (Brown, 1976; Triggs et al., 2000). Repeating this procedure for several images results in a point cloud of the recorded scenery.

Finally the density of the resulting point cloud can be increased by using dense stereo reconstruction methods such as the patch-based multi-view stereo algorithm by Furukawa & Ponce (2010). For an overview of such methods the reader is referred to the paper by Ahmadabadian et al. (2013).

Tuttas et al. (2014) used photogrammetry to monitor the progress of construction projects using building information models (BIM). The work was later extended by using geometric constraints which were derived from BIMs in order to fill in missing informa-

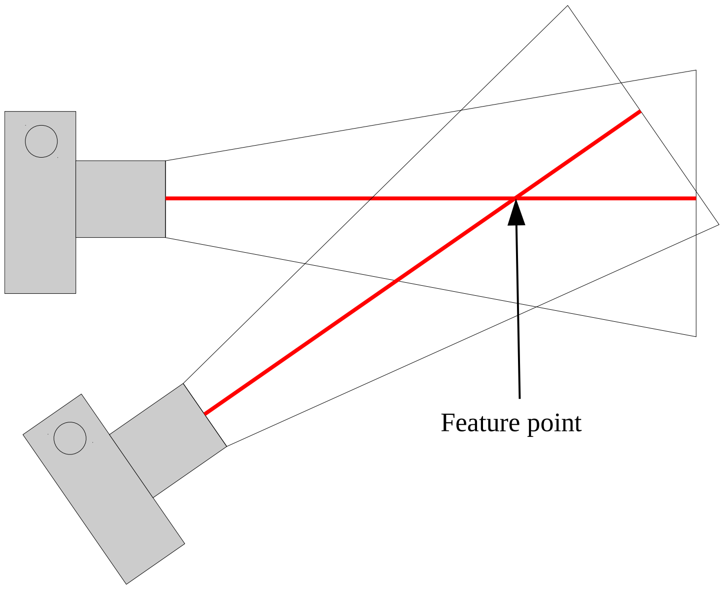


Figure 5.3: The locations of a feature in two two dimensional images reveal its true location, given the locations and parameters of both cameras are known.

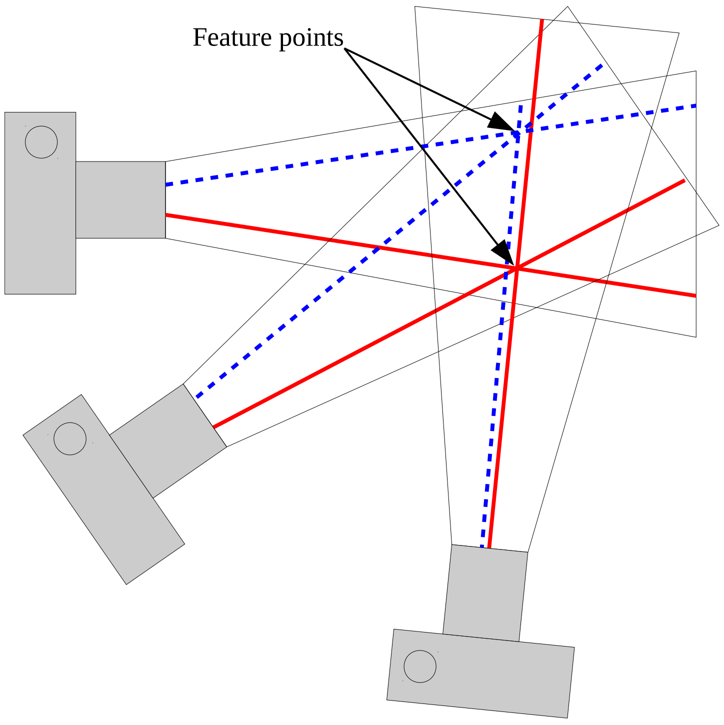


Figure 5.4: Example for two features in three images.

tion (Braun et al., 2015a,b; Stilla et al., 2015; Tuttas et al., 2015).

A method for measuring excavated volume from photogrammetric point clouds of excavation pits has been developed by the author and is presented in Section 6.1.2.

5.5 Barcodes and QR-Codes

Another common approach for material tracking on construction site is the placement of QR or barcodes on parts. The personnel, placing the part, then scans the codes to mark the respective activities as done. Such an approach was first proposed by Echeverry (1996), later refined and combined with geographic information systems (GIS) by Cheng & Chen (2002) and are still subject to ongoing research. (Lin et al., 2014; Wang et al., 2014). The approach only allows the tracking of parts that are preproduced and placed on site during the building process. Parts that are produced on site, such as concrete parts can not be monitored using this approach. Furthermore a code can be damaged during placement of the part, or be painted over in the following building process. Furthermore the process of placing and scanning the codes can hardly be automated.

5.6 Radio-based tracking

The use of radio-based tracking techniques on the other hand is more feasible for automation. In practice three different types of radio based tracking techniques are used.

5.6.1 GPS

The global positioning system (GPS) is based on tracking devices which can be located with a few meters of deviation. In 1995 when the GPS system was officially activated, one of the first commercial use cases was the tracking of heavy agricultural machines (O'Connor et al., 1995). This approach was first presented in a construction site environment by Naresh & Jahren (1997), where trucks were equipped with GPS trackers. At this point equipping a single truck with this technology created costs of several thousand dollars and was therefore hardly feasible on a large scale. When the technology became more widespread and costs dropped drastically, its use became more realistic and new approaches were envisioned. Oloufa et al. (2003) presented a method to not only track trucks, but also on-site equipment and additionally showed that the method can be used to prevent collisions. Furthermore Li et al. (2005) showed that the technology can reduce waste, improve efficiency, and reduce costs as a result. The idea proceeded to also capture the positions of materials on the site, as presented by Song et al. (2005). Pradhananga & Teizer (2013) presented methods for detailed spatial-temporal analysis of the obtained data, to improve efficiency of the on-site operations even further. All those technologies are widely used in many areas today. However, since the costs of GPS trackers still aren't negligible, the technology still isn't efficient for material tracking, especially for small parts.

5.6.2 RFID

A much cheaper approach to tracking materials on-site is radio frequency identification (RFID), which is based on passive trackers which are very cheap to produce and have a tiny form factor. If they are activated by surrounding radio waves, they broadcast their hardcoded ID which can then be received by a reader device. This allows to register RFID tags in the area surrounding a reader. Jaselskis et al. (1995) presented different possible applications of RFID on construction sites. Similar to GPS, a truck can be equipped with a RFID tag, but in comparison it cannot be continuously tracked with this technology, but the times it passes certain points, referred to as gates, can be logged. This would for instance be the times it enters and leaves a construction site. Furthermore RFID tags can be cast into prefabricated concrete parts, whose entrance onto the site can then be logged. Moreover all tagged and placed parts on the site can easily be verified by a walk-through using a RFID reader device, logging all surrounding tags (Song et al., 2005). Costin et al. (2012) developed this technology further into a system that automatically reports the progress of ongoing activities based on the location of RFID tags placed on materials and personnel. Kiziltas et al. (2008) analyzed the practical implications of this technology. They also discuss the active variant of RFID, which is commonly implemented using ultra-wideband technology. Klaubert & Günthner (2011) presented a holistic RFID-based approach for construction information and communication systems.

5.6.3 Ultra-Wideband Radio

Compared to passive RFID, ultra-wideband (UWB) technology is active. The UWB device sends radio waves to allow a receiver infrastructure to track its location. Compared to passive RFID, UWB allows precise location tracking in three dimensions. Furthermore, it is more precise than GPS and works in occluded areas, such as indoor spaces (Teizer et al., 2007b). As materials are often stored in indoor storage places, GPS tracking is not possible in those cases. The system has been used for performance measurements (Cheng et al., 2011; Saidi et al., 2011), resource tracking (Teizer et al., 2007a), and personnel tracking (Sahinoglu, 2008).

5.7 Visual Tracking

Compared to the previously explained methods for tracking objects on construction sites, the use of visual information is perhaps the most intuitive method. It is however much more complicated to be automated.

5.7.1 Range Imaging

One approach to visual tracking, which is similar to the previously explained method of 3D scanning, is range imaging. A range image is a two dimensional image, that is enhanced by adding depth information to each pixel. There are different methods to generate this depth information. The best known system is the use of stereo triangulation, which is also used by the human vision apparatus and accordingly for recording 3D movies. Another common technique is the use of structured light, projecting a special light pattern onto the scene which can be used to retrieve depth information. The most advanced methods use a coded aperture (Levin et al., 2007) or so-called time-of-flight (ToF) methods (Oggier et al., 2004). Time-of-flight methods create a single pulse of light and measure the time it takes to be reflected from the different surfaces in the scene. As ToF works in real-time and is very precise it can be used for real-time tracking of resources on site.

Teizer et al. (2007a) presented the use of range imaging for maintaining a occupancy grid of a construction site, while Kahlmann et al. (2007) presented its use for personnel tracking. The idea was extended towards the creation of as-built building models by Bosche & Haas (2008). Teizer (2008) and Gonsalves & Teizer (2009) proposed the use for safety analysis and human motion analysis. An approach for 3D object detection for heavy equipment tracking was introduced by Son et al. (2010). Ng et al. (2005) describe a new technology, the plenoptic camera, which records images or video in three dimensions using a single camera. As the resolution of such cameras is still limited and the costs are very high, they were not yet subject to research in the area of construction tracking.

Generally the use of range imaging techniques is cost intensive and creates large amounts of data that are hard to process automatically. Therefore two-dimensional imaging techniques are still in the focus of researchers.

5.7.2 Video Analysis

Many processes on construction sites can be monitored visually. A human viewing the site all day from a vantage point, would be able to monitor most processes. Especially in the shoring and excavation phase, all involved personnel and machines are mostly visible. Therefore computer vision-based techniques are of great interest for monitoring those processes automatically. The use of computer vision techniques on video recordings has a long history of research. In the seventies, when computers first became capable of processing video sequences Flachs et al. (1976) and Flachs et al. (1977) presented an automatic object tracking system that formed the basis of decades of following research. Later they managed to run the tracking system in real-time (Gilbert et al., 1980). Despite those early advances in video tracking technology, there are still many challenges subject to ongoing research. Song et al. (2006) published an approach to video based tracking on construction sites to both measure and improve the performance of construction operations. Teizer & Vela (2009), Yang et al. (2009) and Yang et al. (2010) present an approach for automated video-based personnel tracking. A possible approach to assess the productivity of construction operations is to identify the status of different activities by the postures of involved workers (Ray & Teizer, 2012). Furthermore the tracking of resources and materials can be subject of video monitoring approaches (Gong & Caldas, 2010). On many construction sites tower cranes play a key role. Yang et al. (2011) presented an approach for vision-based crane tracking for the analysis of construction activities. Another common key activity of construction projects are earthwork processes. Ogunmakin et al. (2013) presented a method using video cameras to quantify interactions between excavators and dump trucks. A holistic system, including video based monitoring of the construction activities was presented by Leung et al. (2008).

Section 6.2 presents a novel method, that is based on video-based tracking, as well as photogrammetry.

5.8 Summary

This Chapter presented an overview of the state-of-the-art of progress tracking technologies that can be used on construction sites. While manual protocol-based tracking is still widely used in practice, it is also possible to automatically record machine data and extract information about the executed activities from that data. It has been shown that this information can be used to calculate performance factors for the involved resources. Furthermore laser scanning and photogrammetry have been proven effective in generating point clouds with sufficient quality to detect construction elements and measure the as-built performance. Additionally video-based approaches were presented, and have been shown to be effective in tracking construction assets. The presented methods have not been used to monitor excavation processes by obtaining absolute measures. While there are methods to count the number of dump trucks entering and leaving a site, the exact volume of excavated soil is a more precise measure. The following Chapter introduces improved vision-based methods for this purpose.

6 Improved Vision-based Progress Monitoring of Excavation Processes

"There's nothing like matter of fact; seeing is believing." - Arbuthnot (1712)

Many construction projects strongly depend on shoring and excavation processes. Main actors in those processes are excavators, dumpers, and shoring machines. There are several approaches in literature using radio frequency based tracking of machines and trucks, such as RFID or GPS (Cheng et al., 2011; El-Omari & Moselhi, 2011; Klaubert et al., 2010; Lu et al., 2007; Oloufa et al., 2003; Saidi et al., 2011). However, in practical excavation projects the dump trucks are often hired on demand on a daily basis and are therefore difficult to equip with a radio chip. Furthermore counting the number of dump trucks or the movement of machines does not yield detailed progress data for the involved processes. This Chapter introduces two vision-based techniques for this purpose, which can be used to obtain detailed progress information about excavation processes and determine underlying problems when delays occur. The resulting data can be used to update performance factors for the tracked processes as described in Section 2.1, as well as other similar processes that need to be performed in the future. Furthermore, the knowledge about the source of the delay allows the decision makers to put countermeasures into place. Additionally it allows feedback of real-time data into a construction project model and therefore facilitates updated prognoses and reactive optimization of construction schedules.

6.1 Excavation Tracking using Point Clouds

The most vital measure for the progress of an excavation process is the amount of excavated soil, which can be measured by the volume of the excavated pit. As the excavated volume is surrounded by a surface of soil, it can be described by a set of points lying on that surface. When multiple such sets are known for different time instances, the volume of the soil excavated within the time period between the samples can be determined. The two basic methods for generating such point sets, or point clouds, are described in the following two Sections, followed by methods to extract the desired metrics from the generated point clouds. As a result up-to-date performance factors for the involved processes can be calculated.

6.1.1 Laser Scanning

As described in Section 5.3 laser scanning can be used to generate three dimensional point clouds. The laser scanner is placed near the excavation pit to record a laser scan. Due to the concave shape of an excavation pit, a single laser scan can hardly capture the full surface of the pit, as is illustrated in Figure 6.1. Additionally machines, such as excavators and dumpers commonly operate within the pit and create occluded areas, as illustrated in Figure 6.2. Those problems are commonly solved by performing multiple laser scans and register the resulting point clouds to stitch them into one (Makadia et al., 2006). This stitching operation can be time intensive and create errors (Rabbani et al., 2007). In other scenarios the stitching is favored by marker points that are placed in the scene (Bornaz & Rinaudo, 2004). In case of excavation sites this is troublesome due to the constant change of the excavation pit. Additionally, laser scanning requires trained personnel and expensive equipment (Flood, 2001).

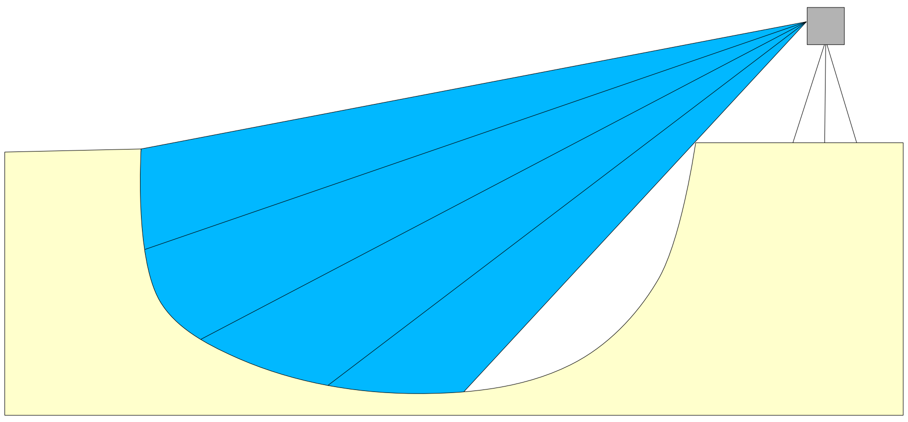


Figure 6.1: Laser scanning an excavation pit. The laser scanner is placed in the right of the image, illustrated by a gray square. The blue area shows the field of view of the laser scanner. Due to the shape of the pit, the scanner can hardly capture the entire soil surface.

6.1.2 Photogrammetry

Compared to laser scanning, photogrammetry is performed using a standard digital camera, as described in Section 5.4. A set of photographs is transformed into a point cloud by matching features among the individual images. A human operator surrounds the scene while taking photographs at regular intervals. In case of an excavation pit, all visible areas of the pit are photographed in an overlapping tile manner, as illustrated in Figure 6.3, from different positions surrounding the pit. It is important that the same areas are captured from multiple perspectives as is illustrated in Figures 6.4 and Figures 6.5. As all the photos taken by the operator should capture even partly occluded areas, the photogrammetric point cloud generated from the photos covers the entire pit unless there are

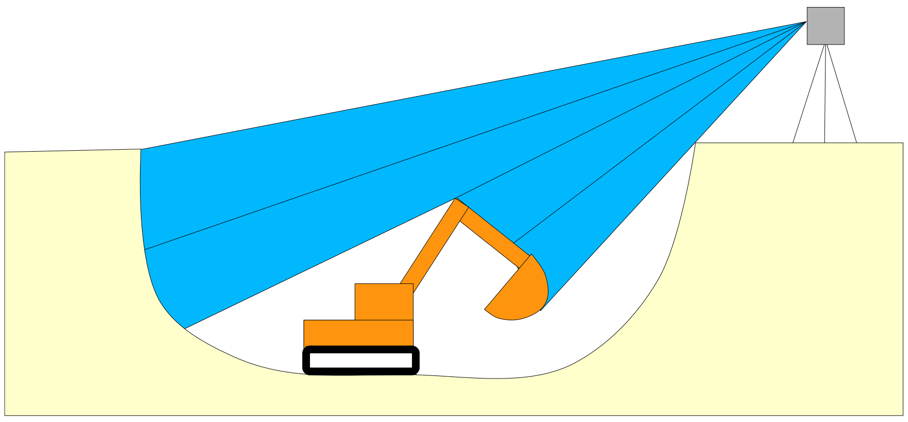


Figure 6.2: Laser scanning an excavation pit. The laser scanner is placed in the right of the image, illustrated by a gray square. The blue area shows the field of view of the laser scanner. Due to the shape of the pit and an excavator occluding parts of the area, the scanner can hardly capture the entire soil surface.

areas not visible from any point outside of the pit. Additionally photos from inside the pit can be taken.

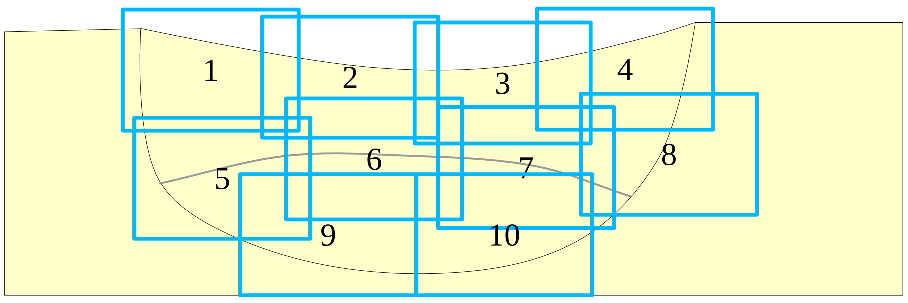


Figure 6.3: The operator's perspective, looking into the excavation pit, while taking photos for photogrammetric point cloud generation. The ten blue rectangles represent the photos taken by the operator. The photos need to overlap such that they share common features.

Figure 6.6 shows a practical example where photos haven been taken surrounding the pit and additional pictures were taken from inside the pit. Furthermore the pictures illustrated in the lower right were taken from an elevated angle off a tower crane. After the images have been processed by the structure from motion algorithm VisualSFM (Wu, 2011), bundle adjustment (Wu et al., 2011), and the patch-based multi-view stereo algorithm (Furukawa & Ponce, 2010) the dense point cloud representation shown in Figure 6.7 was generated.

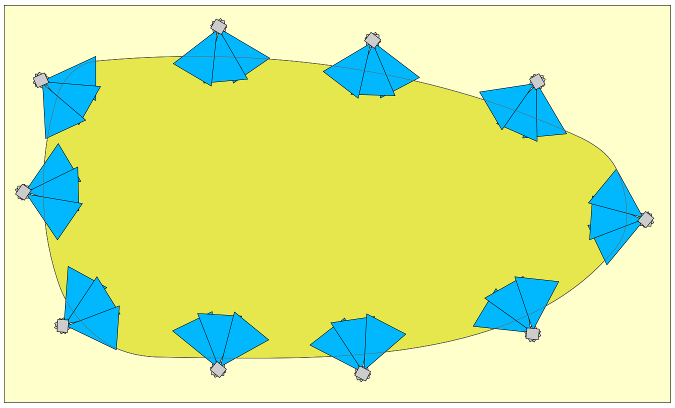


Figure 6.4: Taking photos of an excavation pit for photgrammetric point cloud generation. The darker area in the center is the excavation pit. Each blue triangle shows the orientation of a photo that was taken. The operator moves around the pit taking photos of all currently visible areas of the pit.

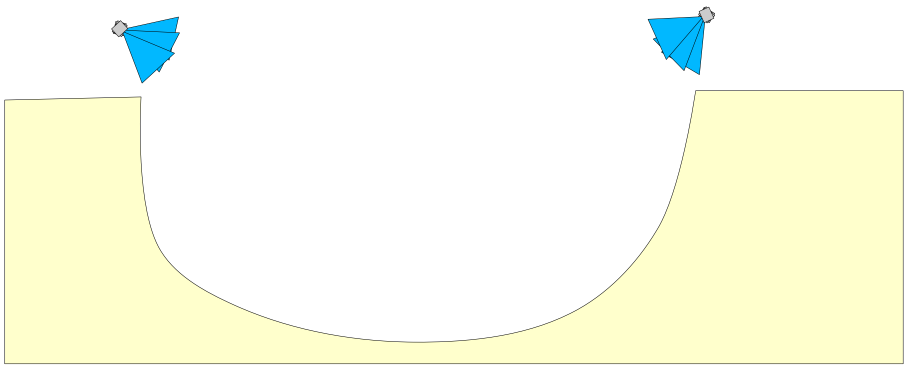


Figure 6.5: Cutting plane of excavation pit while taking photos for photogrammetric point cloud generation. Each blue triangle shows the orientation of a photo that was taken. At each location the operator takes photos of all currently visible areas of the pit.

6.1.3 Processing of Point Clouds

After a point cloud is generated by the algorithms described in the previous Section and in Section 5.4 it contains noise and includes unwanted geometry, created by machines, as well as structures, and buildings surrounding the excavation pit. Furthermore the individual points of the cloud need to be post-processed in order to get the desired volume measure.

6.1.3.1 Clustering

As a first step the point cloud is cleaned from unwanted points outside the region of interest. This is performed using conditional euclidean clustering where points are combined into different clusters, similar to the approach by Trevor et al. (2013). Each point that is

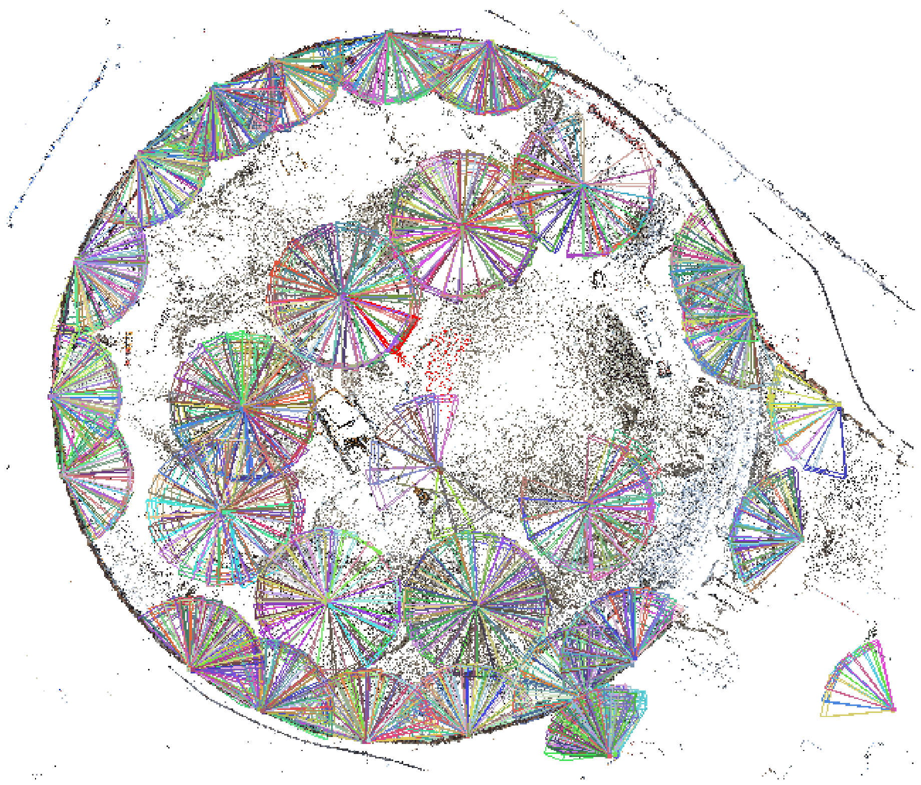


Figure 6.6: Camera positions of photos taken on an actual construction site. The circular area is the excavation pit and each pyramid represents a photo that was taken.

added to a cluster needs to be within a certain euclidean distance to another point within the cluster. The resulting clusters differ in size and the main cluster which forms the excavation pit can be identified as the largest one. The maximum distance of a point to a cluster is a parameter for this operation, but can be determined automatically based on the overall size of the point cloud (Shi et al., 2011). All other clusters are removed from the point cloud. The result of this process is shown in Figure 6.10. A survey of other possible approaches has been published by Nguyen & Le (2013).

6.1.3.2 Cleaning

As a next step the point cloud is analyzed by generating a vertical histogram of the points contained in each layer of the point cloud. The thickness of the layers only has a small effect on the results, but should be sufficiently large to have points in every layer, but sufficiently small to get the desired histogram information. Taking between 10 and 50 slices over the full vertical range of the point cloud, depending on the depth of the excavation pit, has proven to be a good measure. Figure 6.9 shows the histogram for the point cloud in Figure 6.8. The plot shows that the majority of the points lie in the range between the elevations 5 and 36. The large spike around an elevation of 30 indicates the bottom of the excavation pit where most points are located. In case of a non horizontal ground plane, this spike would be less obvious. The other end of the plot, where the number of points

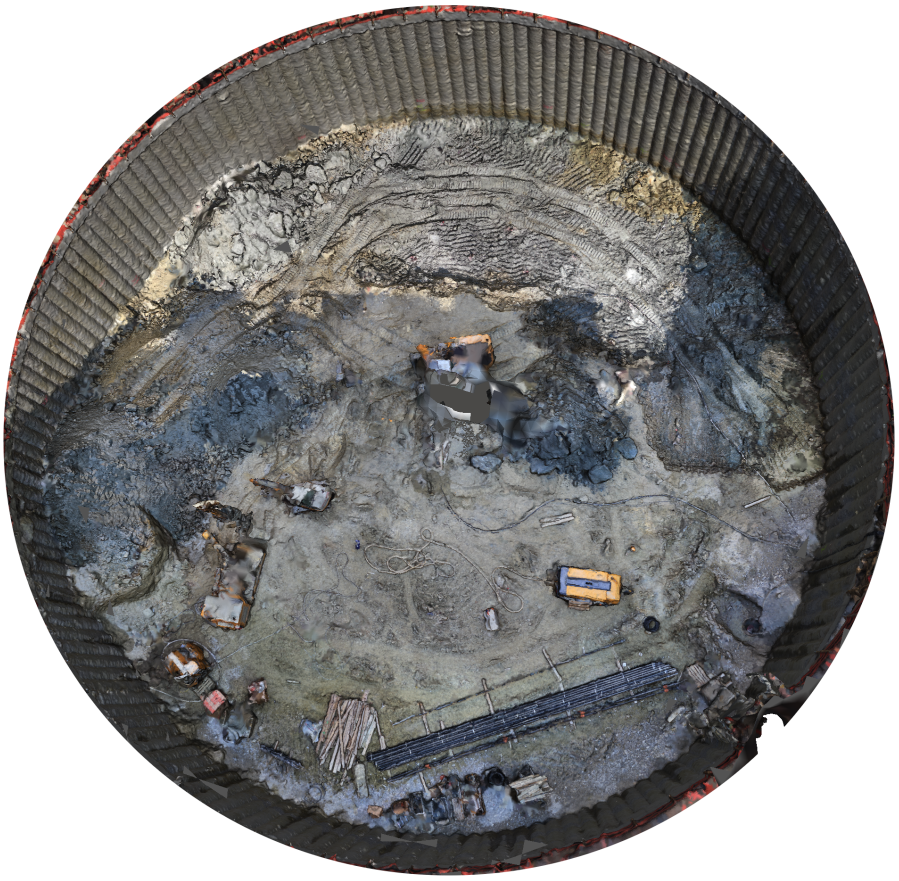


Figure 6.7: Point cloud generated from photographs illustrated in Figure 6.6.

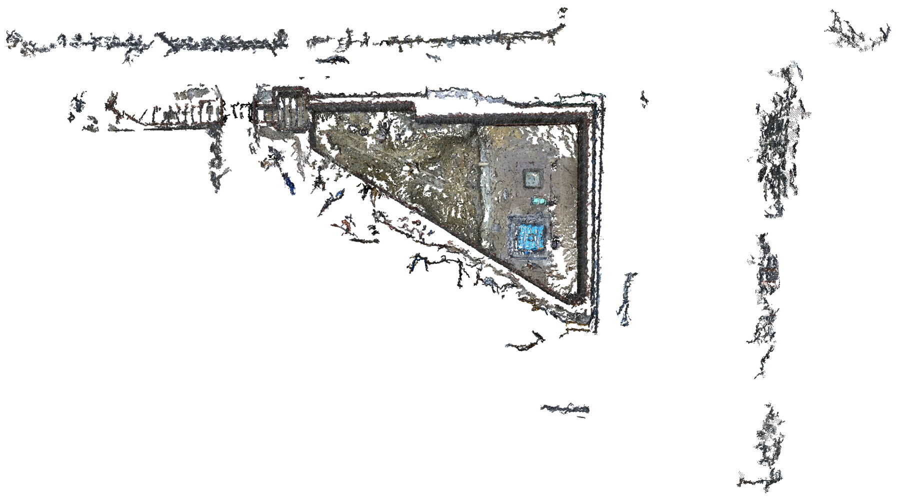


Figure 6.8: Raw unprocessed point cloud. Structures outside the excavation area have been included into the point cloud.

flats out at an elevation of 5 marks the upper end of the excavation pit. This mark can be used for the volume calculation in the following Section. The point cloud resulting from this procedure is illustrated in Figure 6.10.

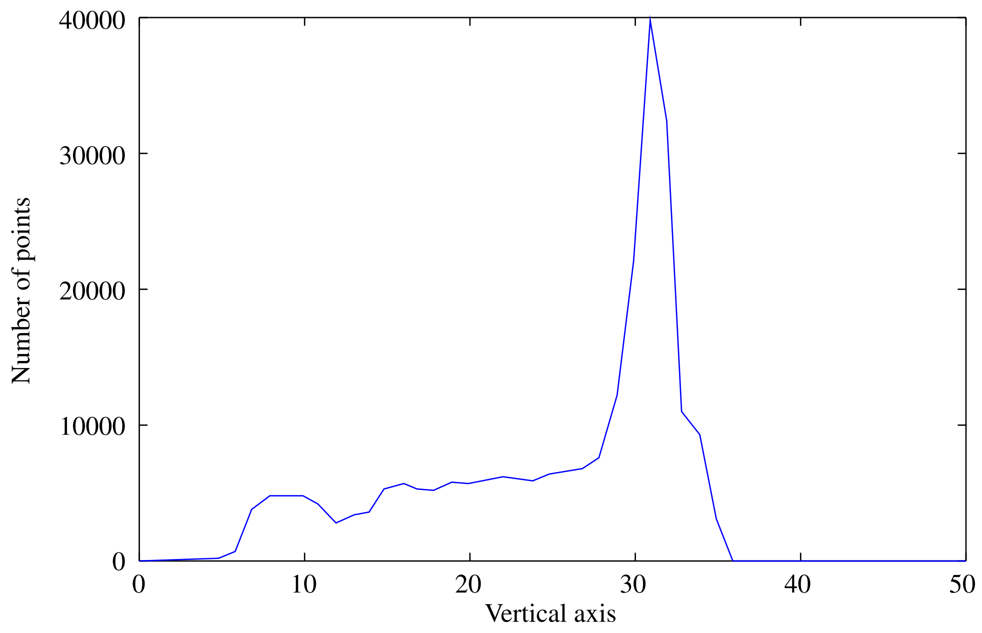


Figure 6.9: Vertical histogram of point cloud. The graph shows the number of point on the different elevations in the point cloud. The bottom layer within the point cloud is clearly visible as the spike at around an elevation of 30. The surrounding ground layer can be estimated to be at the left of the graph where the histogram flats out.

6.1.3.3 Volume Calculation

In order to calculate the volume of the excavated soil, the correct size of the point cloud has to be known. When the point cloud was generated by a laser scanner, the size is known. If however the point cloud was generated by photogrammetry, it needs to be scaled to the correct size. This can either be done manually by taking a known measure, such as the length of the excavation pit, an included bored pile wall, or automatically by placing marker points at known locations. The markers can be located within the point cloud and the scale of the cloud can then be determined by comparing their distance in the real world with their distance in the point cloud.

Once the scale of the point cloud is known, there are different ways to calculate the volume of the excavation pit. The most intuitive way is the create the convex hull of the point cloud. This can be done using the quickhull algorithm by Barber et al. (1996). In most cases the convex hull is larger than the actual volume, especially with irregularly shaped excavation pits, such as the u-shaped pit in Figure 6.11, the error is significant and cannot be neglected. Figure 6.12 shows the convex hull of the point cloud in Figure 6.10.

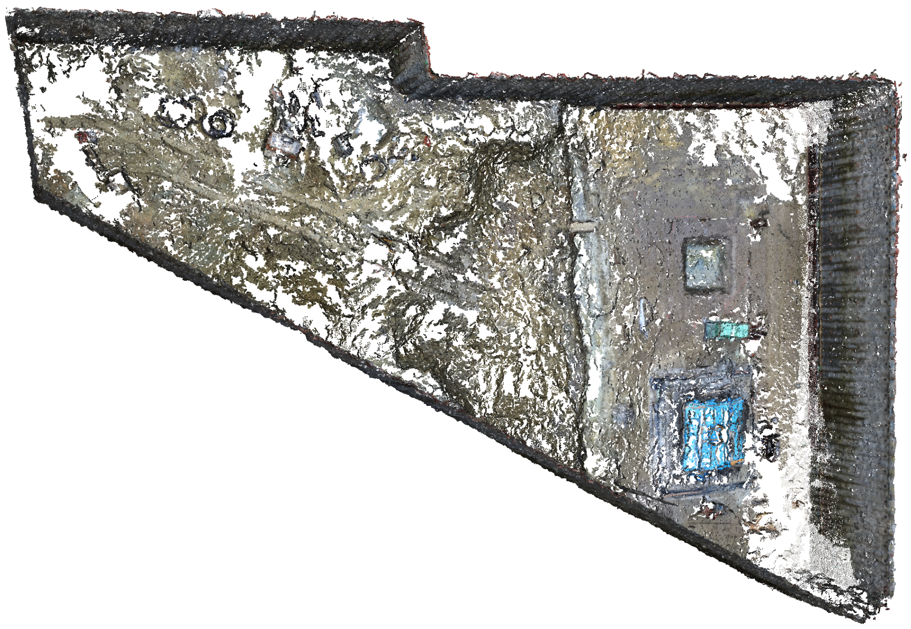


Figure 6.10: The point cloud from Figure 6.8 after cleaning. Points outside the excavation area have been removed.

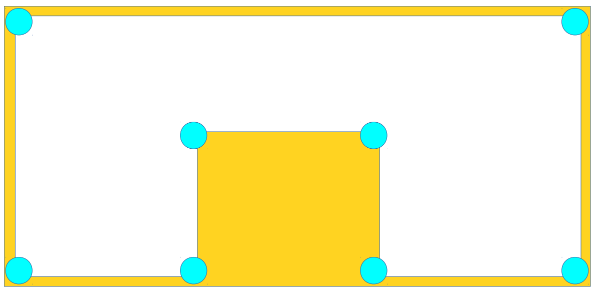


Figure 6.11: The convex hull of an u-shaped excavation pit would have a much higher volume than the excavation pit. The white area is the actual excavation pit, the orange area is the convex hull.

A more advanced approach to determine the volume of an object represented by a point cloud would be to generate a mesh from the point cloud. Calculating the volume of a closed mesh is trivial and can be done using signed tetrahedron volumes (Hoppe, 1999). However, the point cloud does not represent an object with a closed surface in case of an excavation pit since it is open to the top. Meshing algorithms such as the Poisson surface reconstruction algorithm by Kazhdan et al. (2006) will close the top opening, creating a bubble on top of it as is shown in Figure 6.13. The volume of such a mesh does not represent the volume of the excavation pit. The ball pivoting algorithm by Bernardini et al. (1999), on the other hand, does not close the top opening at all, as shown in Figure 6.14. It does therefore not allow the calculation of the mesh volume. Additionally the algorithm can leave holes within the mesh.

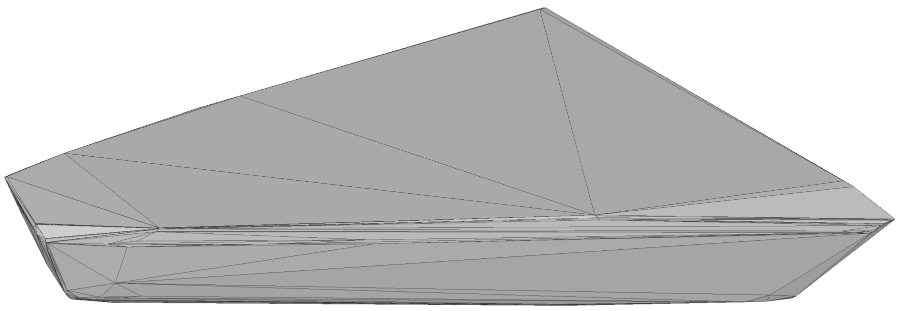


Figure 6.12: Convex hull of the point cloud in Figure 6.10. The hull was created using the quickhull algorithm by Barber et al. (1996).

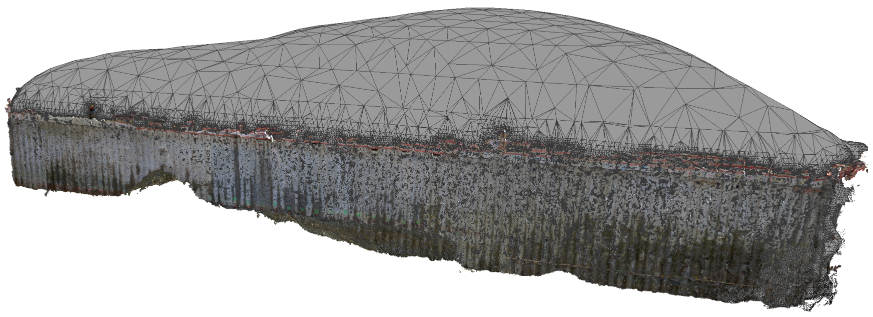


Figure 6.13: Meshed surface of the point cloud in Figure 6.10. The surface was generated using the Poisson surface reconstruction algorithm by Kazhdan et al. (2006).

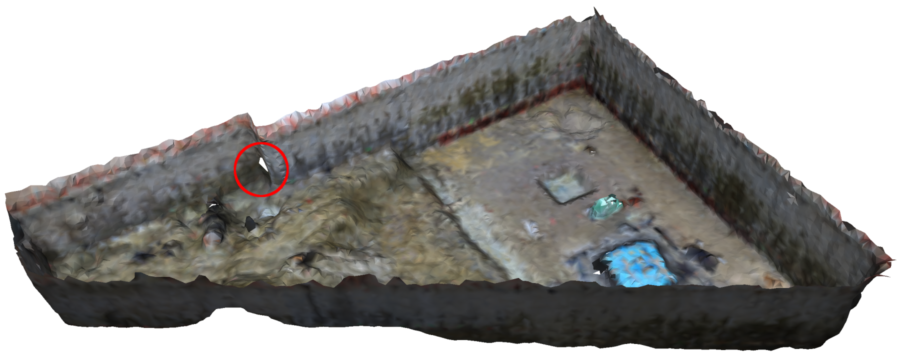


Figure 6.14: Meshed surface of the point cloud in Figure 6.10. The surface was generated using the the ball pivoting algorithm by Bernardini et al. (1999). The red circle shows a hole that was left open.

Based on the observation that the walls of an excavation pit hardly allow any overhang, because overhanging soil would cave in, a more specialized algorithm was developed[7]. The elevation of the surrounding soil is known due to the histogram analysis described in Section 6.1.3.2.

Under the assumption of walls without overhang the point cloud can be reduced to its ground layer by removing all vertical duplicates of points, except the lowest ones. Depending on the density of the point cloud a tolerance can be defined in order to determine which points cover each other. Once all vertical duplicates have been removed, the point cloud is reduced to a ground layer as shown in Figure 6.15.

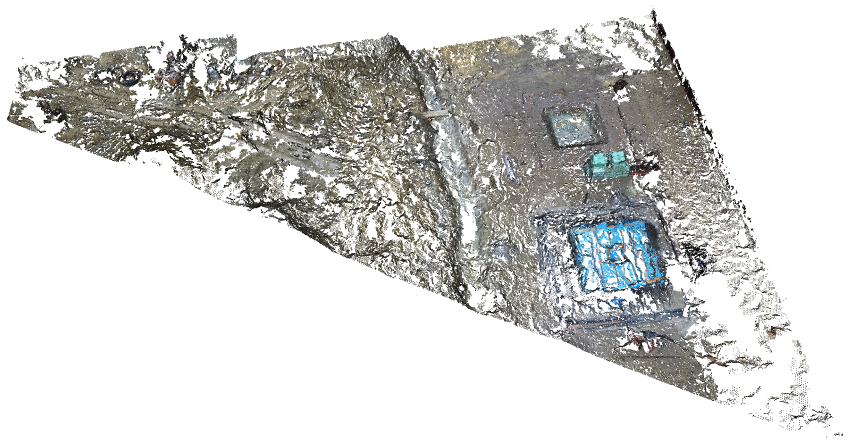


Figure 6.15: The point cloud from 6.10 after all vertical duplicate points have been removed.

Since the ground layer could be irregular or have gaps, it can be homogenized and thinned out. First of all all gaps are linearly interpolated, creating a refined ground layer as is shown in Figure 6.16. Then the cloud can be thinned out to a lower density with points at regular distances as is shown in Figure 6.17. The ground layer can then be triangulated into a mesh such that it forms a set of triangles. Each of those triangles is then extruded towards the elevation of the surrounding soil which was determined by the histogram analysis. Each triangle creates a prism of which the volume can be calculated. The sum of all prism volumes adds up to the sum of the excavation pit.

The described method was successfully tested on multiple construction sites to determine the excavated volume at multiple time steps. Detailed results are described in the context of multiple case studies in Section 8. Situations where the surrounding ground is very uneven or at a steep slope might cause the results to decrease in quality. Such situations need further field studies in order to be evaluated.

[7]The idea for this algorithm was developed in close cooperation with Benjamin Steer in the context of his interdisciplinary project "Processing of Point Clouds of Excavation Sites to obtain Progress Information".

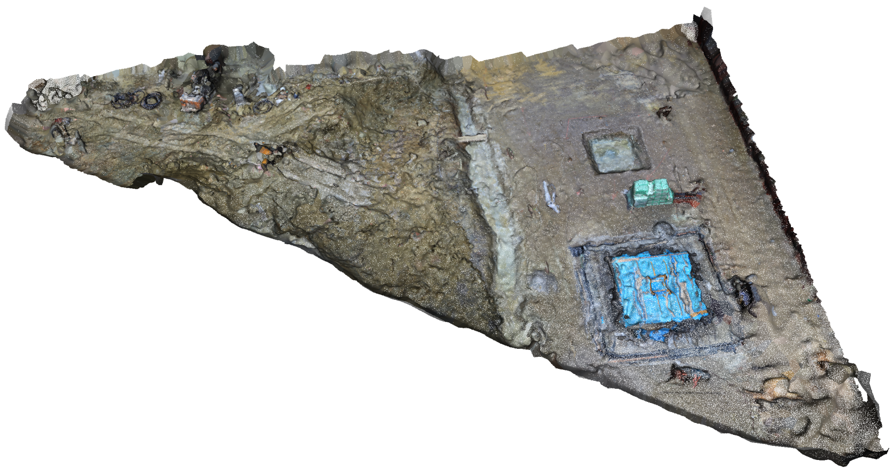


Figure 6.16: The point cloud from Figure 6.15 after refinement to fill holes.

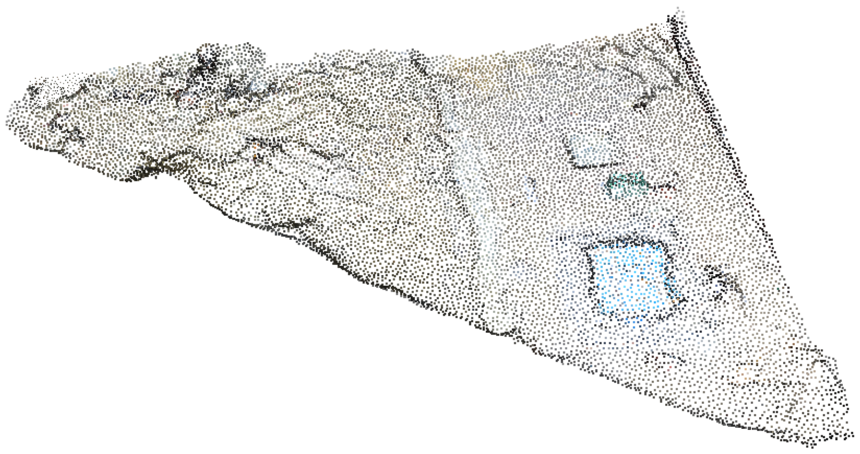


Figure 6.17: The point cloud from Figure 6.16 after resampling to reduce the number of points to be evenly distributed.

6.1.4 Comparison of Laser-scanning and Photogrammetry

During a comparison study of laser scanning and photogrammetry, several key differences were determined. The laser scanner needs fewer accessible points on the site, and modern scanners can capture the point clouds in less than a second (Kusnoto & Evans, 2002). The laser scanner however is an expensive piece of equipment and requires trained personnel. For instance, a Leica ScanStation P40 costs more than 100.000EUR. Furthermore it can be sensitive to weather conditions, such as rain. As a single laser scan can rarely capture the entire excavation pit, the resulting point clouds need to be stitched. This requires manual work or the placement of marker points on the site. Additionally laser scanning is, due to the fewer perspectives that are captured, more sensitive to occlusions as shown in Figure 6.2. Photogrammetry on the other hand only requires a standard digital camera and a non professional operator. In the performed field studies, a basic explanation of the scheme described in Section 6.1.2 sufficed for the personnel to take the right photos. The approach however requires the area around the excavation pit to be accessible. Large spans around the pit that are inaccessible can cause the structure from motion approach to fail. Furthermore puddles of water in the excavation pit create errors. If they are small, they can be filled in using the approach described in Section 6.1.3.3. Another problem that may occur is snow. If the pit is covered in snow, the feature extraction of the structure from motion algorithm will not be able to gather sufficient data to triangulate a dense point cloud. In summary these limitations did not create major problems during the case studies.

6.2 Video-Based Tracking of Excavation Processes

In the previous Section an approach to measure the progress of excavation processes was introduced. However, a detected delay cannot be traced back to the underlying problem that caused it. Ogunmakin et al. (2013) presented a system to monitor machines involved in excavation processes and quantify their interactions. The work focuses on tracking excavators and dumpers in video streams and classify their states. This includes counting the number of dumpers that enter and leave the site as well as the number of bucket loads an excavator fills into a dumper. This allows the estimation of the excavation productivity. However, as the precise volume of a bucket load and of a dumper load cannot be determined, the approach suffers from a cumulative error when measuring the overall excavation progress.

The weaknesses of both approaches can be compensated if they are used in synergy[8] as is illustrated in Figure 6.18. The cumulative error of the video analysis approach can be corrected by the absolute volume measurements provided by the photogrammetry, while the underlying problems of delays can be identified using the video analysis approach.

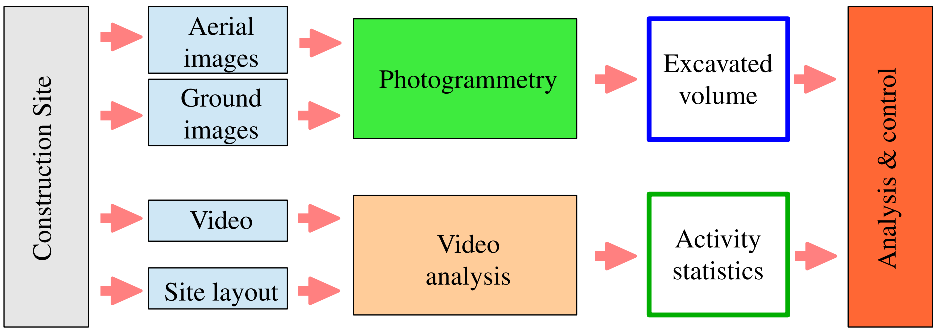


Figure 6.18: Overview of the system combining the presented photogrammetry approach with the video analysis system by Ogunmakin et al. (2013).

[8]This synergetic approach was developed in close cooperation with Gbolabo Ogunmakin, Patricio A. Vela from the Georgia Institute of Technology, André Borrmann, and Jochen Teizer.

As previously described, the photogrammetry approach can be used to generate time series of the excavated volume of an excavation pit. This information can be used to calculate performance factors for the involved resources as described in Section 2.1. However, changes in the performance cannot be attributed to the individual machines.

While performance can be defined as the amount of time spent working, a more reliable measure of performance is the amount of work performed in a certain time. In case of excavation processes this commonly translates to the volume of soil removed from the site per day. When the involved machines are monitored it is possible to detect what ratio of the time the machines are idling. On the other hand a machine can be working all day but not be productive. For instance soil could be moved around within the excavation pit while not dumper is present to speed up the loading process once a dumper arrives at the site. This however does not increase the volume of the excavation pit. Furthermore the excavator could be working very slow but still be active all day. In order to get an absolute measure on the productivity of the entire excavation process the amount of soil removed from the site is the most accurate metric for this performance factor.

The video analysis approach adds detailed information about the machines states over time to the system. It distinguishes four different states of the dump trucks: absent, static, moving, and filling. The individual states are illustrated in Figure 6.19. Absent means that no dump truck is visible in the video stream. Static means that a dump truck is present, but not interacting with the excavators. Moving means that the dumper is either moving in or out of the construction site. Filling means that the excavator is currently filling the dumper with soil. When a decrease in performance is observed, it can be attributed to the individual machines based on these states. If absent, the dumpers are unavailable and a probable cause of the delay is a traffic jam on the way between the excavation and the dump site. Employing additional dumpers can help solving the problem. If dumpers are in the static state, hence queuing up on the excavation site, the excavators are the bottleneck in the process. Deploying additional excavators can solve the problem, given there is sufficient space.

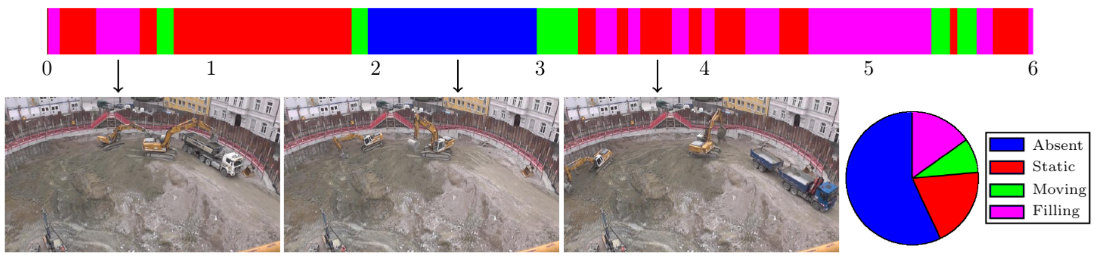


Figure 6.19: Dump truck state estimates for a video segment of 6 minutes duration and the activity states in a pie chart (Bügler et al., 2016).

The synergetic approach was successfully tested on two large excavation sites as described in detail in Section 8. The main problem of the photogrammetry approach were missing measurements due to inclement weather. Though, due to the absolute measurements obtained by the volume calculation, the missing data points could be interpolated. Another problem is that the processing of the image data to generate the point clouds and calculate the excavated volume caused a delay of about half a day. The same problem occurred with the video analysis, where both real-time transfer and processing of the video data was not possible. On the upside, obtaining measurements of the previous day in the morning of the next day was feasible and sufficient for decision makers to react to observed problems. Errors in the video processing were mainly caused by temporary occlusions such as machines moving in front of each other. This can be addressed by placing multiple cameras at different locations. Generally, even when occlusions caused high error rates in the video analysis, the acquired measurements still correlated with the actual activity and provided accurate estimates of the performance. Future research should address the use of multiple cameras in conjunction and speeding up video processing to near real-time.

In summary, the video analysis approach adds both temporal information as well as resource attribution to the data acquisition approach. In a reactive scheduling system where delays need to be attributed to processes or involved resources and the respective performance factors it is vital to obtain accurate metrics on the progress of the involved processes as well as to determine reasons for observed delays. The photogrammetry approach in combination with the the video analysis approach proved effective for this purpose.

6.3 Summary

This Chapter introduced two novel methods for the monitoring of excavation processes. Photogrammetry is the creation of point clouds based on photographs taken of an object. This approach has been used to generate point clouds of excavation pits of which the volume could be calculated afterwards. While this process showed to be sensitive to reflective surfaces and snow, field studies showed that it is a viable approach to monitor the progress of excavation processes.

However it is not possible to detect underlying reasons for observed delays as the capturing of such point clouds is only feasible with low temporal resolution. In order to address this issue a second visual approach, video analysis, was introduced. This approach allows to quantify the interactions between excavators and dump trucks, measure idle times and count the number of shovel loads and dump trucks entering and leaving the site. Combining the two sources of information allows to determine the absolute productivity of the excavation process as well as determining reasons for periods of low productivity.

This process results in performance factors that can be updated in regular intervals. Additionally the variations on productivity can be quantified in terms of uncertainty. Both the performance factors and the probability distributions they are subject to can be used to update a reactive simulation model and produce updated schedules taking the newly arisen circumstances into account.

7 Data-driven Reactive Scheduling

"For every action, there is an equal and opposite reaction." - Newton et al. (1850)

This thesis presents the concept of reactive scheduling in the domain of construction projects consisting of excavation and shoring operations. The presented methods in combination form a synergic approach that is in focus of this Chapter.

7.1 Concept

The overall concept of the data-driven reactive scheduling approach presented in this thesis is illustrated in Figure 7.1. First, a model of the construction project is created based on the specific requirements of the project. This process is described in detail in Section 2.6. The modeling can be done on the basis of process models as discussed in Section 2.5. During this step, the right choice of granularity as described in Section 2.4 and the inclusion of uncertainty as described in Section 2.2 have to be considered. Including uncertainty into the model, given the required data is available, allows for more accurate predictions and can minimize the amount of re-planning required. The model also needs to include the required resources, precedence constraints and other required elements as described in Section 2.6.2. For this process Building Information Models can be utilized as described in Section 2.6.3 to extract elements that need to be built as well as precedence relationships among them.

Once a model has been created it can be used to generate schedules taking all specified circumstances into account. Since not every schedule will meet the requirements of the planner, a metric on how well a schedule suits the purpose is required as described in terms of objective functions in Section 3.2. The most commonly used metric in research is the makespan of the entire project, however in real projects costs, resource utilization and robustness (Section 3.2.1) commonly form a multi-objective optimization problem as is subject to Section 3.2.2.

After an objective has been defined, schedules can be generated as described in Section 3.4. As the underlying Resource-Constrained Project Scheduling Problem (RCPSP) is considered NP-Hard, heuristic algorithms are of high interest. This thesis introduced two new heursitic methods for optimizing schedules. The iterative limited-depth local

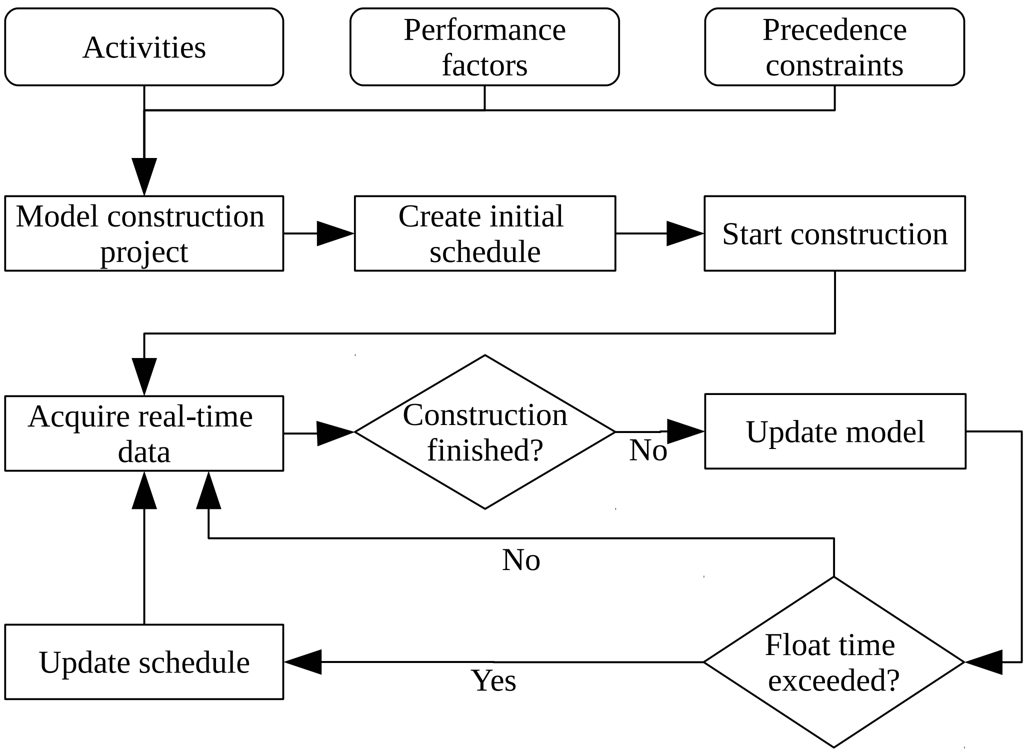


Figure 7.1: Flowchart of the overall concept of the reactive scheduling approach. An initial schedule is built on the basis of a model of the construction project. Once construction has started, real-time data is acquired. If construction is not yet finished, the model is updated and it is verified whether the float time of an activity was exceeded. If that is the case an updated schedule is generated. This process is repeated until construction has finished.

search algorithm in Section 4.3 and the adaption of the water wave optimization metaheuristic optimization algorithm by Zheng (2015), presented in Section 4.2. Both algorithms can utilize the novel pseudo-serial schedule generation scheme, which in turn is described in Section 4.1. This scheme combines the advantages of both common parallel and serial schedule generation schemes. Both presented novel scheduling algorithms can be used to generate an initial schedule based on the developed model as well as update the schedule once new information has been acquired. Updating a schedule is in essence the same process as generating a new schedule, except that all activities that have already been started in the past are locked into place during a schedule update. Furthermore no new activities can be scheduled into the past.

On basis of the initial schedule the construction project is started. During the course of construction, data is acquired for the individual activities, updating the model reactively. Excavation activities can be monitored using the presented photogrammetry and video-analysis approaches in Sections 6.1.2 and 6.2. As a result, delays of activities are observed and can furthermore give rise to the underlying performance factors of the involved machines as presented in Section 2.1. Performance factors are generally defined as the amount of work performed over time. The performance factors as well as the activity durations are commonly subject to uncertainty which can be estimated based on the collected data as described in Section 2.2. If for instance an excavator does not perform as expected because the soil conditions are different than initially measured, the performance factor of the machine is decreased. Therefore activities involving the excavator will be updated by extending their duration respectively.

Once real-time data has been acquired the updated performance factors and accordingly estimated activity durations are verified for exceedance of the float times which can be calculated using the Dori Algorithm (DA) which is introduced in Section 3.4.6. If the float time of an activity is exceeded, a re-planning process is initiated, generating a new schedule based on the newly arisen circumstances. If the uncertainty in the activity durations has been estimated in terms of probability distributions as discussed in Section 2.2 Monte-Carlo simulation as described in Section 2.2.6 can be used to obtain metrics on the effects of the uncertainty on the resulting schedules. Furthermore those metrics can be used to generate particularly robust schedules minimizing the amount of re-planning required in the future.

7.2 Illustrative Example

Figure 7.2 shows an illustrative example of a schedule of a simple construction project involving an excavation activity and four shoring activities. While activity A does not require any resources that are shared with other activities, all other activities require the same resource and only one instance of that resource is available. The depicted schedule is optimal for the problem as there is no other schedule with a shorter makespan.

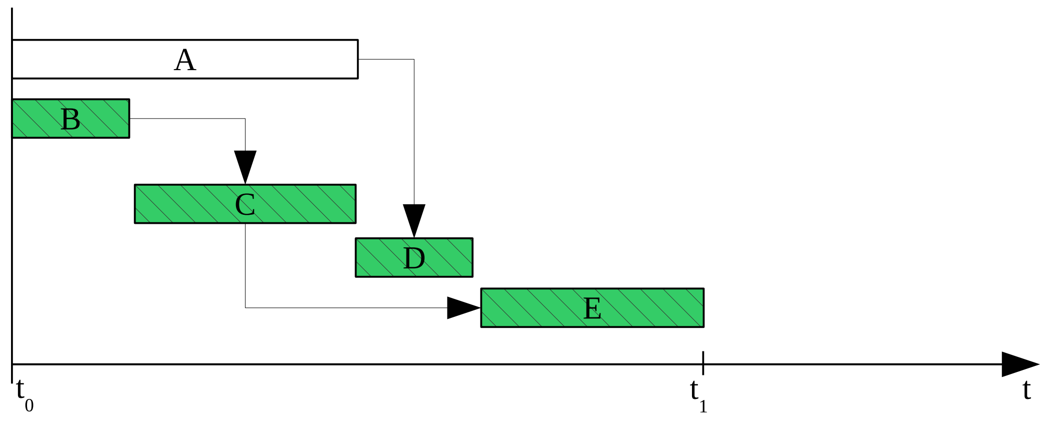


Figure 7.2: Example of a simple construction schedule. The excavation activity A requires no shared resources, while the shoring activities B through E share the same resource, of which only one instance is available. Parallel execution of any of the activities B, C, D, and E is hence prohibited.

Applying the Dori Algorithm (DA) described in Section 3.4.6 provides the information that all activities in the schedule have a float time of zero and hence are critical.

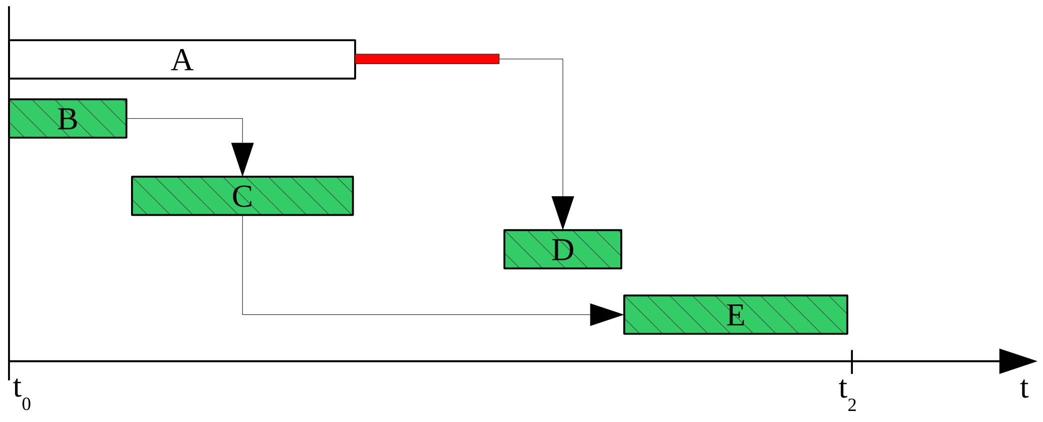


Figure 7.3: The schedule from Figure 7.2 after a delay of activity A has been observed. The delay propagated through the remaining schedule because the float time of activity A has been exceeded. Activities D and E are now scheduled to be executed later.

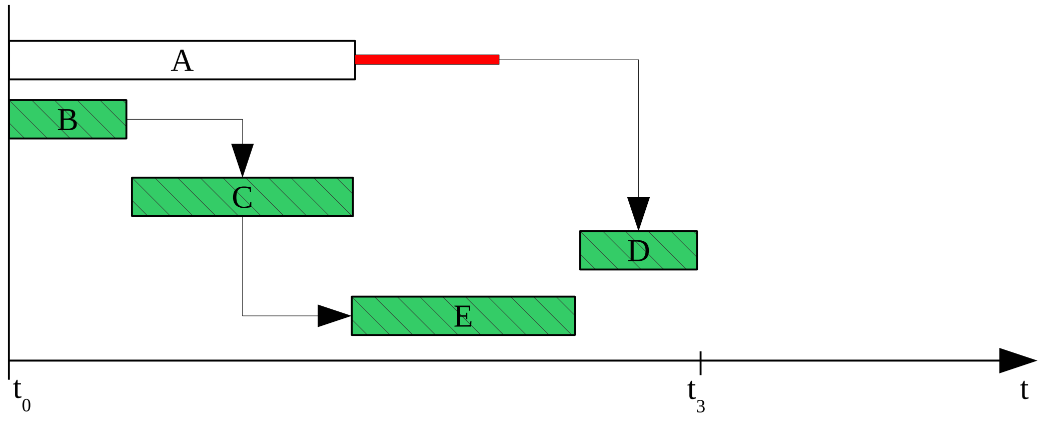


Figure 7.4: The schedule from Figure 7.2 after a delay of activity A has been observed, and since the float time was exceeded, a new schedule was generated. Now activity E is executed during the delay of activity A.

Therefore every delay in the schedule will propagate through the entire remaining schedule and increase the overall makespan by the amount of the delay. It is assumed that activity A is monitored, and prior to its finish, it is expected that it will take longer than anticipated.

This expected delay can be determined on the basis of the photogrammetry-based monitoring approach presented in Section 6.1. The duration of such an excavation activity can be a product of the performance factors of the involved excavators and dumpers. When the measured progress falls behind the expectations, the video-based monitoring approach described in Section 6.2 helps to attribute the delay to either the involved excavators or dumpers. If the decision makers react by deploying additional resources, those can be accounted for in terms of updated performance factors, when calculating the new duration of the activity. When no or insufficient counter measures are taken, the result is a delay of the activity.

The red bar after activity A in Figure 7.3 illustrates such a delay. As no re-planning has yet been performed the entire schedule would be delayed pushing activities D and E into the future and prolonging the schedule. As this is expected prior to the start of activity D it is still possible to change the schedule at this point. The presented concept advises to generate a new schedule in order to react to the exceeded float time.

Figure 7.4 illustrates a reactive schedule that has been updated after the observation of the delay of activity A. As a result the effects of the delay have been counteracted and the new schedule is exactly as long as the initial one from Figure 7.2. Given the longer duration of activity A, this new schedule is optimal.

However, in many practical cases a constant resource usage is desirable because the idling of resources creates unnecessary costs. Figure 7.5 shows the resource profile of the green resource of the initial schedule from Figure 7.2. The green resource is in constant use throughout the entire schedule.

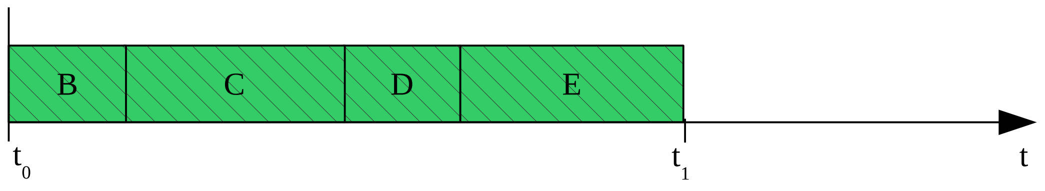


Figure 7.5: The resource profile for the green resource of the schedule in Figure 7.2.

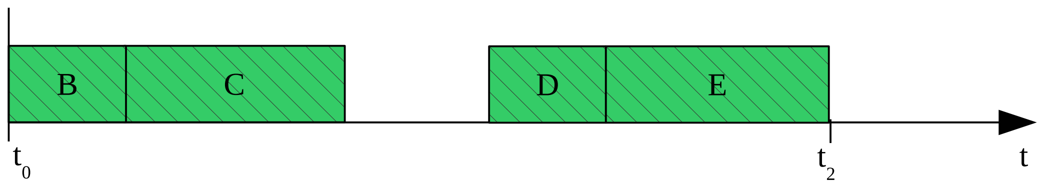


Figure 7.6: The resource profile for the green resource of the schedule in Figure 7.3.

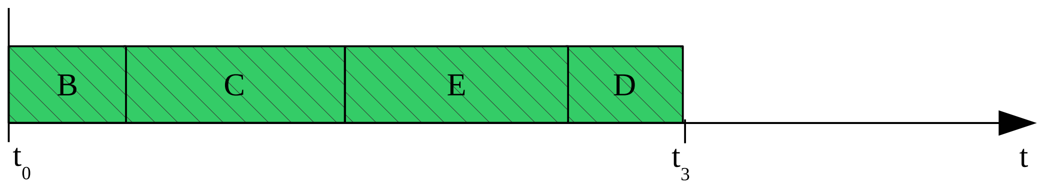


Figure 7.7: The resource profile for the green resource of the schedule in Figure 7.4.

Figure 7.6 shows the same resource profile for the delayed schedule from Figure 7.3. The delay prevents activity D from execution and therefore creates a gap in the resource profile.

The resource profile in Figure 7.7 corresponds to the reactive schedule from Figure 7.4. It is visible that the resource gap has been closed. In many cases optimizing a schedule with respect to short makespan will also yield a lean resource profile. Additionally the costs for hiring the resource could be included as an optimization objective, creating more costs for the schedule containing the gap. However at a certain duration of the gap it may become worthwhile to make the resource available to other construction sites in the meantime.

When taking robustness into account, the schedule shown in Figure 7.4 would have been a better initial schedule as activity A has a float time equal to the duration of activity E and the observed delay would not have had any effect on the schedule. This illustrates that scheduling as an multi-objective optimization problem is more practical. Taking robustness in terms of maximizing float times into account, so for instance using the sum of all float times as a second objective, would have selected the final schedule as the initial schedule, as both have equal makespans.

Another possible approach would be to take uncertainties in the activity durations or the underlying performance factors into account. If it would have been known that activity A is likely to be delayed, an approach such as Monte-Carlo simulation as described in Section 2.2.6 would have also shown the schedule in Figure 7.4 as the better initial option. However, when it is more likely that the sum of the delays of activities B and C is larger than the delay of A, the schedule in Figure 7.2 would have been an equally good choice. Since it is very hard to manually make such considerations in case of more complex schedules it is advantageous to include uncertainty into the project model beforehand, given the required data is available. When only coarse data is available, methods like VIBES (AbouRizk et al., 1991), as presented in Section 2.2.1, have been shown to be effective in estimating the required probability distributions. If detailed sample data is available from previous projects or other activities, goodness-of-fit tests, as described in Section 2.2.5 are useful for finding the most suitable probability distribution. Once a distribution is selected, its parameters can be updated based on newly acquired data in order to aid the reactive scheduling approach.

7.3 Summary

This Chapter showed how the methods presented throughout this thesis interlock into a synergic approach. It has been described how the overall process of creating a data-driven reactive scheduling model for construction projects works and an illustrative example has been presented. The example illustrated how a delay can cause an optimal schedule to become suboptimal.

Furthermore it showed how anticipation of delays can prevent the necessity of re-planning. This can be realized by taking robustness into account in terms of an multi-objective optimization approach during the generation of the initial schedule. Furthermore, if the uncertainty of activity durations or performance factors is known, performing Monte-Carlo simulation in order to evaluate the schedule during generation is a viable approach towards obtaining more robust schedules.

The following Chapter introduces three case studies which have been performed to show how the concepts described in this thesis can be put into practice.

8 Case Studies

"Experience is the only source of knowledge." - Ranganathananda (1990)

This Chapter presents experimental results obtained through case studies that have been performed on real construction sites. In order to illustrate the feasibility of the approaches presented in Sections 6.1.2 and 6.2, three case studies have been performed on different large real-life construction sites. This Chapter describes each of the cases and discusses the data that was collected and evaluated.

8.1 Office Building

In 2013 a large office building was built in the Erika-Mann-Straße in Munich. For this site a large pit had to be excavated. The progress of the excavation was monitored for two weeks in order to evaluate the capabilities of the photogrammetry-based volume calculation approach that was presented in Section 6.1.2. Figure 8.1 shows a point cloud that was generated from the site. Figure 8.3 shows the volume that was calculated from the point clouds. The low change between days 2 and days 8 can also be verified visually from the point clouds in Figure 8.2. Furthermore days 6 and 7 were a weekend. The Figure shows all nine point clouds that were generated during the course of the observation.

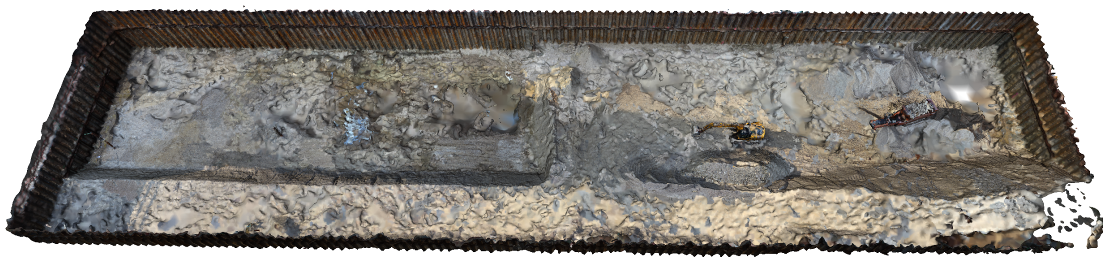


Figure 8.1: Point cloud of the office building excavation site.

Figure 8.4 shows performance factors that have been calculated based on the volume measurements. The blue bars attribute the change of each day to the day of measuring the difference. The red bars are interpolated, assuming constant change has happened since the last measurement. Figure 8.5 shows the probability density function of a beta

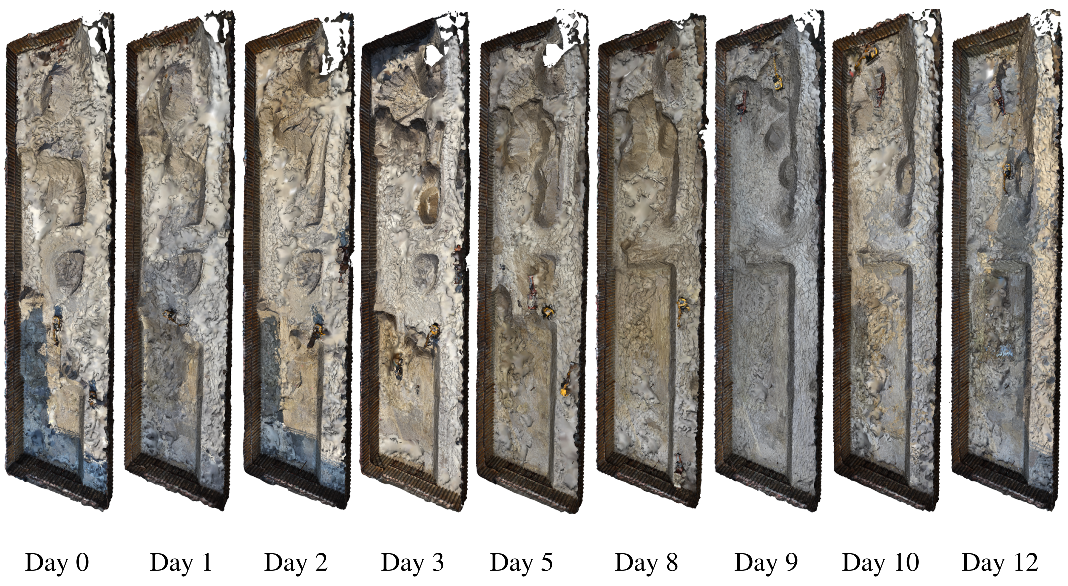


Figure 8.2: Time series of point clouds of the office building excavation site.

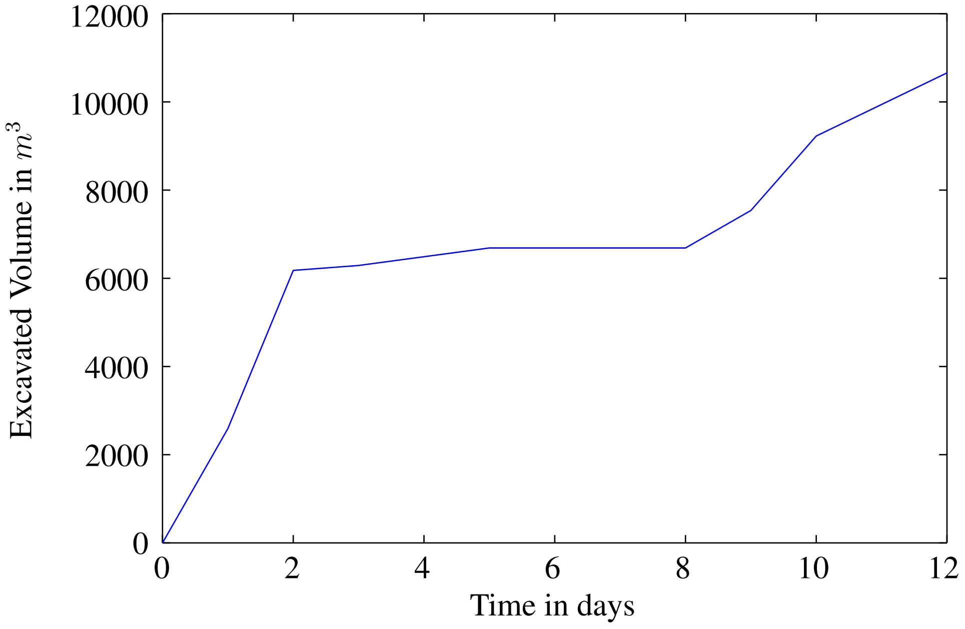


Figure 8.3: Volumes of the series of point clouds in Figure 8.2.

distribution that was created based on the sample data. This distribution could be used to make forecasts on the future performance of the excavation process. Both goodness-of-fit tests presented in Section 2.2.5 confirm that this probability distribution fits the data. The mean excavation performance was $1227.5m^3$/day. Moving the excavated soil out of the pit was particularly easy due to the ramp, that is visible in Figure 8.1 on the bottom, leaving the pit towards the right.

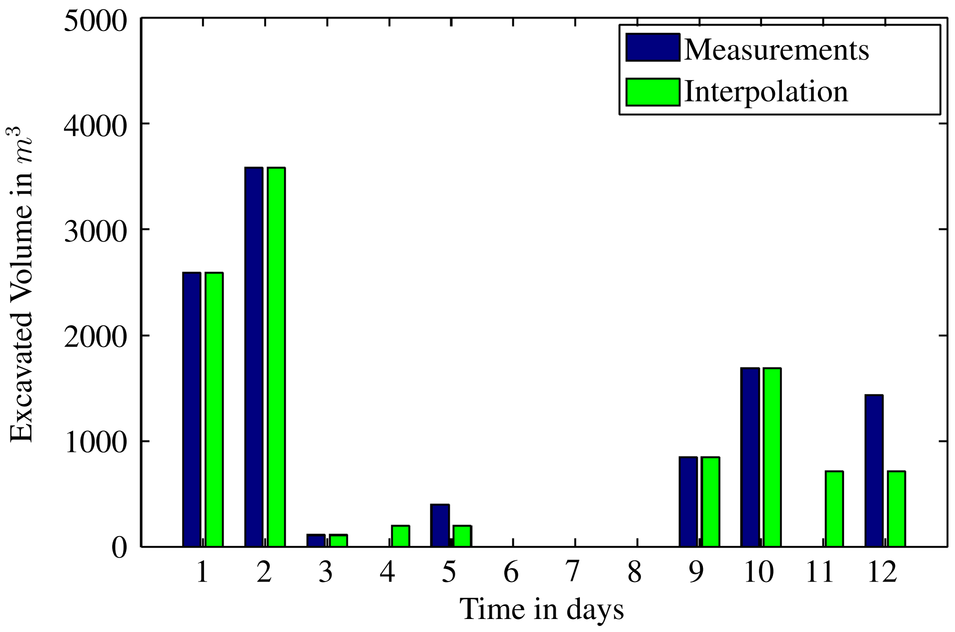


Figure 8.4: Performance factors calculated based on the volume calculations in Figure 8.3.

This case study illustrated the capabilities of the photogrammetry based progress monitoring on a rectangular construction site with a large surface area. Furthermore the calculation of performance factors was demonstrated and the uncertainty can successfully be described using a beta distribution. The resulting data can be used to maintain synchronization of a reactive simulation model that updates construction schedules based on the newly obtained information.

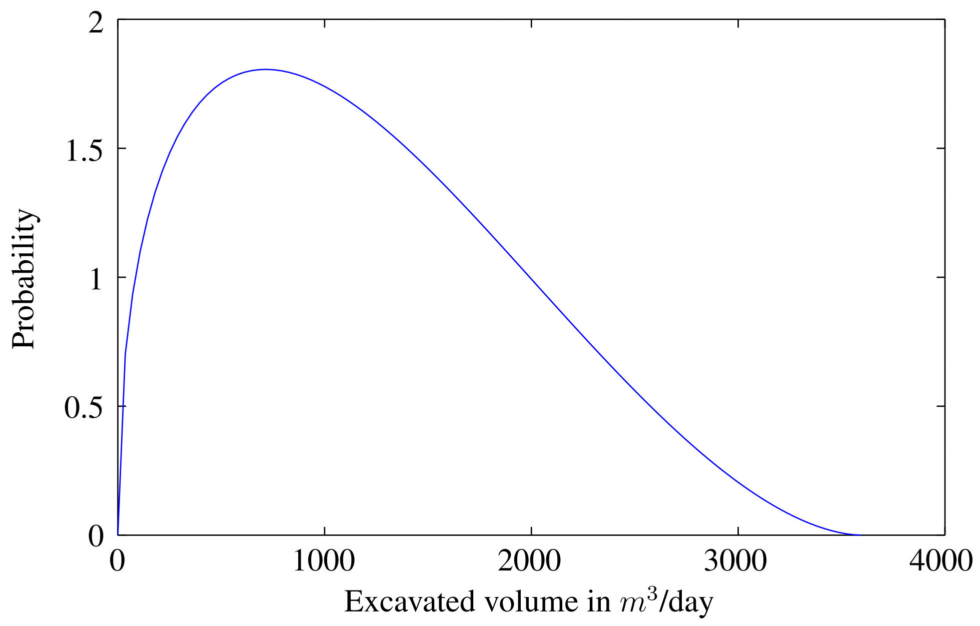


Figure 8.5: Beta distribution estimated based on the performance factors in Figure 8.4 with $\alpha = 1.4427$ and $\beta = 2.7839$. The mean performance was $1227.5m^3/$day.

8.2 Underground Parking Garage

Due to the lack of parking spaces in downtown Munich, a new underground parking garage was built. For this purpose a deep round pit had to be excavated. During the observation period of 36 days, $12000m^3$ of soil were excavated. Figure 8.6 shows a point cloud that was created during the excavation process. It is visible that the soil was lifted out of the pit by an excavator through a step at about half the depth of the pit. This slowed down the excavation performance compared to the previous site to $429m^3$/day. An additional factor was the traffic situation in Munich made it particularly difficult for the dumpers to get to the dump site and back. This fact could be noticed by the extended absence of dumb trucks on the site. The excavation progress over time is plotted in Figure 8.7. The derived performance factors are illustrated in Figure 8.8. The blue bars show the observed performance attributed to the day of recording the point clouds, while the red bars spread the volume delta over the workdays since the last recording. The graph in Figure 8.9 shows the probability density function of the beta distribution estimated from the performance data. This distribution can be used to make predictions on the excavation performance since both goodness-of-fit tests presented in Section 2.2.5 confirm that this probability distribution fits the data.

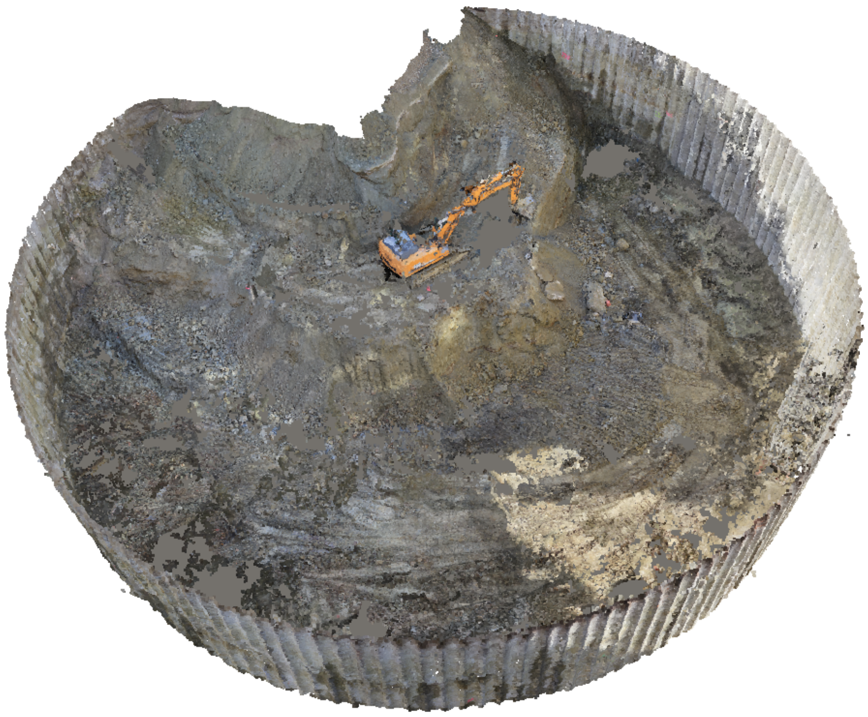


Figure 8.6: Point cloud of the underground parking garage excavation site. Lifting soil out of the pit slowed down the excavation progress.

Additionally a video camera was placed during day 26 of the observation period. 4 hours of video were recorded. A point cloud has been created prior to the recording and a second one after the recording. The event processor, presented in Ogunmakin et al.

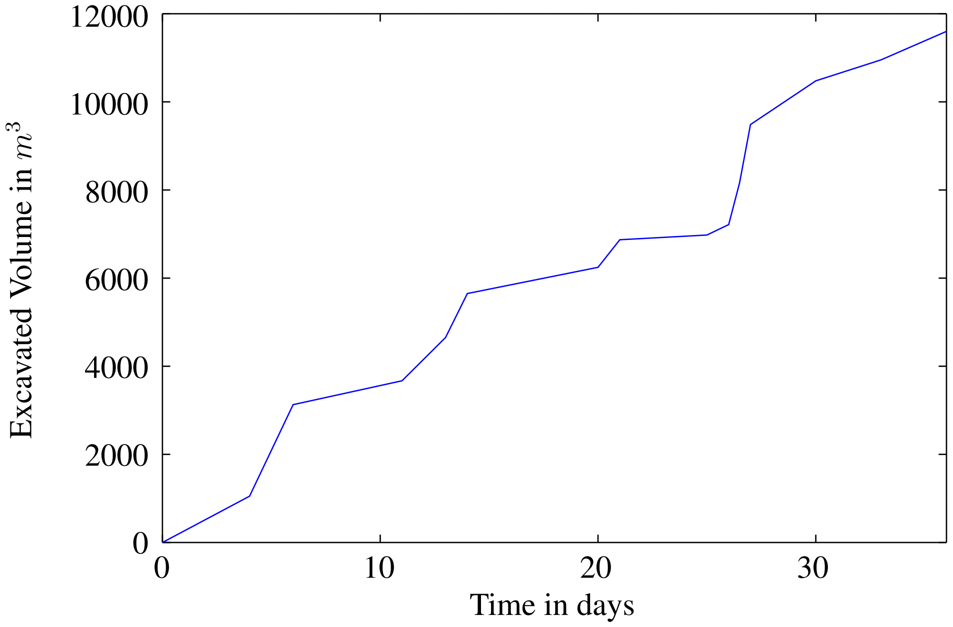


Figure 8.7: Volume plot of the underground parking garage excavation site.

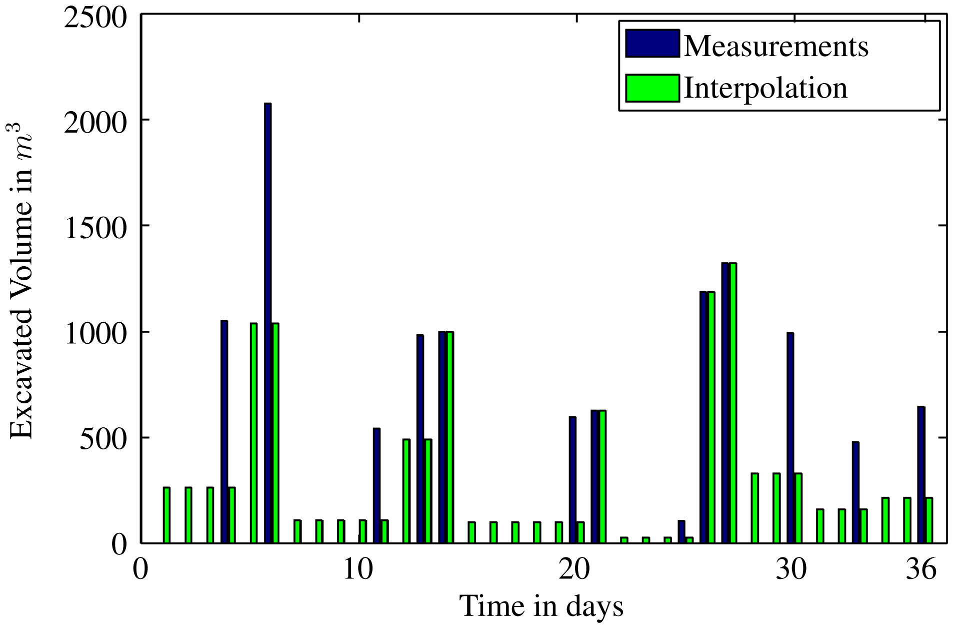


Figure 8.8: Performance factors calculated based on the volume calculations in Figure 8.7.

(2013), combined the tracking and activity estimation results to generate the statistics illustrated in Figure 8.10. The statistics were generated by counting excavator bucket loads and dump trucks entering and leaving the site. In order to measure the performance

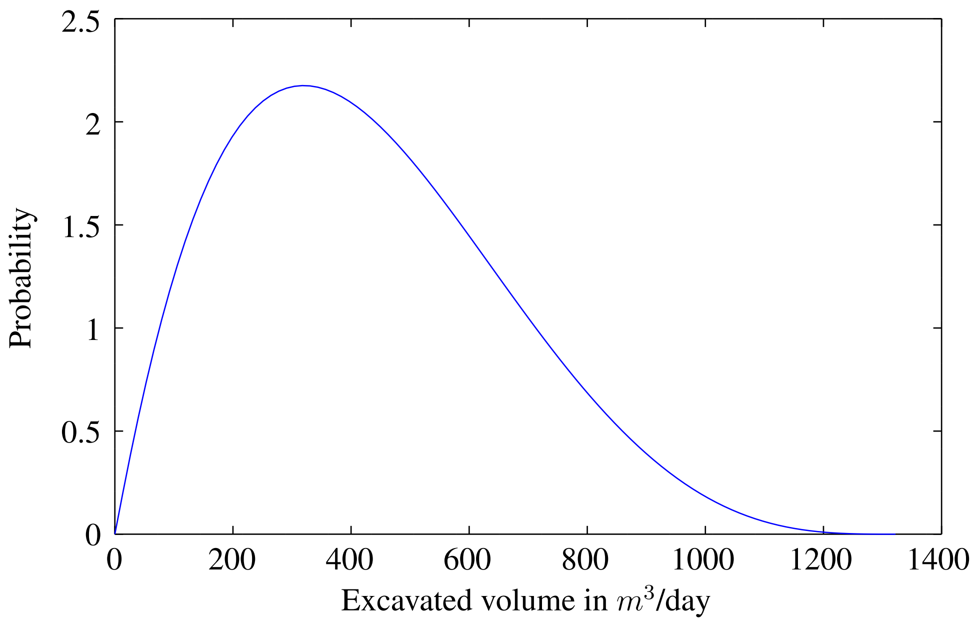
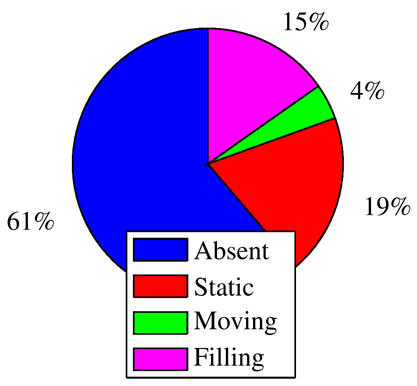


Figure 8.9: Beta distribution estimated based on the performance factors in Figure 8.8 with $\alpha = 2.0244$ and $\beta = 4.2109$. The mean performance was $429.21m^3$/day.

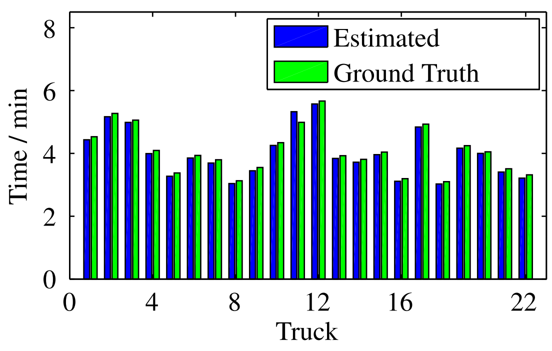
of the algorithm the ground truth was manually counted in the video. A total of n_{trucks} = 22 were detected in the processed video, which exactly matched the ground truth. A total of $n_{buckets} = 171$ bucket loads were detected compared to the ground truth of $n_{buckets} = 177$ bucket loads, implying an error of 3.4%. The total volume of the excavator's bucket was $2.5m^3$. Using the ground truth $n_{buckets,gt} = 177$, results in a performance factor of $\tau_{v,gt} = \frac{2.5m^3 \cdot 177}{4h} = 110\frac{m^3}{h}$. Using the estimated $n_{buckets} = 171$, results in a performance factor of $\tau_v = \frac{2.5m^3 \cdot 171}{4h} = 107/fracm^3h$. Through the total excavated volume, calculated on the basis of the point clouds, the corrected volume per bucket can be calculated. Using the ground truth bucket count, this results in $v_{bucket_cor,gt} = \frac{418m^3}{177} = 2.36m^3$, while using the measured data it results in $v_{bucket_cor,gt} = \frac{418m^3}{171} = 2.44m^3$, also yielding an error of 3.4%. The corrected bucket volume accounts for the swell factor of the soil as well as for the fact that the buckets aren't always completely filled. From the pie chart in Figure 8.10a it is visible that the majority of the time, the excavators were waiting for the dumpers[9].

The presented results show that the photogrammetry approach can be used to monitor the excavation process for long periods of time. Furthermore it shows that the additional information obtained through the video analysis approach presented by Ogunmakin et al. (2013) can be used to obtain deeper insight into the progress and to detect underlying problems causing delays. The precise metrics obtained through this approach can be used as feedback for a reactive scheduling model either updating performance factors or estimating updated activity durations and their respective probability distributions.

[9]This synergetic approach was developed in close cooperation with Gbolabo Ogunmakin, Patricio A. Vela, André Borrmann, and Jochen Teizer.



(a) Pie chart with aggregate dump truck states



(b) Loading time per truck

Figure 8.10: Aggregate dump truck statistics for underground parking garage. The left part shows a pie chart with the ratios of time spent in the different states. The right part shows the loading time for the individual trucks (Bügler et al., 2016).

8.3 Hospital Building

A third construction site was subject to the research presented in this paper. In 2014 the construction of a large new hospital building with multiple sub-basements was started. For the purpose of monitoring the excavation progress of the project, six cameras have been permanently placed around the construction site. With the use of these cameras and some additional photos taken by on-site personnel point clouds were generated at regular intervals for a period of 89 days in which more than $20,000m^3$ of soil were excavated. One of the point clouds is visualized in Figure 8.11. The calculated excavation progress is plotted in Figure 8.12. It is visible, that there were two main excavation phases during which most of the progres was made. During the flat saddle point around day 20 the bored and sheet pile walls securing the excavation area were produced and anchored.

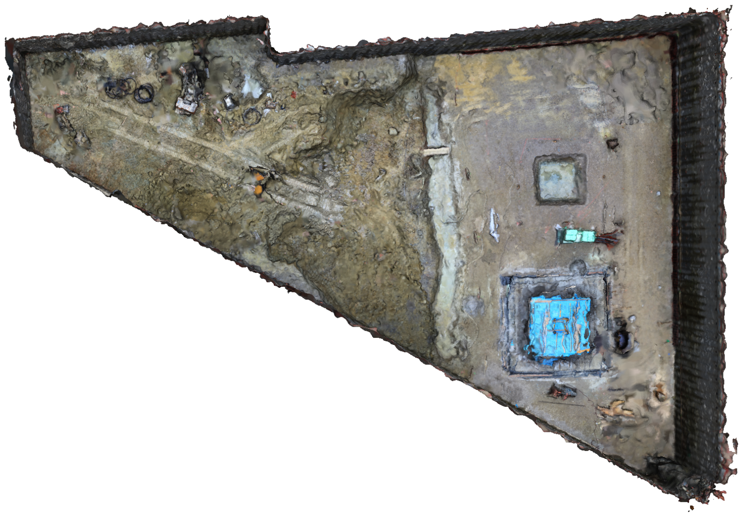


Figure 8.11: Point cloud of the hospital excavation site.

Performance factors for the entire period are visualized in Figure 8.13 and the resulting probability distribution is plotted in Figure 8.14, which was confirmed to fit the data using the goodness-of-fit test presented in Section 2.2.5. Due to the long recording period in this case study, the data contains many days on which no excavation processes were active. Therefore the calculated performance factors do not reflect the actual performance of the excavation processes in this case. Detailed data on the times when excavation was idling was not recorded, and therefore exact performance factors could not be calculated a posteriori. Based on the areas with high activity it can be estimated that the first excavation period between days 4 and 8 had a performance of $1498\frac{m^3}{d}$, while the second excavation period between days 35 and 40 had a performance of $808\frac{m^3}{d}$. For better calculation of the performance factors, machine data of the excavators could be used in order to measure the amount of time they were running. Alternatively a video monitoring approach such as the one presented in Section 6.2 can be used. Nevertheless this case study showed that photogrammetry can be used to monitor the progress of large excavation projects over long periods of time. Given additional machine data or video recordings, precise calculation of performance factors would be possible.

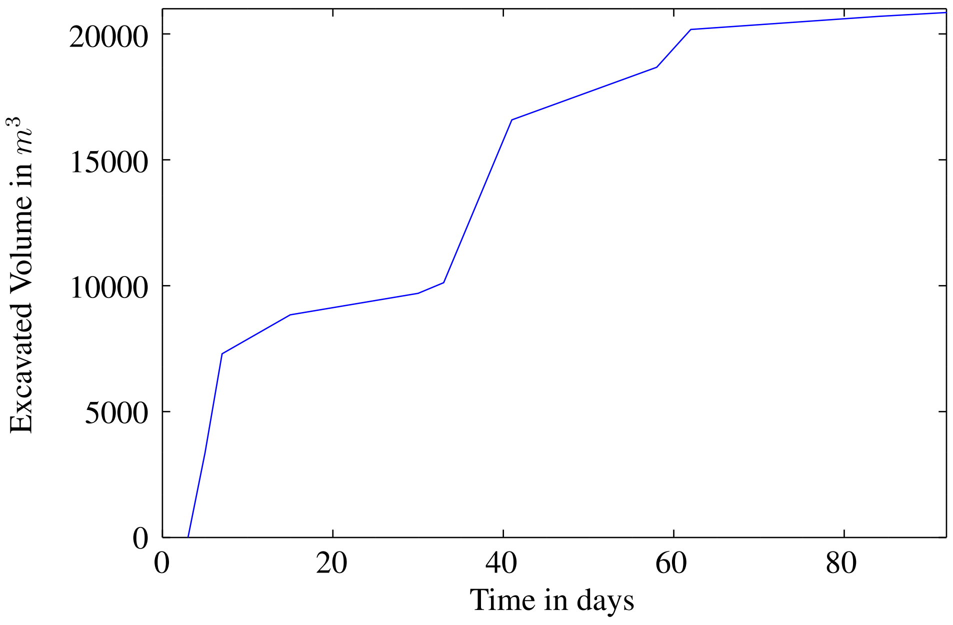


Figure 8.12: Excavation progress of the hospital excavation site

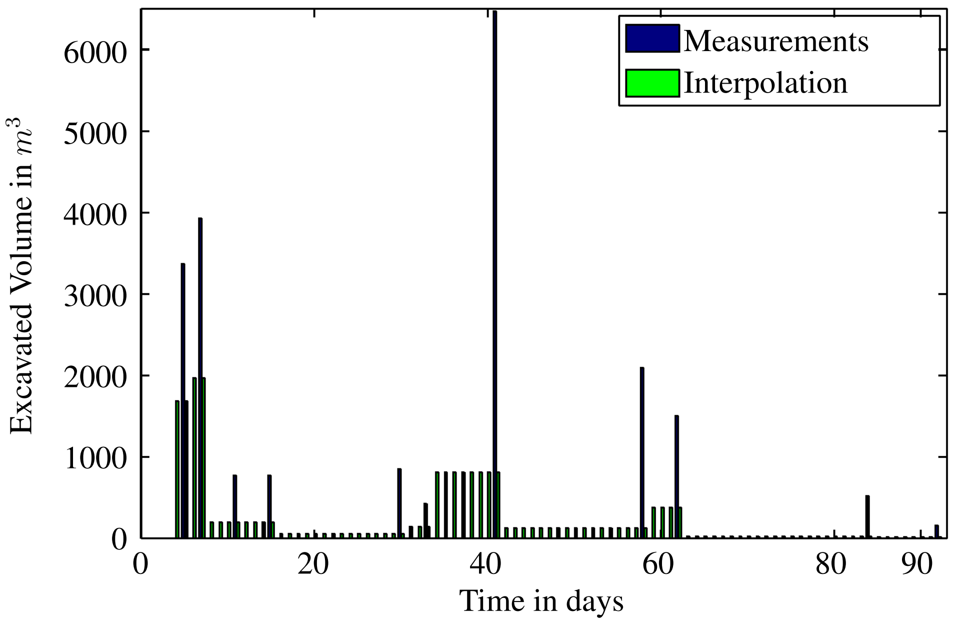


Figure 8.13: Performance factors calculated based on the volume calculations in Figure 8.12.

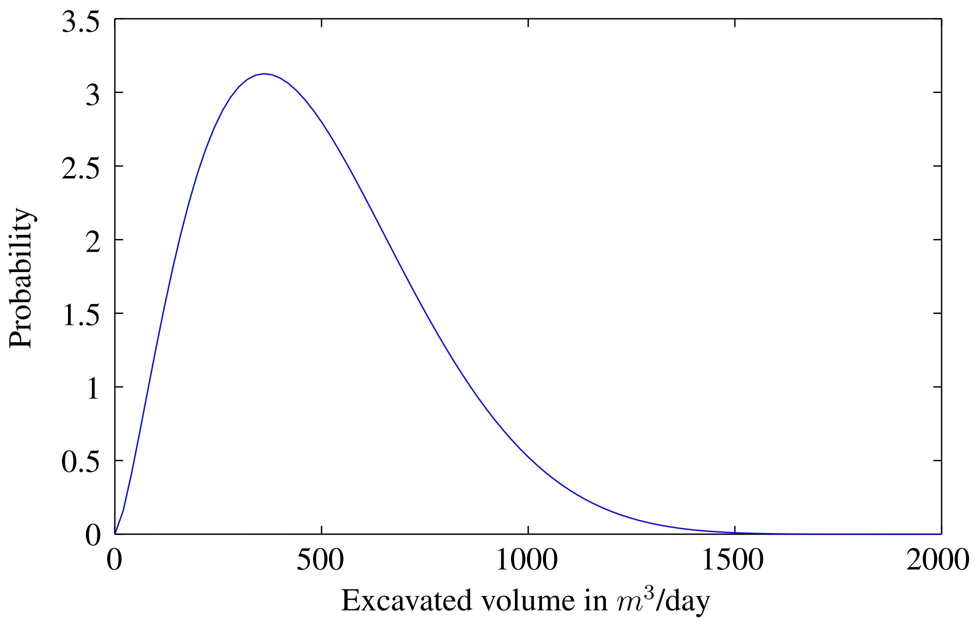


Figure 8.14: Beta distribution estimated based on the performance factors in Figure 8.13 with $\alpha = 2.4661$ and $\beta = 7.6387$. The mean performance was $488.10 m^3$/day.

8.4 Summary

The results illustrate that the photogrammetry approach to measure the excavated volume of an excavation pit can be used to monitor an excavation site for long periods of time. Furthermore performance measures can be calculated and used to make predictions on the future progress.

The video monitoring approach presented in Section 6.2 was successfully tested and allowed the exact attribution of delays to underlying circumstances. This information allows the decision makers to react appropriately to unexpected events. In combination with the calculated performance factors and their respective probability distributions it is possible to make accurate predictions for the future. Additionally, the data can be used to estimate the performance of similar processes on other construction sites.

While the case studies were mostly performed without major problems, some issues arose during their execution. Bad weather conditions, such as heavy rain, causing large pools of water, forming large reflective surfaces caused problems when capturing point clouds using photogrammetry. Since this did not occur during long periods of time and only single sessions didn't return a usable point cloud, the problems were practically neglectable.

In case of the hospital construction site there were long periods of time where no excavation took place. Those periods need to be marked manually not to be taken into account when performance factors are updated. Commonly this is happening during the anchoring of bored and sheet pile walls.

9 Summary and Outlook

While unexpected delays in large construction projects commonly render plans invalid and require replanning, taking new circumstances into account, scheduling in construction industry is still performed in a manual and error-prone way. According to Koehn et al. (1978) unexpected delays comprise up to 30 percent of the overall project costs.

This thesis presented a concept for the reactive scheduling of large shoring and excavation projects including new scheduling methods and on-site data acquisition. Additionally methods for modeling, simulation, and optimization of construction projects are introduced. Furthermore a representation of uncertainty is included and methods for including those methods into the concept are presented.

The main contributions of this thesis are the new pseudo-serial schedule generation scheme (Section 4.1), the iterative limited-depth tree search algorithm (Section 4.3) for resource constrained project scheduling problems (RCPSPs), the application of the water-wave optimization algorithm by Zheng (2015) to RCPSPs (Section 4.2), the uncertainty extension of the project scheduling library PSPLIB (Section 4.5), as well as the automatic progress tracking of excavation processes using point clouds (Section 6.1) and its combination with video-based monitoring (Section 6.2). A number of theoretical and practical case studies have been performed using the project scheduling library PSPLIB (Section 4.4) and its extenstion, the UPSPLIB (Section 4.5), as well as three different real-life construction projects which have been accompanied in the course of this thesis (Chapter 8).

Chapter 2 introduced the state-of-the-art in modeling and simulation of construction operations. A main measure to estimate how long a machine or worker takes to execute a certain activity are performance factors quantifying the amount of work done in a certain time (Section 2.1). Those performance factors, and therefore the durations of activities, are subject to uncertainties which can be modeled using different probability distributions (Section 2.2). A measure on how good a distribution fits a given set of data can be acquired using goodness-of-fit tests (Section 2.2.5). Furthermore, Monte-Carlo-Simulation can be used to estimate the consequences of uncertainties in a larger context, such as a whole construction project (Section 2.2.6). Another, if not the most important aspect of a construction project is the cost factor (Section 2.3). Costs can be generated in numerous ways and can be static or dynamic. They are created through the use of materials, resources, workers, or even fines that have to be paid due to missed milestones. A very important decision to be made when modeling construction operations is the granularity of the model (Section 2.4).

While it is possible to model the precise movement of each machine, such a model is hard to evaluate and even harder to maintain, when changes occur. Therefore it is wise to make abstractions where they do not have drastic effect on the outcome. For instance a two step method can be used. First modeling construction processes, such as the production of a bored pile wall, as a process model, based on the performance factors of the involved machines (Section 2.5). The resulting process model can then be used to obtain an expected duration for the production of that bored pile wall, which then can be included as one activity into a model of the entire project. For the purpose of modeling such processes, different models can be found in literature, such as CYCLONE (Section 2.5.4) or STROBOSCOPE (Section 2.5.5). The resulting construction project model needs to contain specific information in order to adequately represent the system to be studied (Section 2.6.2). This information can in part be automatically obtained from digital representations of the building to be constructed (Section 2.6.3). Once a model has been successfully created it can be used to perform extensive simulation studies (Section 2.6.4) as well as generate and optimize construction schedules.

Chapter 3 presents a detailed survey on the state-of-the-art in construction schedule optimization. In any optimization endeavor an objective is of utmost importance (Section 3.2). The possible objectives include costs, makespan, as well as other schedule metrics. As previously stated, uncertainty is an important aspect of any construction schedule. Therefore the robustness of a schedule is an important schedule metric (Section 3.2.1). As often multiple objectives are in the scope of the decision makers, multi-criteria optimization methods are of high interest (Section 3.2.2). The project scheduling problem (PSP) is the problem to schedule a given set of activities under precedence constraints (Section 3.3). This can be solved using the critical path method (Section 3.3.3) or the programm evaluation and review technique (Section 3.3.4). This, however, neglects the limited availability of resources, which adds considerably more complexity the problem. The resulting problem is called the resource constrained project scheduling problem (Section 3.4). In order to generate feasible schedules for a given RCPSP, two elementary schedule generation schemes, the serial and the parallel scheme, can be found in literature (Section 3.4.4). Problems can be evaluated prior to scheduling using different problem characteristics (Section 3.4.2).

Furthermore schedules can be generated by simulation (Section 3.4.5) such as the constraint-based discrete-event simulation. Using simulation-based methods allows to obtain additional measures of the schedules, such as the critical path and float times (Section 3.4.6). Generally RCPSPs can be solved in a deterministic way (Section 3.4.7), but these methods are hardly feasible on a large scale. Therefore heuristic approaches are subject to extensive research (Section 3.4.8) and numerous methods have been published in the last decades (Section 3.4.8.9). Additionally Kolisch & Sprecher (1996) published a project scheduling problem library (PSPLIB) which contains more than 2000 benchmark problems that are widely used in literature (Section 3.4.9).

Chapter 4 presents the main contributions in the field of construction scheduling of this thesis. The pseudo-serial schedule generation, as a compromise of the serial and parallel schedule generation schemes, combines the advantages of both other schemes. It can generate a superset of the set of active schedules, the set of pseudo-active schedules, while at the same time being computationally more efficient and having a more continuous parameter space, that has an increased likelihood of finding good solutions in a confined area (Section 4.1).

Built upon the new scheduling scheme, the water wave optimization algorithm by Zheng (2015) was applied to RCPSPs and has been shown to produce results outperforming many other published heuristics (Section 4.2). The performance of the algorithm was also compared with the results of using the traditional scheduling schemes. The advantage of the pseudo-serial scheduling scheme was shown to be the adjustability of the search space, as well as faster convergence of the solution.

Additionally another heuristic algorithm, the iterative swap-based limited depth tree search was presented (Section 4.3). Compared to the water wave optimization algorithm the results of this algorithm were worse, but since it starts from a given schedule by applying swaps and delays, it can be particularly useful for updating available schedules in the context of reactive re-planning.

The commonly used PSPLIB lacks the availability of problems containing uncertain activity durations. Therefore it was extended by adding beta distributed delays to each of the activities of each of the PSPLIB single mode problems. Additionally eight objective functions have been designed to allow optimization for different aspects of the uncertainties using Monte-Carlo-Simulation (Section 4.5). If newly acquired data changes the circumstances, durations, performance factors, or probability distributions, an update of the existing schedule might be required, forming the reactive component of the proposed approach (Section 4.6).

Experiments performed using the PSPLIB showed clearly that the presented algorithms compare well to other heuristic algorithms in the field of RCPSPs and even outperform many of them (Section 4.4) . Furthermore, it has been shown that the adaption of the water wave optimization to RCPSPs using the pseudo-serial schedule generation scheme is capable of creating good and robust solutions for the uncertainty extensions of the PSPLIB (Section 4.5).

Chapter 5 provides an overview of the state-of-the-art in data acquisition on construction sites. Most commonly, manual protocol-based tracking, such as noting milestones in spreadsheets, is used for progress monitoring (Section 5.1). Additionally some modern machines record data on the machine state during operation, which can be used for progress monitoring (Section 5.2). Another viable data source is laser scanning (Section 5.3). However, due to its need for trained personnel and expensive equipment, modern advances in the field of photogrammetry have gained increased interest by the community (Section 5.4). Additionally radio-based tracking, including GPS, and active, as well as passive RFID methods, are drawing attention, due to the steadily decreasing costs for the required hardware (Section 5.6). Only recently, methods of range imaging and video analysis have become computationally and technically feasible enough to have entered the realm of construction monitoring research (Section 5.7).

Chapter 6 described advances in the progress monitoring of excavation processes that were made in the course of this thesis and form additional contributions to the field of construction progress monitoring. In order to track the total volume of excavated soil a point cloud based method is presented (Section 6.1). The point clouds are obtained through photogrammetry and reduced to the area of interest by smart clustering and cleaning methods.

Finally the volume of the resulting representation of an excavation pit is calculated and can be used as a metric for the excavation progress. As this method only yields the absolute excavated volume, it does not allow to track down the reasons for observed delays. For this reason the method has been extended by a video-based tracking algorithm which was developed in synergy with Gbolabo Ogunmakin, Patricio A. Vela from Georgia Tech, André Borrmann, and Jochen Teizer (Section 6.2). This approach allows to count the number of trucks entering and leaving the site, as well as the number of buckets loaded onto the trucks. Furthermore the state of excavators and trucks can be determined. This allows for detailed analysis of the reason for delays, such as a lack of trucks or excavators.

Chapter 7 summarizes the overall concept presented in this thesis by illustrating how the individual contributions interlock into a synergic approach for data-driven reactive scheduling. The concept is furthermore illustrated by an executed example of a simple schedule. The example shows how re-planning effects the performance of a construction project and how the modeling of uncertainty and robustness can be utilized in a multi-objective optimization approach in order to reduce the probability of requiring re-planning during project execution.

Chapter 8 presents three case studies that have been performed on real construction sites in order to evaluate the performance of the contributions of this thesis. The case studies showed that the photogrammetry-based progress monitoring of excavation sites provides a powerful tool for monitoring excavation processes under different circumstances. Time series of point clouds were recorded and the progress of excavation could be tracked precisely. Additionally it was shown, that through the addition of the video-based approach by Bügler et al. (2016) the underlying reasons for delays could be determined efficiently. Through the calculation of excavated volume and the detection of machine states, periods of low performance could be attributed to the involved resources and their performance factors could be estimated accordingly.

While the presented concept is applicable in the discussed scenarios, it is subject to several limitations. The presented scheduling algorithms, given scenarios of modern large scale construction projects involving thousands of activities are still requiring large amounts of computational power and long periods of time. Finding the right level of abstraction is vital. However combining small activities to reduce the complexity of the model may create inaccurate resource allocations. On the other hand the presented methods will, even in complex cases, be able to return good solutions in acceptable time.

Construction sites that are very constricted and offer little possibility to be photographed from outside the pit are hard to be monitored using the photogrammetry approach. Furthermore the video analysis concept requires cameras to be on site. Those cameras need power and a way to transfer the recorded image data to a server for processing. If no wireless network or fast mobile network is available, storage cards would need to be retrieved from the cameras, making the approach rather impractical. The use of unmanned aerial vehicles (UAV) may be a solution to this limitation.

In practice many construction projects involve different contractors to perform different activities. Once the initial plans are in motion it may not be possible to shift the activity performed by another contractor in time as the contractor works on his own schedule on different projects. Furthermore different parts such as precast concrete parts might be produced just in time and be delivered to the site. If there is no space to intermediately store the delivered parts, shifting that activity may not be possible. In the case of excavation and shoring operations, the projects are however usually performed by a single contractor and hardly involve any large just in time delivered parts.

The presented work could be extended by a more detailed representation of the costs of construction projects. The transportation of additional resources to and from the construction sites create costs that are currently not reflected in the described representation. The presented scheduling algorithms could be extended towards to resource leveling problem (RLP) in order to compare the results with respect to robustness. Additionally the performance factors for excavation processes could be extended. Špačková & Straub (2013) presented a dynamic Bayesian network approach to model the uncertainty in tunnel excavation based on geological data, taking extraordinary events into account. Such a model could also be implemented for regular excavation processes. In the process model of excavation processes, the excavation pit could be split into multiple excavation areas, as is done in the example in Section 3.1. The capability of calculating the volumes of each of the excavation areas could be added to the point cloud-based excavation monitoring approach.

10 References

AbouRizk, S. & Mather, K. (2000), Simplifying Simulation Modeling through Integration with 3D CAD, *Journal of Construction Engineering and Management ASCE*, 126(6), 475–483.

AbouRizk, S. M. & Halpin, D. W. (1992), Statistical Properties of Construction Duration Data, *Journal of Construction Engineering and Management*, 118(3), 525–544, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1992)118:3(525).

AbouRizk, S. M.; Halpin, D. W. & Wilson, J. R. (1991), Visual Interactive Fitting of Beta Distributions, *Journal of Construction Engineering and Management*, 117(4), 589–605, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1991)117:4(589).

AbouRizk, S. M.; Halpin, D. W. & Wilson, J. R. (1994), Fitting Beta Distributions Based on Sample Data, *Journal of Construction Engineering and Management*, 120(2), 288–305, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1994)120:2(288).

Adeli, H. & Karim, A. (1997), Scheduling/Cost Optimization and Neural Dynamics Model for Construction, *Journal of Construction Engineering and Management*, 123(4), 450–458, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1997)123:4(450).

Agarwal, A.; Colak, S. & Erenguc, S. (2011), A Neurogenetic approach for the resource-constrained project scheduling problem, *Computers & Operations Research*, 38(1), 44–50, ISSN 03050548, doi:10.1016/j.cor.2010.01.007.

Ahmadabadian, A. H.; Robson, S.; Boehm, J.; Shortis, M.; Wenzel, K. & Fritsch, D. (2013), A comparison of dense matching algorithms for scaled surface reconstruction using stereo camera rigs, *ISPRS Journal of Photogrammetry and Remote Sensing*, 78, 157–167, ISSN 09242716, doi:10.1016/j.isprsjprs.2013.01.015.

Alcaraz, J. & Maroto, C. (2001), A Robust Genetic Algorithm for Resource Allocation in Project Scheduling, *Annals of Operations Research*, 102, 83–109, doi:10.1023/A:1010949931021.

Alcaraz, J. & Maroto, C. (2009), Genetic Algorithms for the Resource-Constrained Project Scheduling Problem, *BEIO, Boletín de Estadística e Investigación Operativa*, 25(1), 22–31.

Alvarez-Valdes, R.; Crespo, E.; Tamarit, J. & Villa, F. (2008), GRASP and path relinking for project scheduling under partially renewable resources, *European Journal of Operational Research*, 189(3), 1153–1170, ISSN 03772217, doi:10.1016/j.ejor.2006.06.073.

Arbuthnot, J. M. (1712), *The History of John Bull.*

Artigues, C.; Leus, R. & Talla Nobibon, F. (2013), Robust optimization for resource-constrained project scheduling with uncertain activity durations, *Flexible Services and Manufacturing Journal*, 25(1-2), 175–205, ISSN 19366582, doi:10.1007/s10696-012-9147-2.

Assaf, S. A. & Al-Hejji, S. (2006), Causes of delay in large construction projects, *International Journal of Project Management*, 24(4), 349–357, ISSN 02637863, doi:10.1016/j.ijproman.2005.11.010.

Astrom, K.; Heyden, A.; Kahl, F. & Oskarsson, M. (1999), Structure and Motion from Lines Under Affine Projections, in *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, ISBN 0769501648, pp. 285–292, doi:10.1109/ICCV.1999.791232.

Atkinson, R. (1999), Project Management: Cost, Time and Wuality, Two Best Guesses and a Phenomenon, its Time to Accept Other Success Criteria, *International Journal of Project Management*, 17(6), 337–342, ISSN 02637863, doi:10.1016/S0263-7863(98)00069-6.

Aziz, R. F.; Hafez, S. M. & Abuel-Magd, Y. R. (2014), Smart optimization for mega construction projects using artificial intelligence, *Alexandria Engineering Journal*, 53(3), 591–606, ISSN 11100168, doi:10.1016/j.aej.2014.05.003.

Baar, T.; Brucker, P. & Knust, S. (1999), Tabu Search Algorithms and Lower Bounds for the Resource-Constrained Project Scheduling Problem, in *Meta-Heuristics*, Springer, pp. 1–18, doi:10.1007/978-1-4615-5775-3{_}1.

Ballestín, F.; Barrios, A. & Valls, V. (2011), An evolutionary algorithm for the resource-constrained project scheduling problem with minimum and maximum time lags, *Journal of Scheduling*, 14(4), 391–406, ISSN 10946136, doi:10.1007/s10951-009-0125-9.

Baloi, D. & Price, A. D. (2003), Modelling global risk factors affecting construction cost performance, *International Journal of Project Management*, 21(4), 261–269, ISSN 02637863, doi:10.1016/S0263-7863(02)00017-0.

Banks, J.; Carson, J.; Nelson, B. & Nicol, D. (2001), *Discrete-Event System Simulation*, Prentice Hall, ISBN 0-13-088702-1.

Barber, C. B.; Dobkin, D. P. & Huhdanpaa, H. (1996), The Quickhull algorithm for convex hulls, *ACM Transactions On Mathematical Software*, 22(4), 469–483.

Bauer, M. (2009), *Resource-Constrained Project Scheduling - Models and Extensions*, Ph.D. thesis, University of Hamburg.

Bay, H.; Tuytelaars, T. & Van Gool, L. (2006), SURF: Speeded Up Robust Features, in *Proceedings of the 9th European Conference on Computer Vision*, pp. 404–417, doi: 10.1007/11744023{_}32.

Beis, J. & Lowe, D. (1997), Shape indexing using approximate nearest-neighbour search in high-dimensional spaces, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1000–1006, ISSN 1063-6919, doi:10.1109/ CVPR.1997.609451.

Beißert, U. (2010), *Constraint-basierte Simulation zur Terminplanung von Ausführungsprozessen Repräsentation baubetrieblichen Wissens mittels Soft Constraints*, Ph.D. thesis, Bauhaus-Universität Weimar.

Beißert, U.; König, M. & Bargstädt, H.-J. (2007), Constraint-Based Simulation of Outfitting Processes in Building Engineering, in *Proceedings of CIB-W78 24th International Conference on Information Technology in Construction*, pp. 491–498.

Beißert, U.; König, M. & Bargstädt, H.-J. (2010), Soft Constraint-based simulation of execution strategies in building engineering, *Journal of Simulation*, 4(4), 222–231, ISSN 1747-7778, doi:10.1057/jos.2010.8.

Bell, C. E. & Han, J. (1991), A New Heuristic Solution Method in Resource-Constrained Project Scheduling, *Naval Research Logistics*, 38, 315–331, ISSN 0894069X, doi:10. 1002/1520-6750(199106)38:3<315::AID-NAV3220380304>3.0.CO;2-7.

Bell, C. E. & Park, K. (1990), Solving resource-constrained project scheduling problems by A* search, *Naval Research Logistics*, 37(1), 61–84, ISSN 0894069X, doi:10.1002/ 1520-6750(199002)37:1<61::AID-NAV3220370104>3.0.CO;2-S.

Benevolenskiy, A.; Roos, K.; Katranuschkov, P. & Scherer, R. J. (2012), Construction processes configuration using process patterns, *Advanced Engineering Informatics*, 26(4), 727–736, ISSN 14740346, doi:10.1016/j.aei.2012.04.003.

Bernardini, F.; Mittleman, J.; Rushmeier, H.; Silva, C. & Taubin, G. (1999), The Ball-Pivoting Algorithm for Surface Reconstruction, *IEEE Transactions on Visualization and Computer Graphics*, 5(4), 349–359, ISSN 10772626, doi:10.1109/2945.817351.

Bershtein, L.; Knyazeva, M. & Rozenberg, I. (2015), Fuzzy resource-constrained project scheduling for GIS software development, in *Proceedings of the 16th World Congress of the International Fuzzy Systems Association (IFSA) and the 9th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT)*, pp. 1542–1548.

Bettemir, Ö. H. & Sonmez, R. (2012), Hybrid Genetic Algorithm with Simulated Annealing for Resource-Constrained Project Scheduling, *Journal of Management in Engineering*, 31(5), 1–8, doi:10.1061/(ASCE)ME.1943-5479.0000323.

Björk, B.-C. (1989), Basic structure of a proposed building product model, *Computer-Aided Design*, 21(2), 71–78, doi:10.1016/0010-4485(89)90141-3.

Björk, B.-C. (1992), A Unified Approach for Modeling Construction Information, *Building and Environment*, 27(2), 173–194.

Blazewicz, J.; Lenstra, J. & Kan, A. (1983), Scheduling subject to resource constraints: classification and complexity, *Discrete Applied Mathematics*, 5(1), 11–24.

Blum, C. & Roli, A. (2003), Metaheuristics in combinatorial optimization: overview and conceptual comparison, *ACM Computing Surveys*, 35(3), 189–213, ISSN 02545330, doi:10.1007/s10479-005-3971-7.

Boctor, F. F. (1990), Some efficient multi-heuristic procedures for resource-constrained project scheduling, *European Journal of Operational Research*, 49(1), 3–13, ISSN 03772217, doi:10.1016/0377-2217(90)90116-S.

Boctor, F. F. (1993), Heuristics for scheduling projects with resource restrictions and several resource-duration modes, *International Journal of Production Research*, 31(11), 2547–2558, ISSN 0020-7543, doi:10.1080/00207549308956882.

Boctor, F. F. (1996), Resource-constrained project scheduling by simulated annealing, *International Journal of Production Research*, 34(8), 2335–2351, ISSN 0020-7543, doi:10.1080/00207549608905028.

Bogenstätter, U. (2000), Prediction and optimization of life-cycle costs in early design, *Building Research & Information*, 28(5-6), 376–386, ISSN 0961-3218, doi:10.1080/096132100418528.

Bornaz, L. & Rinaudo, F. (2004), Terrestrial laser scanner data processing, *Proceedings of XX ISPRS Commission V Congress*, 45(B5), 514–519.

Bosché, F. (2010), Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction, *Advanced Engineering Informatics*, 24(1), 107–118, ISSN 14740346, doi:10.1016/j.aei.2009.08.006.

Bosché, F.; Ahmed, M.; Turkan, Y.; Haas, C. T. & Haas, R. (2015), The value of integrating Scan-to-BIM and Scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components, *Automation in Construction*, 49, 201–213, ISSN 09265805, doi:10.1016/j.autcon.2014.05.014.

Bosche, F. & Haas, C. (2008), Automated retrieval of 3D CAD model objects in construction range images, *Automation in Construction*, 17(4), 499–512, ISSN 09265805, doi:10.1016/j.autcon.2007.09.001.

Böttcher, J. & Drexl, A. (1999), Project scheduling under partially renewable resource constraints, *Management Science*, 45(4), 543–559.

Bouleimen, K. & Lecocq, H. (2003), A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version, *European Journal of Operational Research*, 149(2), 268–281, ISSN 03772217, doi: 10.1016/S0377-2217(02)00761-0.

Brandt, S. & Bode, M. (2014), Ablaufplanung unter Berücksichtigung von Unschärfe, in *Forum Bauinformatik 2014*.

Braun, A.; Tuttas, S.; Borrmann, A. & Stilla, U. (2015a), A concept for automated construction progress monitoring using BIM-based geometric constraints and photogrammetric point clouds, *Journal of Information Technology in Construction*, 20(8), 68–79.

Braun, A.; Tuttas, S.; Borrmann, A. & Stilla, U. (2015b), Automated progress monitoring based on photogrammetric point clouds and precedence relationship graphs, in *The 32nd International Symposium on Automation and Robotics in Construction and Mining (ISARC)*.

Brooks, G. H. & White, C. R. (1965), An algorithm for finding optimal or near optimal solutions to the production scheduling problem, *Journal of Industrial Engineering*, 16(1), 34.

Brooks, R. J. & Tobias, a. M. (1996), Choosing the best model: Level of detail, complexity, and model performance, *Mathematical and Computer Modelling*, 24(4), 1–14, ISSN 08957177, doi:10.1016/0895-7177(96)00103-3.

Brown, D. C. (1976), The bundle adjustment—progress and prospects, *Int. Archives Photogrammetry*, 21(3), 1.

Browning, T. R. & Yassine, A. a. (2010), Resource-constrained multi-project scheduling: Priority rule performance revisited, *International Journal of Production Economics*, 126(2), 212–228, ISSN 09255273, doi:10.1016/j.ijpe.2010.03.009.

Bügler, M. & Borrmann, A. (2014), Using Swap-Based Search Trees to obtain Solutions for Resource Constrained Project Scheduling Problems, in *Proceedings in Applied Mathematics and Mechanics*, doi:10.1002/pamm.201410385.

Bügler, M.; Dori, G. & Borrmann, A. (2013), Swap Based Process Schedule Optimization using Discrete-Event Simulation, in *Proceedings of the International Conference on Construction Applications of Virtual Reality*, doi:10.13140/RG.2.1.2008.8162.

Bügler, M.; Ogunmakin, G.; Vela, P. A.; Borrmann, A. & Teizer, J. (2016), Fusion of Photogrammetry and Video Analysis for Productivity Assessment of Earthwork Processes (Accepted), *Computer-Aided Civil and Infrastructure Engineering*.

Błażewicz, J.; Domschke, W. & Pesch, E. (1996), The job shop scheduling problem: Conventional and new solution techniques, *European Journal of Operational Research*, 93(1), 1–33, ISSN 03772217, doi:10.1016/0377-2217(95)00362-2.

Calhoun, K. M.; Deckro, R. F.; Moore, J. T.; Chrissis, J. W. & Van Hove, J. C. (2002), Planning and re-planning in project and production scheduling, *Omega*, 30(3), 155–170, ISSN 03050483, doi:10.1016/S0305-0483(02)00024-5.

Cameron, W. B. (1963), *Informal sociology: A casual introduction to sociological thinking*, Random House New York.

Chahrour, R. (2006), Integration von CAD und Simulation auf Basis von Produktmodellen im Erdbau.

Chahrour, R. & Franz, V. (2006), Seamless data model for a CAD-based simulation system, in *Proceedings of the Joint International Conference on Computing and Decision Making in Civil and Building Engineering*, pp. 3958–3967.

Chan, W.; Chua, D. & Kannan, G. (1996), Construction resource scheduling with genetic algorithms, *Journal of Construction Engineering and Management*, 122(2), 125–132.

Chassein, A. & Goerigk, M. (2016), A bicriteria approach to robust optimization, *Computers & Operations Research*, 66, 181–189, ISSN 03050548, doi:10.1016/j.cor.2015.08.007.

Chen, R. M.; Wu, C. L.; Wang, C. M. & Lo, S. T. (2010), Using novel particle swarm optimization scheme to solve resource-constrained scheduling problem in PSPLIB, *Expert Systems with Applications*, 37(3), 1899–1910, ISSN 09574174, doi:10.1016/j.eswa.2009.07.024.

Cheng, M.-Y. & Chen, J.-C. (2002), Integrating barcode and GIS for monitoring construction progress, *Automation in Construction*, 11(1), 23–33, ISSN 09265805, doi:10.1016/S0926-5805(01)00043-7.

Cheng, T.; Venugopal, M.; Teizer, J. & Vela, P. (2011), Performance evaluation of ultra wideband technology for construction resource location tracking in harsh environments, *Automation in Construction*, 20(8), 1173–1184, ISSN 09265805, doi:10.1016/j.autcon.2011.05.001.

Cheok, G. S. & Stone, W. C. (1999), Non-intrusive scanning technology for construction assessment, in *Proceedings of the 16th International Symposium on Automation and Robotics in Construction*, pp. 645–650.

Cheok, G. S.; Stone, W. C.; Lipman, R. R. & Witzgall, C. (2000), Ladars for construction assessment and update, *Automation in Construction*, 9(5-6), 463–477, ISSN 09265805, doi:10.1016/S0926-5805(00)00058-3.

Chtourou, H. & Haouari, M. (2008), A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling, *Computers & Industrial Engineering*, 55(1), 183–194, ISSN 03608352, doi:10.1016/j.cie.2007.11.017.

Clark, W.; Polakov, W. N. & Trabold, F. W. (1922), *The Gantt chart: A working tool of management*, The Ronald Press Company.

Cooper, D. F. (1976), Heuristics for Scheduling Resource-Constrained Projects: An Experimental Investigation, *Management Science*, 22(11), 1186–1194, ISSN 0025-1909, doi:10.1287/mnsc.22.11.1186.

Costin, A.; Pradhananga, N. & Teizer, J. (2012), Leveraging passive RFID technology for construction resource field mobility and status monitoring in a high-rise renovation project, *Automation in Construction*, 24, 1–15, ISSN 09265805, doi:10.1016/j.autcon.2012.02.015.

Daum, S. & Borrmann, A. (2014), Processing of Topological BIM Queries using Boundary Representation Based Methods, *Advanced Engineering Informatics*, 28(4), 272–286, doi:10.1016/j.aei.2014.06.001.

Davis, E. (1973), Project scheduling under resource constraints - historical review and categorization of procedures, *AIIE Transactions*, 5(4), 37–41, doi:10.1080/05695557308974916.

Davis, E. W. & Patterson, J. H. (1975), A Comparison of Heuristic and Optimum Solutions in Resource-Constrained Project Scheduling, *Management Science*, 21(8), 944–955, ISSN 0025-1909, doi:10.1287/mnsc.21.8.944.

Deb, K. & Gupta, H. (2006), Introducing robustness in multi-objective optimization, *Evolutionary computation*, 14(4), 463–494, ISSN 1063-6560, doi:10.1162/evco.2006.14.4.463.

Debels, D.; De Reyck, B.; Leus, R. & Vanhoucke, M. (2006), A hybrid scatter search/electromagnetism meta-heuristic for project scheduling, *European Journal of Operational Research*, 169(2), 638–653, ISSN 03772217, doi:10.1016/j.ejor.2004.08.020.

Debels, D. & Vanhoucke, M. (2005), A Bi-population Based Genetic Algorithm for the Resource-Constrained Project Scheduling Problem, in *Artificial Evolution*, Springer Berlin Heidelberg, pp. 378–387, doi:10.1007/11740698{_}23.

Demeulemeester, E. L. & Herroelen, W. S. (1997), New Benchmark Results for the Resource-Constrained Project Scheduling Problem, *Management Science*, 43(11), 1485–1492, ISSN 0025-1909, doi:10.1287/mnsc.43.11.1485.

Dore, C. & Murphy, M. (2014), Semi-automatic generation of as-built BIM façade geometry from laser and image data, *Journal of Information Technology in Construction*, 19(January), 20–46, ISSN 14036835.

Dori, G. (2016), *Simulation-based methods for float time determination and schedule optimization for construction projects*, Ph.D. thesis, Technische Universität München.

Dori, G.; Borrmann, A.; Szczesny, K.; Hamm, M. & König, M. (2012), Combining forward and backward process simulation for generating and analysing construction schedules, in *Proceedings of the 14th Int. Conf. on Computing in Civil and Building Engineering*, Moscow, Russia.

Dorigo, M.; Maniezzo, V. & Colorni, A. (1996), Ant system: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(1), 29–41, ISSN 10834419, doi:10.1109/3477.484436.

Dossick, C. S. & Neff, G. (2010), Organizational Divisions in BIM-Enabled Commercial Construction, *Journal of Construction Engineering and Management*, 136, 459–467, ISSN 0733-9364, doi:10.1061/(ASCE)CO.1943-7862.0000109.

Drexl, a. (1991), Scheduling of Project Networks by Job Assignment, *Management Science*, 37(12), 1590–1602, ISSN 0025-1909, doi:10.1287/mnsc.37.12.1590.

Dzeng, R.-J. & Tommelein, I. D. (2004), Product modeling to support case-based construction planning and scheduling, *Automation in Construction*, 13(3), 341–360, ISSN 09265805, doi:10.1016/j.autcon.2003.10.002.

Echeverry, D. (1996), Adaptation of barcode technology for construction project control, in *Proceedings of the ASCE Conference on Computing in Civil Engineering*, pp. 1034–1040.

El-Omari, S. & Moselhi, O. (2011), Integrating automated data acquisition technologies for progress reporting of construction projects, *Automation in Construction*, 20(6), 699–705, ISSN 09265805, doi:10.1016/j.autcon.2010.12.001.

Elsayed, E. A. (1982), Algorithms for project scheduling with resource constraints, *International Journal of Production Research*, 20(1), 95–103, ISSN 0020-7543, doi: 10.1080/00207548208947751.

Feng, C.-W.; Liu, L. & Burns, S. a. (1997), Using Genetic Algorithms to Solve Construction Time-Cost Trade-Off Problems, *Journal of Computing in Civil Engineering*, 11(3), 184–189, ISSN 0887-3801, doi:10.1061/(ASCE)0887-3801(1997)11:3(184).

Fente, J.; Knutson, K. & Schexnayder, C. (1999), Defining a Beta distribution function for construction simulation, in *Proc. of the 1999 Winter Simulation Conference*, ISBN 0-7803-5780-9, ISSN 02750708, pp. 1010–1015, doi:10.1109/WSC.1999.816813.

Fischer, M. & Aalami, F. (1996), Scheduling with Computer-Interpretable Construction Method Models, *Construction Engineering and Management*, 122(4), 337–347, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1996)122:4(337).

Fishman, G. S. & Kiviat, P. J. (1967), The statistics of discrete-event simulation, *Simulation*, 10(4), 185–195.

Flachs, G. M.; Thompson, W. E.; Black, R. J.; Taylor, J. M.; Cannon, W.; Rogers, R. & Others (1977), An automatic video tracking system, in *Proceedings of the National Aerospace and Electronics Conference*, pp. 361–368.

Flachs, G. M.; Thompson, W. E.; Gilbert, A. L. & Others (1976), A real-time Structural Tracking Algorithm, in *Proceedings of the National Aerospace and Electronics Conference*, pp. 161–168.

Fleszar, K. & Hindi, K. S. (2004), Solving the resource-constrained project scheduling problem by a variable neighbourhood search, *European Journal of Operational Research*, 155(2), 402–413, ISSN 03772217, doi:10.1016/S0377-2217(02)00884-6.

Flood, M. (2001), LiDAR activities and research priorities in the commercial sector, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIV(Part 3), 22–24, ISSN 1098-6596, doi:10.1017/CBO9781107415324.004.

Fu, N.; Lau, H. C.; Varakantham, P. & Xiao, F. (2012), Robust local search for solving RCPSP/max with durational uncertainty, *Journal of Artificial Intelligence Research*, 43, 43–86, ISSN 10769757, doi:10.1613/jair.3424.

Furukawa, Y. & Ponce, J. (2010), Accurate, dense, and robust multiview stereopsis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 1362–76, ISSN 1939-3539, doi:10.1109/TPAMI.2009.161.

Gantt, H. L. (1903), A Graphical Daily Balance in Manufacture, *ASME Transactions*, 24(1), 1322–1336.

Ghoddousi, P.; Eshtehardian, E.; Jooybanpour, S.; Javanmardi, A. & Ghoddousia, P. (2013), Multi-mode resource-constrained discrete time–cost-resource optimization in project scheduling using non-dominated sorting genetic algorithm, *Automation in Construction*, 30, 216–227, ISSN 09265805, doi:10.1016/j.autcon.2012.11.014.

Gilbert, A. L.; Giles, M. K.; Flachs, G. M.; Rogers, R. B. & U, Y. E. E. H. (1980), A Real-Time Video Tracking System, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(1), 47–56, ISSN 0162-8828, doi:10.1109/TPAMI.1980.4766969.

Glover, F. (1989), Tabu Search - Part I, *ORSA Journal on Computing*, 1(3), 190–206, ISSN 0899-1499, doi:10.1287/ijoc.1.3.190.

Glover, F. (1990a), Tabu Search - Part II, *ORSA Journal on Computing*, 2(1), 4–32.

Glover, F. (1990b), Tabu search: A tutorial, *Interfaces*, 20(4), 74–94.

Golenko-Ginzburg, D. & Gonik, A. (1997), Stochastic network project scheduling with non-consumable limited resources, *International Journal of Production Economics*, 48(1), 29–37.

Gonçalves, J.; Mendes, J. & Resende, M. (2008), A genetic algorithm for the resource constrained multi-project scheduling problem, *European Journal of Operational Research*, 189(3), 1171–1190, ISSN 03772217, doi:10.1016/j.ejor.2006.06.074.

Goncharov, E. (2011), A greedy heuristic approach for the Resource-Constrained Project Scheduling Problem, *Studia Informatica Universalis*, 9(3), 79–90.

Gong, J. & Caldas, C. H. (2010), Computer Vision-Based Video Interpretation Model for Automated Productivity Analysis of Construction Operations, *Journal of Computing in Civil Engineering*, 24(3), 252–264, ISSN 0887-3801, doi:10.1061/(ASCE)CP.1943-5487.0000027.

Gonsalves, R. & Teizer, J. (2009), Human motion analysis using 3D range imaging technology, *Proceedings of the 26th International Symposium on Automation and Robotics in Construction*, 76–85.

Gordon, G. (1961), A general purpose systems simulation program, in *Proc. of the AFIPS '61 - Eastern Joint Computer Conference: Computers - Key to Total Systems Control*, pp. 87–104.

Grima, M. a.; Bruines, P. a. & Verhoef, P. N. W. (2000), Modeling tunnel boring machine performance by neuro-fuzzy methods, *Tunnelling and Underground Space Technology*, 15(3), 259–269, ISSN 08867798, doi:10.1016/S0886-7798(00)00055-9.

Halpin, D. & Martinez, L. (1999), Real world applications of construction process simulation, in *Proceedings of the 31st Winter Simulation Conference*.

Halpin, D. W. (1977), CYCLONE: Method for modeling of job site processes, *Journal of the Construction Division*, 3(103), 489–499.

Halpin, D. W. & Riggs, L. S. (1992), *Planning and Analysis of Construction Operations*, John Wiley & Sons, ISBN 047155510X.

Halpin, D. W. & Senior, B. A. (2010), *Construction management*, John Wiley & Sons, ISBN 0471787477.

Hartley, R. & Zisserman, A. (2002), *Multiple view geometry in computer vision*, Cambridge University Press, ISBN 0521540518.

Hartmann, S. (1998), A Competitive Genetic Algorithm for Resource-Constrained Project Scheduling, *Naval Research Logistics (NRL)*, 45, 733–750, ISSN 1520-6750, doi:10.1002/(SICI)1520-6750(199810)45:7<733::AID-NAV5>3.0.CO;2-C.

Hartmann, S. & Kolisch, R. (2000), Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem, *European Journal of Operational Research*, 127(2), 394–407, ISSN 03772217, doi:10.1016/S0377-2217(99)00485-3.

Hashash, Y. M. A.; Oliveira Filho, J. N.; Su, Y. Y. & Liu, L. Y. (2005), 3D laser Scanning for Tracking supported Excavation Construction, *Geo-Frontiers*, 24–26, doi:10.1061/40785(164)2.

Hendrickson, C.; Zozaya-Gorostiza, C.; Rehak, D.; Baracco-Miller, E. & Lim, P. (1987), An expert system for construction planning, *Journal of Computing in Civil Engineering*, 1(4), 253–269, ISSN 08873801, doi:10.1061/(ASCE)0887-3801(1987)1:4(253).

Herroelen, W. & Leus, R. (2004), Robust and reactive project scheduling: a review and classification of procedures, *International Journal of Production Research*, 42(8), 1599–1620.

Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, The University of Michigan Press.

Hoos, H. H. & Stützle, T. (2004), *Stochastic local search: Foundations & applications*, Elsevier.

Hoppe, H. (1999), New quadric metric for simplifying meshes with appearance attributes, *Proceedings Visualization '99 (Cat. No.99CB37067)*, ISSN 1, doi:10.1109/VISUAL.1999.809869.

Horenburg, T.; Wimmer, J. & Günthner, W. A. (2012), Resource Allocation in Construction Scheduling based on Multi-Agent Negotiation, in *Proceedings of the 14th Int. Conference on Computing in Civil & Building Engineering*.

Huang, R.-y. & Hsieh, B.-c. (2012), System Modeling of Object-Oriented Construction Process Simulations, in *Proceedings of the 29th ISARC*, pp. 1–5.

Huhnt, W. & Enge, F. (2006), Can algorithms support the specification of construction schedules, *Journal of Information Technology in Construction*, 11, 547–564, ISSN 14006529.

Jackson, J. (1957), Simulation research on job shop production, *Naval Research Logistics Quarterly*, 4(4), 287–295.

Jaselskis, E.; Anderson, M. R.; Jahren, C. T.; Rodriguez, Y. & Njos, S. (1995), Radio-frequency identification applications in construction industry, *Journal of Construction Engineering and Management*, 121(2), 189– 196.

Jaśkowski, P. & Sobotka, A. (2006), Scheduling construction projects using evolutionary algorithm, *Journal of Construction Engineering and Management*, 132(August), 861–870.

Jedrzejowicz, P. & Ratajczak-Ropel, E. (2014), Reinforcement Learning strategies for A-Team solving the Resource-Constrained Project Scheduling Problem, *Neurocomputing*, 146, 301–307, ISSN 09252312, doi:10.1016/j.neucom.2014.05.070.

Józefowska, J.; Mika, M. & Różycki, R. (2001), Simulated annealing for multi-mode resource-constrained project scheduling, *Annals of Operations Research*, 102(1-4), 137–155.

Kagioglou, M.; Cooper, R.; Aouad, G. & Sexton, M. (2000), Rethinking Construction: The Generic Design and Construction Process Protocol, *Engineering Construction and Architectural Management*, 7(2), 141–153, ISSN 0969-9988, doi:10.1108/eb021139.

Kahlmann, T.; Remondino, F. & Guillaume, S. (2007), Range Imaging Technology: New Developments and applications for people identification and tracking, in *Proceedings of the SPIE - The International Society for optical Engineering*, doi:10.1117/12.702512.

Kargul, A.; Bügler, M.; Borrmann, A. & Günthner, W. A. (2015), Web based field data analysis and data-driven simulation application for construction performance prediction, *Journal of Information Technology in Construction*, 20, 479–494.

Kazhdan, M.; Bolitho, M. & Hoppe, H. (2006), Poisson surface reconstruction, in *Proceedings of the fourth Eurographics symposium on Geometry processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SGP '06, ISBN 3-905673-36-3, pp. 61–70.

Kelley, J. & Walker, M. (1961), Critical-Path Planning and Scheduling: Mathematical Basis, *Operations Research*, 9(3), 296–320.

Kern, F. (2002), Precise determination of volume with terestical 3D-laserscanner, in *Geodesy for Geotechnical and Structural Engineering II*, pp. 531–534.

Kim, K.; Yun, Y.; Yoon, J.; Gen, M. & Yamazaki, G. (2005), Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling, *Computers in Industry*, 56(2), 143–160, ISSN 01663615, doi:10.1016/j.compind.2004.06.006.

Kiran, A. S. & Smith, M. L. (1984), Simulation studies in job shop scheduling - A Survey, *Computers & Industrial Engineering*, 8(2), 87–93, ISSN 03608352, doi: 10.1016/0360-8352(84)90002-0.

Kirkpatrick, S.; Gelatt, C. D. & Vecchi, M. P. (1983), Optimization by Simulated Annealing, *Science*, 220(4598), 671–680, ISSN 00368075, doi:10.1126/science.220.4598.671.

Kiziltas, S.; Burcu, A.; Ergen, E. & Pingbo, T. (2008), Technological assessment and process implications of field data capture technologies for construction and facility/infrastructure management, *ITcon*, 13, 134–154.

Klaubert, C. & Günthner, W. A. (2011), *Entwicklung eines RFID-basierten Informations- und Kommunikationssystems für die Baulogistik*, Ph.D. thesis.

Klaubert, C.; Schorr, M. & Günthner, W. A. (2010), Real time construction progress control using NFC, *ITG-Fachbericht - RFID Systech*.

Klein, R. (2000), Bidirectional planning: Improving priority rule-based heuristics for scheduling resource-constrained projects, *European Journal of Operational Research*, 127(3), 619–638, ISSN 03772217, doi:10.1016/S0377-2217(99)00347-1.

Klein, R. W. & Baris, P. M. (1991), Selecting and generating variates for modeling service times, *Computers & Industrial Engineering*, 20(1), 27–33.

Kochetov, Y. & Stolyar, A. (2003), Evolutionary local search with variable neighborhood for the resource constrained project scheduling problem, in *Proceedings of the Workshop on Computer Science and Information Technologies CSIT'2003*.

Koehn, E.; Young, R.; Kuchar, J. & Seling, F. (1978), Cost of delays in construction, *Journal of the Construction Division*, 104(3), 323–331.

Kolisch, R. (1996a), Efficient priority rules for the resource-constrained project scheduling problem, *Journal of Operations Management*, 14(3), 179–192, ISSN 02726963, doi:10.1016/0272-6963(95)00032-1.

Kolisch, R. (1996b), Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation, *European Journal of Operational Research*, 90(2), 320–333, doi:10.1016/0377-2217(95)00357-6.

Kolisch, R. & Drexl, A. (1996), Adaptive search for solving hard project scheduling problems, *Naval Research Logistics*, 43(1), 23–40, ISSN 0894069X, doi:10.1002/(SICI)1520-6750(199602)43:1<23::AID-NAV2>3.3.CO;2-4.

Kolisch, R. & Hartmann, S. (1999), Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis, in Weglarz, J. (ed.), *Handbook on Recent Advances in Project Scheduling*, Springer, ISBN 978-1-4613-7529-6, pp. 147–178, doi:10.1007/978-1-4615-5533-9{_}7.

Kolisch, R. & Hartmann, S. (2006), Experimental investigation of heuristics for resource-constrained project scheduling: An update, *European Journal of Operational Research*, 174(1), 23–37, ISSN 03772217, doi:10.1016/j.ejor.2005.01.065.

Kolisch, R. & Sprecher, A. (1996), Project Scheduling Problem Library (PSPLIB).

Kolisch, R. & Sprecher, A. (1997), PSPLIB - A project scheduling problem library, *European Journal of Operational Research*, 96(1), 205–216, ISSN 03772217, doi:10.1016/S0377-2217(96)00170-1.

König, M. (2011a), Generation of robust construction schedules using evolution strategies, in *Proceedings of the 2011 ASCE International Conference*, 2005, pp. 1–8.

König, M. (2011b), Robust construction scheduling using discrete-event simulation, in *Proceedings of the 2011 ASCE International Workshop on Computing in Civil Engineering*, ASCE, American Society of Civil Engineers, pp. 446–453, doi:10.1061/41182(416)55.

König, M. & Beißert, U. (2009), Construction scheduling optimization by simulated annealing, in *Proceedings of ASCE Workshop on Computing in Civil Engineering, Austin, TX*, pp. 183–190.

König, M.; Beißert, U.; Steinhauer, D. & Bargstädt, H.-J. (2007), Constraint-based Simulation of Outfitting Processes in Shipbuilding and Civil Engineering, in *Proceedings of the 6th EUROSIM Congress on Modeling and Simulation*.

König, M.; Koch, C.; Habenicht, I. & Spieckermann, S. (2012), Intelligent BIM-based construction scheduling using discrete event simulation, in *Proceedings of the 2012 Winter Simulation Conference*, ISBN 9781467347808, pp. 1219–1229.

Koulinas, G.; Kotsikas, L. & Anagnostopoulos, K. (2014), A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem, *Information Sciences*, 277, 680–693, ISSN 00200255, doi:10.1016/j.ins.2014.02.155.

Kroese, D. P.; Brereton, T.; Taimre, T. & Botev, Z. I. (2014), Why the Monte Carlo method is so important today, *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6), 386–392, ISSN 19395108, doi:10.1002/wics.1314.

Kurtulus, I. & Davis, E. W. (1982), Multi-Project Scheduling: Categorization of Heuristic Rules Performance, *Management Science*, 28(2), 161–172, ISSN 00251909, doi:10.1287/mnsc.28.2.161.

Kusnoto, B. & Evans, C. A. (2002), Reliability of a 3D surface laser scanner for orthodontic applications, *American Journal of Orthodontics and Dentofacial Orthopedics*, 122(4), 342–348, ISSN 08895406, doi:10.1067/mod.2002.128219.

Law, K. H. & Jouaneh, M. K. (1986), Data modeling for building design, in *Proceedings of the fourth Conference on Computing in Civil Engineering, New York*, ASCE, pp. 21–36.

Lenstra, J. & Rinnooy Kan, A. (1979), Computational complexity of discrete optimization problems, *Annals of Discrete Mathematics*, 4, 121–140, ISSN 0167-5060.

Leon, V. J. & Balakrishnan, R. (1995), Strength and adaptability of problem-space based neighborhoods for resource-constrained scheduling, *OR Spektrum*, 17(2), 173–182, ISSN 0171-6468, doi:10.1007/BF01719262.

Leung, S.-w.; Mak, S. & Lee, B. L. (2008), Using a real-time integrated communication system to monitor the progress and quality of construction works, *Automation in Construction*, 17(6), 749–757, ISSN 09265805, doi:10.1016/j.autcon.2008.02.003.

Levin, A.; Fergus, R.; Durand, F. & Freeman, W. T. (2007), Image and depth from a conventional camera with a coded aperture, *ACM Transactions on Graphics*, 26(3), 70, ISSN 07300301, doi:10.1145/1276377.1276464.

Li, H.; Chen, Z.; Yong, L. & Kong, S. C. (2005), Application of integrated GPS and GIS technology for reducing construction waste and improving construction efficiency, *Automation in Construction*, 14(3), 323–331, ISSN 09265805, doi:10.1016/j.autcon.2004.08.007.

Li, H. & Love, P. (1997), Using Improved Genetic Algorithms to Facilitate Time-Cost Optimization, *Journal of Construction Engineering and Management*, 123(3), 233–237, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1997)123:3(233).

Li, K. & Willis, R. (1992), An iterative scheduling technique for resource-constrained project scheduling, *European Journal of Operational Research*, 56(3), 370–379, ISSN 03772217, doi:10.1016/0377-2217(92)90320-9.

Likins, G. E.; Rausche, F.; Goble, G. & Shafer, N. (1999), Pile Installation Recording System.

Lin, Y.-C.; Cheung, W.-F. & Siao, F.-C. (2014), Developing mobile 2D barcode/RFID-based maintenance management system, *Automation in Construction*, 37, 110–121, ISSN 09265805, doi:10.1016/j.autcon.2013.10.004.

Long, L. D. & Ohsato, A. (2009), A genetic algorithm-based method for scheduling repetitive construction projects, *Automation in Construction*, 18(4), 499–511, ISSN 09265805, doi:10.1016/j.autcon.2008.11.005.

Lowe, D. (1999), Object recognition from local scale-invariant features, in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, ISBN 0-7695-0164-8, pp. 1150–1157, doi:10.1109/ICCV.1999.790410.

Lowe, D. (2004), Distinctive Image Features from Scale-Invariant Keypoints, *Int. J. Comput. Vision*, 60(2), 91–110, doi:10.1023/B:VISI.0000029664.99615.94.

Lu, M.; Chen, W.; Shen, X.; Lam, H. C. & Liu, J. (2007), Positioning and tracking construction vehicles in highly dense urban areas and building construction sites, *Automation in Construction*, 16(5), 647–656, ISSN 09265805, doi:10.1016/j.autcon.2006.11.001.

Lu, M.; Lam, H.-C. & Dai, F. (2008), Resource-constrained critical path analysis based on discrete event simulation and particle swarm optimization, *Automation in Construction*, 17(6), 670–681, doi:10.1016/j.autcon.2007.11.004.

MacCrimmon, K. R. & Ryavec, C. a. (1964), An Analytical Study of the PERT Assumptions, doi:10.1287/opre.12.1.16.

Makadia, A.; Patterson, A. & Daniilidis, K. (2006), Fully Automatic Registration of 3D Point Clouds, *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 1, 1297–1304, ISSN 1063-6919, doi:10.1109/CVPR.2006.122.

Malcolm, D. G.; Roseboom, J. H.; Clark, C. E. & Fazar, W. (1959), Application of a Technique for Research and Development Program Evaluation, *Operations Research*, 7(5), 646–669, ISSN 0030-364X, doi:10.1287/opre.7.5.646.

Marler, R. T. & Arora, J. S. (2004), Survey of multi-objective optimization methods for engineering, *Structural and Multidisciplinary Optimization*, 26(6), 369–395, ISSN 1615147X, doi:10.1007/s00158-003-0368-6.

Martinez, J. & Ioannou, P. (1994), General purpose simulation with Stroboscope, in *Proc. of the Winter Simulation Conference*, ISBN 0-7803-2109-X, ISSN 02750708, pp. 1159–1166, doi:10.1109/WSC.1994.717503.

Marx, A. & König, M. (2011), Preparation of Constraints for Construction Simulation, in *Proceedings of the 2011 ASCE International Workshop on Computing in Civil Engineering*, pp. 1–8.

Marzouk, M. M. & Zaher, M. M. (2015), Tracking Construction Projects Progress using Mobile Hand-Held Devices, in *Proceedings of the 5th International/11th Construction Specialty Conference*, doi:10.13140/RG.2.1.1968.7529.

Masmoudi, M. & Haït, A. (2013), Project scheduling under uncertainty using fuzzy modelling and solving techniques, *Engineering Applications of Artificial Intelligence*, 26(1), 135–149, ISSN 09521976, doi:10.1016/j.engappai.2012.07.012.

Matiukas, V. & Miniotas, D. (2011), Point Cloud Merging for Complete 3D Surface Reconstruction, *Electronics And Electrical Engineering*, 113(7), 73–76, ISSN 2029-5731, doi:10.5755/j01.eee.113.7.616.

Matthews, J.; Love, P. E.; Heinemann, S.; Chandler, R.; Rumsey, C. & Olatunj, O. (2015), Real time progress management: Re-engineering processes for cloud-based BIM in construction, *Automation in Construction*, 58, 38–47, ISSN 09265805, doi:10.1016/j.autcon.2015.07.004.

Mejía, G.; Niño, K.; Montoya, C.; Sánchez, M. A.; Palacios, J. & Amodeo, L. (2016), A Petri Net-based framework for realistic project management and scheduling: An application in animation and videogames, *Computers & Operations Research*, 66, 190–198, ISSN 03050548, doi:10.1016/j.cor.2015.08.011.

Melzner, J.; Zhang, S.; Teizer, J.; Hollermann, S. & Bargstädt, H.-j. (2012), Safety planning based on an object-oriented building model, in *Proceedings of the EG-ICE Conference 2012*.

Merkle, D.; Middendorf, M. & Schmeck, H. (2002), Ant colony optimization for resource-constrained project scheduling, *IEEE Transactions on Evolutionary Computation*, 6(4), 333–346, ISSN 1089-778X, doi:10.1109/TEVC.2002.802450.

Metropolis, N. & Ulam, S. (1949), The Monte Carlo Method, *Journal of the American Statistical Association*, 44(247), 335–341, ISSN 0162-1459, doi:10.1080/01621459.1949.10483310.

Mika, M.; Waligóra, G. & Weglarz, J. (2005), Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models, *European Journal of Operational Research*, 164(3), 639–668, ISSN 03772217, doi:10.1016/j.ejor.2003.10.053.

Möhring, R. & Schulz, A. (2003), Solving project scheduling problems by minimum cut computations, *Management Science*, 49(3), 330–350.

Naresh, B. A. L. & Jahren, C. T. (1997), Communications and tracking for construction vehicles, *Journal of Construction Engineering and Management*, 123(3), 261–268, doi:10.1061/(ASCE)0733-9364(1997)123:3(261).

Navinchandra, D.; Sriram, D. & Logcher, R. (1988), GHOST: Project network generator, *Journal of Computing in Civil Engineering, ASCE*, 2(3), 239 – 254, doi:10.1061/(ASCE)0887-3801(1988)2:3(239).

Navon, R. (2005), Automated project performance control of construction projects, *Automation in Construction*, 14(4), 467–476, ISSN 09265805, doi:10.1016/j.autcon.2004.09.006.

Navon, R. (2007), Research in automated measurement of project performance indicators, *Automation in Construction*, 16(2), 176–188, ISSN 09265805, doi:10.1016/j.autcon.2006.03.003.

Newton, I.; Motte, A. & Chittenden, N. (1850), *Newton's principia: The mathematical principles of natural philosophy*, Geo. P. Putnam.

Ng, R.; Levoy, M.; Brédif, M.; Duval, G.; Horowitz, M. & Hanrahan, P. (2005), Light Field Photography with a Hand-Held Plenoptic Camera, Technical report, Stanford University, doi:10.1.1.163.488.

Nguyen, A. & Le, B. (2013), 3D Point Cloud Segmentation: A survey, in *Proceedings of the IEEE International Conference on Robotics, Automation and Mechatronics*, ISBN 978-1-4799-1201-8, ISSN 2158219X, pp. 225–230, doi:10.1109/RAM.2013.6758588.

Obergrießer, M.; Pfäffinger, A.; Euringer, T. & Borrmann, A. (2009), Automatisierte Auswertung von Produktionsdaten für den geometrischen und prozessualen Soll-Ist-Vergleich bei dem Verfahren der Pfahlherstellung, in *Tagungsband des 21. Forum Bauinformatik, Karlsruhe*.

O'Connor, M.; Elkaim, G. & Parkinson, B. (1995), Kinematic GPS for closed-loop control of farm and construction vehicles, in *Proceedings of the ION GPS*.

Oggier, T.; Lehmann, M.; Kaufmann, R.; Schweizer, M.; Richter, M. & Peter Metzler (2004), An all-solid-state optical range camera for 3D real-time imaging with sub-centimeter depth resolution (SwissRanger), *Proceedings of SPIE*, 5249, 534–545, ISSN 0277786X, doi:10.1117/12.513307.

Ogunmakin, G.; Teizer, J. & Vela, P. (2013), Quantifying Interactions Amongst Construction Site Machines, in *Proceedings of the EG-ICE Workshop on Intelligent Computing in Engineering, Vienna, Austria*.

Oloufa, A. a.; Ikeda, M. & Oda, H. (2003), Situational awareness of construction equipment using GPS, wireless and web technologies, *Automation in Construction*, 12(6), 737–748, ISSN 09265805, doi:10.1016/S0926-5805(03)00057-8.

Özdamar, L. & Ulusoy, G. (1995), A survey on the resource-constrained project scheduling problem, *IIE Transactions*, 27(5), 574–586, ISSN 0740-817X, doi:10.1080/07408179508936773.

Panchal, J. H.; Paredis, C. J.; Allen, J. K. & Mistree, F. (2008), A value-of-information based approach to simulation model refinement, *Engineering Optimization*, 40(3), 223–251, ISSN 0305-215X, doi:10.1080/03052150701690764.

Pareto, V. (1906), *Manuale di economia politica*, volume 13, Societa Editrice.

Park, H.-S.; Thomas, S. R. & Tucker, R. L. (2005), Benchmarking of Construction Productivity, *Journal of Construction Engineering and Management*, 131(7), 772–778, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(2005)131:7(772).

Patterson, J. H. (1976), Project Scheduling: The Impact of Instance Structure on Heuristic Performance, *Naval Research Logistics*, 23(1), 95–123, doi:10.1002/nav.3800230110.

Patterson, J. H. (1984), A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem, *Management science*, 30(7), 854–867, ISSN 0025-1909, doi:10.1287/mnsc.30.7.854.

Pinson, E.; Prins, C. & Rullier, F. (1994), Using tabu search for solving the Resource-Constrained Project Scheduling Problem, in *Proceedings of the 4. International Workshop on Project Management and Scheduling*, pp. 102–106.

Plackett, R. L. (1983), Karl Pearson and the Chi-Squared Test, *International Statistical Review*, 51(1), 59–72.

Pocock, J. B.; Hyun, C. T.; Liu, L. Y. & Kim, M. K. (1996), Relationship between Project Interaction and Performance Indicators, *Journal of Construction Engineering and Management*, 122(2), 165–176, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(1996) 122:2(165).

Poirier, E. a.; Staub-French, S. & Forgues, D. (2015), Measuring the impact of BIM on labor productivity in a small specialty contracting enterprise through action-research, *Automation in Construction*, 58, 74–84, ISSN 09265805, doi:10.1016/j.autcon.2015. 07.002.

Ponz-Tienda, J. L.; Pellicer, E.; Alarcón, L. F. & Rojas-Quintero, J. S. (2015), Integrating Task Fragmentation and Earned Value Method into the Last Planner System using Spreadsheets, in *Proceedings of the Annual Conference of the International Group for Lean Construction*, pp. 63–72.

Ponz-Tienda, J. L.; Yepes, V.; Pellicer, E. & Moreno-Flores, J. (2013), The Resource Leveling Problem with multiple resources using an adaptive genetic algorithm, *Automation in Construction*, 29(February 2016), 161–172, ISSN 09265805, doi:10.1016/j.autcon. 2012.10.003.

Popov, V.; Mikalauskas, S.; Migilinskas, D. & Vainiunas, P. (2006), Complex usage of 4D information modelling concept for building design, estimation, sheduling and determination of effective variant, *Technological and economic development of economy*, 12(2), 91–98, ISSN 20294913, doi:10.1080/13928619.2006.9637728.

Pradhananga, N. & Teizer, J. (2013), Automatic spatio-temporal analysis of construction site equipment operations using GPS data, *Automation in Construction*, 29, 107–122, doi:10.1016/j.autcon.2012.09.004.

Pritsker, A. A. B.; Waiters, L. J. & Wolfe, P. M. (1969), Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach, doi:10.1287/mnsc.16.1.93.

Rabbani, T.; Dijkman, S.; van den Heuvel, F. & Vosselman, G. (2007), An integrated approach for modelling and global registration of point clouds, *ISPRS Journal of Photogrammetry and Remote Sensing*, 61(6), 355–370, ISSN 09242716, doi:10.1016/j.isprsjprs.2006.09.006.

Ranganathananda, S. (1990), *Swami Vivekananda and Human Excellence*, Advaita Ashrama.

Ray, S. J. & Teizer, J. (2012), Real-time construction worker posture analysis for ergonomics training, *Advanced Engineering Informatics*, 26(2), 439–455, ISSN 14740346, doi:10.1016/j.aei.2012.02.011.

Riley, L. (2012), Managing and Controlling Risk in Complex Infrastructure Projects: Using Discrete Event Simulation for Stochastic Scheduling in Construction Engineering, in *Proceedings of the Autumn Simulation Multi-Conference*.

Rivera, J. (2013), A Hybrid Heuristic Algorithm For Solving The Resource Constrained Project Scheduling Problem (RCPSP)., *Revista EIA*, 87–100.

Rommelfanger, H. (1990), Fulpal - An Interactive Method for Solving (Multiobjective) Fuzzy Linear Programming Problems, in Slowinski, R. & Teghem, J. (eds.), *Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty*, Springer Netherlands, volume 6 of *Theory and Decision Library*, ISBN 978-94-010-7449-0, pp. 279–299, doi:10.1007/978-94-009-2111-5{_}14.

Sabuncuoglu, I. & Goren, S. (2009), Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research, *International Journal of Computer Integrated Manufacturing*, 22(2), 138–157.

Sahinoglu, Z. (2008), Ultra-Wideband Positioning Systems, Technical report, Mitsubishi Electric Research Laboratories, Massachusetts.

Saidi, K. S.; Haas, C. T. & Balli, N. a. (2002), The Value of Handheld Computers in Construction, in *Proceedings of the 19th International Symposium on Automation and Robotics in Construction, IAARC, Washington, DC, USA*.

Saidi, K. S.; Teizer, J.; Franaszek, M. & Lytle, A. M. (2011), Static and dynamic performance evaluation of a commercially-available ultra wideband tracking system, *Automation in Construction*, 20(5), 519–530, ISSN 09265805, doi:10.1016/j.autcon.2010.11.018.

Salem, O. & Zimmer, E. (2005), Application of lean manufacturing principles to construction, *Lean construction journal*, 2(2), 51–54.

Salewski, F.; Schirmer, A. & Drexl, A. (1997), Project scheduling under resource and mode identity constraints: Model, complexity, methods, and application, *European Journal of Operational Research*, 102(1), 88–110, ISSN 03772217, doi:10.1016/S0377-2217(96)00219-6.

Schäffter, M. W. (1997), Scheduling with forbidden sets, *Discrete Applied Mathematics*, 72(1–2), 155–166, ISSN 0166-218X, doi:10.1016/S0166-218X(96)00042-X.

Scherer, R. J. & Ismail, A. (2011), Process-Based Simulation Library for Construction Project Planning, in *Proceedings of the 2011 Winter Simulation Conference*.

Schirmer, A. (2000), Case-based reasoning and improved adaptive search for project scheduling, *Naval Research Logistics*, 47(3), 201–222, ISSN 0894-069X, doi:10.1002/(SICI)1520-6750(200004)47:3<201::AID-NAV2>3.0.CO;2-L.

Schirmer, a. (2001), Resource-constrained project scheduling: An evaluation of adaptive control schemes for parameterized sampling heuristics, *International Journal of Production Research*, 39(7), 1343–1365, ISSN 0020-7543, doi:10.1080/00207540010022368.

Schruben, L. (1983), Simulation modeling with event graphs, *Communications of the ACM*, 26(11), 957–963, ISSN 00010782, doi:10.1145/182.358460.

Schruben, L. & Yücesan, E. (1993), Complexity of Simulation Models: A Graph Theoretic Approach, in *Proceedings of 1993 Winter Simulation Conference*, ISBN 0-7803-1381-X, pp. 641–649, doi:10.1109/WSC.1993.718302.

Senouci, A. & Eldin, N. (2004), Use of genetic algorithms in resource scheduling of construction projects, *Journal of Construction Engineering and Management*, 130(6), 869–877, doi:10.1061/(ASCE)0733-9364(2004)130:6(869).

Shi, B. Q.; Liang, J. & Liu, Q. (2011), Adaptive simplification of point cloud using k-means clustering, *CAD Computer Aided Design*, 43(8), 910–922, ISSN 00104485, doi:10.1016/j.cad.2011.04.001.

Shi, J. J. (1999), Activity-based construction (ABC) modeling and simulation method, *Journal of construction engineering and management*, 125(5), 354–360.

Shih, N.-J. (2003), The Application of 3D Scanner in the Representation of Building Construction Site, Technical report, NIST Special Publication SP.

Shih, N.-J. & Wang, P.-H. (2004), Point-Cloud-Based Comparison between Construction Schedule and As-Built Progress: Long-Range Three-Dimensional Laser Scanner's Approach, *Journal of Architectural Engineering*, 10(3), 98–102, ISSN 1076-0431, doi:10.1061/(ASCE)1076-0431(2004)10:3(98).

Sigalov, K. & König, M. (2015), Similarity estimation of BIM-based schedules, in *Proceedings of the EG-ICE Workshop on Intelligent Computing in Engineering*, pp. 60–69.

Silpa-Anan, C. & Hartley, R. (2008), Optimised KD-trees for fast image descriptor matching, *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8, ISSN 1063-6919, doi:10.1109/CVPR.2008.4587638.

Simpson, W. P. & Patterson, J. H. (1996), A multiple-tree search procedure for the resource-constrained project scheduling problem, *European Journal of Operational Research*, 89(3), 525–542, ISSN 03772217, doi:10.1016/0377-2217(94)00247-9.

Smirnov, N. (1948), Table for Estimating the Goodness of Fit of Empirical Distributions, *The Annals of Mathematical Statistics*, 19(2), 279–281, ISSN 0003-4851, doi:10.1214/aoms/1177733256.

Son, H.; Kim, C. & Choi, K. (2010), Rapid 3D object detection and modeling using range data from 3D range imaging camera for heavy equipment operation, *Automation in Construction*, 19(7), 898–906, ISSN 09265805, doi:10.1016/j.autcon.2010.06.003.

Song, J.; Haas, C.; Caldas, C. & Liapi, K. (2005), Locating Materials on Construction Site Using Proximity Techniques, in *Proceedings of the Construction Research Congress*, doi:10.1061/40754(183)108.

Song, J.; Haas, C. T. & Caldas, C. H. (2006), Tracking the Location of Materials on Construction Projects, *Journal of Construction Engineering and Management*, 132(9), 911–918, doi:10.1061/(ASCE)0733-9364(2006)132:9(911).

Špačková, O. & Straub, D. (2013), Dynamic Bayesian Network for Probabilistic Modeling of Tunnel Excavation Processes, *Computer-Aided Civil and Infrastructure Engineering*, 28(1), 1–21, ISSN 10939687, doi:10.1111/j.1467-8667.2012.00759.x.

Sprecher, A.; Kolisch, R. & Drexl, A. (1995), Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem, *European Journal of Operational Research*, 80(5), 94–102, doi:10.1016/0377-2217(93)E0294-8.

Stilla, U.; Tuttas, S.; Braun, A. & Borrmann, A. (2015), Baufortschrittskontrolle basierend auf photogrammetrischen Punktwolken und 4D-Gebäudeinformationsmodellen (BIM), in Hanke, K. & Weinold, T. (eds.), *Proceedings der 18. Internationaler Geodätischen Woche*, ISBN 9783879075546, pp. 157–163.

Su, Y. Y.; Hashash, Y. M. a. & Liu, L. Y. (2006), Integration of Construction As-Built Data Via Laser Scanning with Geotechnical Monitoring of Urban Excavation, *Journal of Construction Engineering and Management*, 132(12), 1234–1241, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(2006)132:12(1234).

Szczesny, K. & König, M. (2015), Reactive scheduling based on actual logistics data by applying simulation-based optimization, *Visualization in Engineering*, 3(10), ISSN 2213-7459, doi:10.1186/s40327-015-0020-8.

Szczesny, K.; König, M.; Laußat, L. & Helmus, M. (2013), Integration of Uncertain Real-Time Logistics Data for Reactive Scheduling using Fuzzy Set Theory, in *Proceedings of the 30th International Symposium of Automation and Robotics in Construction and Mining (ISARC)*, pp. 691–698.

Talbot, F. B. & Patterson, J. H. (1978), An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems, *Management Science*, 24(11), 1163–1174, ISSN 0025-1909, doi:10.1287/mnsc.24.11.1163.

Tang, P.; Huber, D.; Akinci, B.; Lipman, R. & Lytle, A. (2010), Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques, *Automation in Construction*, 19(7), 829–843, ISSN 09265805, doi: 10.1016/j.autcon.2010.06.007.

Tauscher, E. (2011), *Vom Bauwerksinformationsmodell zur Terminplanung*, Ph.D. thesis, Bauhaus-Universität Weimar.

Tauscher, E.; Smarsly, K.; König, M. & Beucke, K. (2014), Automated Generation of Construction Sequences using Building Information Models, in *Proceedings of the International Conference on Computing in Civil and Building Engineering*, ISBN 4047503398, ISSN 0926-5805, pp. 745–752, doi:10.1061/9780784413616.053.

Teizer, J. (2008), 3D range imaging camera sensing for active safety in construction, *Electronic Journal of Information Technology in Construction*, 13(April), 103–117, ISSN 14036835.

Teizer, J.; Caldas, C. H. C. & Haas, C. T. C. (2007a), Real-time three-dimensional occupancy grid modeling for the detection and tracking of construction resources, *Journal of Construction Engineering and Management*, 133(11), 880–888, ISSN 0733-9364, doi:10.1061/(ASCE)0733-9364(2007)133:11(880).

Teizer, J.; Lao, D. & Sofer, M. (2007b), Rapid automated monitoring of construction site activities using ultra-wideband, in *24th International Symposium on Automation and Robotics in Construction, ISARC 2007*, 2, ISBN 9788190423519 (ISBN), pp. 23–28.

Teizer, J. & Vela, P. A. (2009), Personnel tracking on construction sites using video cameras, *Advanced Engineering Informatics*, 23(4), 452–462, ISSN 14740346, doi: 10.1016/j.aei.2009.06.011.

Thesen, A. (1976), Heuristic Scheduling of Activities under Resource and Precedence Restrictions, *Management Science*, 23(4), 412–422, ISSN 0025-1909, doi:10.1287/mnsc.23.4.412.

Thomas, P. & Salhi, S. (1998), A tabu search approach for the resource constrained project scheduling problem, *Journal of Heuristics*, 4(3), 123–139.

Tocher, K. D. (1963), *The Art of Simulation*, The English Universities Press Ltd., doi: 10.1080/03043799408923267.

Toklu, Y. C. (2002), Application of genetic algorithms to construction scheduling with or without resource constraints, *Canadian Journal of Civil Engineering*, 29(3), 421–429, ISSN 0315-1468, doi:10.1139/l02-034.

Tormos, P. & Lova, A. (2001), A Competitive Heuristic Solution Technique for Resource-Constrained Project Scheduling, *Annals of Operations Research*, 102(1-4), 65–81, ISSN 02545330, doi:10.1023/A:1010997814183.

Trevor, A. J. B.; Gedikli, S.; Rusu, R. B. & Christensen, H. I. (2013), Efficient organized point cloud segmentation with connected components, in *Proceedings of Semantic Perception Mapping and Exploration*.

Triggs, B.; McLauchlan, P. F.; Hartley, R. I. & Fitzgibbon, A. W. (2000), Bundle Adjustment - A Modern Synthesis, in Triggs, B.; Zisserman, A. & Szeliski, R. (eds.), *Vision Algorithms: Theory and Practice*, Springer, pp. 298–372, doi:10.1007/3-540-44480-7{_}21.

Tuttas, S.; Braun, A.; Borrmann, A. & Stilla, U. (2014), Comparison of Photogrammetric Point Clouds with BIM Building Elements for Construction Progress Monitoring, in *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume XL-3, ISSN 2194-9034, pp. 341–345, doi:10.5194/isprsarchives-XL-3-341-2014.

Tuttas, S.; Braun, A.; Borrmann, A. & Stilla, U. (2015), Validation of Bim Components By Photogrammetric Point Clouds for Construction Site Monitoring, in *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume II-3/W4, ISSN 2194-9050, pp. 231–237, doi:10.5194/isprsannals-II-3-W4-231-2015.

Ullman, S. (1979), The Interpretation of Structure from Motion, *Proceedings of the Royal Society B: Biological Sciences*, 203, 405–426, ISSN 0962-8452, doi:10.1098/rspb.1979.0006.

Valls, V. & Ballestín, F. (2002), A Hybrid Genetic Algorithm for the RCPSP with the Peak Crossover Operator, in *Proceedings of 8th International Workshop on Project Management and Scheduling*.

Valls, V.; Ballestín, F. & Quintanilla, S. (2004), A population-based approach to the resource-constrained project scheduling problem, *Annals of Operations Research*, 131(1-4), 305–324, ISSN 02545330, doi:10.1023/B:ANOR.0000039524.09792.c9.

Valls, V.; Ballestin, F. & Quintanilla, S. (2005), Justification and RCPSP: A technique that pays, *European Journal of Operational Research*, 165(2), 375–386, doi:10.1016/j.ejor.2004.04.008.

Valls, V.; Ballestín, F. & Quintanilla, S. (2008), A hybrid genetic algorithm for the resource-constrained project scheduling problem, *European Journal of Operational Research*, 185(2), 495–508, ISSN 03772217, doi:10.1016/j.ejor.2006.12.033.

Valls, V.; Quintanilla, S. & Ballestin, F. (2001), An evolutionary approach to the resource-constrained project scheduling problem, in *Proceedings of the 4th Metaheuristics International Conference (MIC)*, pp. 217–220.

Valls, V.; Quintanilla, S. & Ballestin, F. (2003), Resource-constrained project scheduling: A critical activity reordering heuristic, *European Journal of Operational Research*, 149(2), 282–301, ISSN 03772217, doi:10.1016/S0377-2217(02)00768-3.

Van Peteghem, V. & Vanhoucke, M. (2014), An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances, *European Journal of Operational Research*, 235(1), 62–72, ISSN 03772217, doi:10.1016/j.ejor.2013.10.012.

Viljamaa, E. & Peltomaa, I. (2014), Intensified construction process control using information integration, *Automation in Construction*, 39, 126–133, ISSN 09265805, doi:10.1016/j.autcon.2013.08.015.

Waligóra, G. (2009), Tabu search for discrete–continuous scheduling problems with heuristic continuous resource allocation, *European Journal of Operational Research*, 193(3), 849–856, ISSN 03772217, doi:10.1016/j.ejor.2007.11.009.

Wang, X.; Truijens, M.; Hou, L.; Wang, Y. & Zhou, Y. (2014), Integrating Augmented Reality with Building Information Modeling: Onsite construction process controlling for liquefied natural gas industry, *Automation in Construction*, 40, 96–105, ISSN 09265805, doi:10.1016/j.autcon.2013.12.003.

Webb, J. A. & Aggarwal, J. (1982), Structure from motion of rigid and jointed objects, *Artificial Intelligence*, 19(1), 107–130, ISSN 00043702, doi:10.1016/0004-3702(82) 90023-6.

Weibull, W. (1951), A Statistical Distribution Function of Wide Applicability, *Journal of Applied Mechanics*, 103, 293–97.

Whitehouse, G. E. (1979), GENRES: An extension of Brooks Algorithm for project scheduling with resource constraints, *Computers & Industrial Engineering*, 3(3), 261–268, doi:10.1016/0360-8352(79)90020-2.

Wiest, J. D. (1962), The Scheduling of large Projects with limited resources., Technical report, Carnegie Institute of Technology.

Wiest, J. D. (1967), A heuristic model for scheduling large projects with limited resources, *Management Science*, 13(6), B—-359.

Winstanley, G.; Chacon, M. a. & Levitt, R. E. (1993a), An integrated project planning environment, *Intelligent SystemsEngineering*, 2(2), 91–106.

Winstanley, G.; Chacon, M. A. & Levitt, R. E. (1993b), Model-Based Planning: Scaled-Up Construction Application, *Journal of Computing in Civil Engineering*, 7(2), 199–217.

Wu, C. (2011), VisualSFM: A Visual Structure from Motion System.

Wu, C.; Agarwal, S.; Curless, B. & Seitz, S. M. (2011), Multicore Bundle Adjustment, *CVPR 2011*.

Wu, I.-C. C.; Borrmann, A.; Beißert, U.; König, M. & Rank, E. (2010), Bridge construction schedule generation with pattern-based construction methods and constraint-based simulation, *Advanced Engineering Informatics*, 24(4), 379–388, ISSN 14740346, doi:10.1016/j.aei.2010.07.002.

Wübbeler, G.; Krystek, M. & Elster, C. (2008), Evaluation of measurement uncertainty and its numerical calculation by a Monte Carlo method, *Measurement Science and Technology*, 19(8), 084009, ISSN 0957-0233, doi:10.1088/0957-0233/19/8/084009.

Yagiz, S.; Gokceoglu, C.; Sezer, E. & Iplikci, S. (2009), Application of two non-linear prediction tools to the estimation of tunnel boring machine performance, *Engineering Applications of Artificial Intelligence*, 22(4-5), 808–814, ISSN 09521976, doi:10.1016/j.engappai.2009.03.007.

Yang, J.; Arif, O.; Vela, P. A.; Teizer, J. & Shi, Z. (2010), Tracking multiple workers on construction sites using video cameras, *Advanced Engineering Informatics*, 24(4), 428–434, ISSN 14740346, doi:10.1016/j.aei.2010.06.008.

Yang, J.; Vela, P. A.; Shi, Z. & Teizer, J. (2009), Probabilistic Multiple People Tracking through Complex Situations Multiple People Tracking, in *Proceedings of the 11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pp. 79–86.

Yang, J.; Vela, P. a.; Teizer, J. & Shi, Z. K. (2011), Vision-based crane tracking for understanding construction activity, *2011 ASCE International Workshop on Computing in Civil Engineering*, 28(February), 258–265, ISSN 0887-3801, doi:10.1061/41182(416)32.

Yang, Z. Y. & Wang, Z. F. (2010), Comparison between AON and AOA network diagrams, in *Proceedings of the IEEE 17th International Conference on Industrial Engineering and Engineering Management*, ISBN 9781424464814, pp. 1507–1509, doi:10.1109/ICIEEM.2010.5646036.

Yates, F. (1934), Tables Involving Small Numbers and the Chi-Squared Test, *Supplement to the Journal of the Royal Statistical Society*, 1(2), 217–235.

Yu, P. L. & Leitmann, G. (1974), Compromise solutions, domination structures, and Salukvadze's solution, *Journal of Optimization Theory and Applications*, 13(3), 362–378, ISSN 00223239, doi:10.1007/BF00934871.

Zadeh, L. A. (1978), Fuzzy sets as a basis for a theory of possibility, *Fuzzy sets and systems*, 1, 3–28.

Zhang, H.; Li, H. & Tam, C. (2006), Permutation-based particle swarm optimization for resource-constrained project scheduling, *Journal of computing in civil engineering*, 20, 141–149.

Zhang, H.; Li, X.; Li, H. & Huang, F. (2005a), Particle swarm optimization-based schemes for resource-constrained project scheduling, *Automation in Construction*, 14(3), 393–404, ISSN 09265805, doi:10.1016/j.autcon.2004.08.006.

Zhang, H.; Tam, C. & Li, H. (2005b), Activity object-oriented simulation strategy for modeling construction operations, *Journal of Computing in Civil engineering*, 19(3), 313–322.

Zheng, Y.-J. (2015), Water wave optimization: A new nature-inspired metaheuristic, *Computers & Operations Research*, 55, 1–11, ISSN 03050548, doi:10.1016/j.cor.2014.10.008.

List of Figures

List of Tables

A List of Symbols

α	Shape parameter of the beta distribution (Section 2.2.1)
α^*	Probability of accepting a false hypothesis (Section 2.2.5)
α_i	Slack indicator of activity i (Section 3.2.1)
$\boldsymbol{A}_g$	The set of active activities at stage g (Section 3.4.4)
$B(\alpha_j, \beta_j)$	Function taking a sample from a beta distribution with parameters α_j and β_j (Section 4.5)
β	Shape parameter of the beta distribution (Section 2.2.1)
γ_k	Resource constrainedness of resource k (Section 3.4.2)
Γ	Gamma function (Section 2.2.3)
$\boldsymbol{C}$	The set of already scheduled activities (Section 3.4.4)
$\boldsymbol{D}_g$	The set of delayed activities at stage g (Section 4.1)
d_j	Duration of activity j (Section 3.4.7)
$\bar{d}_j$	Realized duration of activity j (Section 4.5)
$\hat{d}_j$	Maximum increase of duration of activity j (Section 4.5)
$\boldsymbol{E}_i$	The set of possible start times for activity i (Section 3.4.7)
$\mathcal{E}(g)$	The set of activities which can be started precedence-feasible in stage g (Section 3.4.4)
$\mathcal{E}_r(g)$	The set of activities which can be started resource- and precedence-feasible in stage g (Section 3.4.4)
e	Euler's number (Section 2.2.2)
θ	Scale parameter of Gamma distribution (Section 2.2.3)
F^o	The utopia point in a search space (Section 3.2.2)
F_i	The finish time of activity i
$\mathcal{F}(j)$	Maximum finish time of all predecessors of activity j (Section 3.4.4)
$\boldsymbol{J}$	The set of all activities (Sections 3.4.2)
$\mathcal{K}$	The set of Monte Carlo simulation runs (Section 4.5)
κ_i	Average percentage increase in activity i's duration (Section 3.2.1)
k	Shape parameter of the Weibull distribution (Section 2.2.4)
k_r	The limit of resource r (Section 3.4.7)
k_{ir}	The consumption of resource r for activity i per time period (Section 3.4.7)
λ	Scale parameter of the Weibull distribution (Section 2.2.4)
L_i	Late start time of activity i (Section 3.4.4)
μ	Mean of a random variable (Section 2.2)

π	The ratio of a circle's circumference to its diameter (Section 2.2.2)
m_k	The makespan of the schedule generated in the k-th Monte Carlo simulation run (Section 4.5)
N	Total number of activities (Section 3.4.8.8)
N_{succ_i}	The number of direct successors of activity i (Section 3.2.1)
$\mathcal{O}_j$	Objective function with index j for optimizing a schedule involving uncertainty (Section 4.5)
ξ_k	Resource strength of resource k (Section 3.4.2)
$\mathcal{P}_j$	Priority value of activity j (Section 4.1)
ρ_{ik}	Indicator whether activity i uses resource k (Section 3.4.2)
σ	Standard deviation of a random variable (Section 2.2)
$\boldsymbol{Q}_{it}$	The set of time periods where activity i is in progress if it was started at time t (Section 3.4.7)
$\mathcal{R}_j$	Fractional part of the priority value $\mathcal{P}_j$ of activity j (Section 4.1)
$\boldsymbol{R}$	The set of all resources (Section 3.4.7)
r_{ik}	The number of units of resource k required by activity i (Section 3.4.2)
s_i	Slack time of activity i (Section 3.3.3)
$\boldsymbol{S}$	The set of all schedules (Section 3.4.4)
$\boldsymbol{S}_A$	The set of all active schedules (Section 3.4.4)
$\boldsymbol{S}_F$	The set of all feasible schedules (Section 3.4.4)
$\boldsymbol{S}_{ND}$	The set of all non-delay schedules (Section 3.4.4)
$\boldsymbol{S}_{PA}$	The set of all pseudo-active schedules (Section 4.1)
$\boldsymbol{S}_{SA}$	The set of all semi-active schedules (Section 3.4.4)
Θ_j	Metric with index j for a given schedule involving uncertainty (Section 4.5)
$\boldsymbol{T}$	The set of all time periods (Section 3.4.7)
t_E	Expected activity duration in PERT (Section 3.3.4)
t_{ES}	Early start time in PERT (Section 3.3.4)
t_{EF}	Early finish time in PERT (Section 3.3.4)
t_{LS}	Late start time in PERT (Section 3.3.4)
t_{LF}	Late finish time in PERT (Section 3.3.4)
t_M	Most-likely activity duration in PERT (Section 3.3.4)
t_O	Optimistic activity duration in PERT (Section 3.3.4)
t_P	Pessimistic activity duration in PERT (Section 3.3.4)
t_S	Slack time in PERT (Section 3.3.4)
τ_e	Performance factor of an excavation process (Section 2.1)
$\boldsymbol{U}_s$	Set of possible swaps for a schedule (Section 4.3)
$\boldsymbol{U}_d$	Set of possible delays for a schedule (Section 4.3)
u_{ir}	The overall demand of resource r by activity i (Section 3.4.7)
$\boldsymbol{V}$	The set of precedence constraints (Section 3.4.7)
w_i	The weight of criterion i (Section 3.2.2)
ϕ_k	The average of resource k required by all activities requiring at least one unit (Section 3.4.2)
$\boldsymbol{X}$	The set of all feasible solutions (Section 3.2.2)
Ψ	The total possible number of swaps (Section 3.4.8.8)

B Benchmark Results

B.1 PSPLIB Results

The data listed in the tables contains the following abbreviations:

- EA: Evolutionary algorithm by Valls et al. (2001).
- GA: Genetic algorithm by Alcaraz & Maroto (2001).
- GH: Greedy heuristic algorithm by Goncharov (2011).
- HSS: Hybrid scatter search by Debels et al. (2006).
- ILDTS: Iterative limited depth tree search algorithm presented in Section 4.3.
- PSO: Particle swarm optimization algorithm by Zhang et al. (2005a).
- PSO2: Particle swarm optimization algorithm by Chen et al. (2010).
- RL: Agent-based reinforcement learning by Jedrzejowicz & Ratajczak-Ropel (2014).
- WWO: Water wave optimization algorithm by Zheng (2015), presented in Section 4.2.

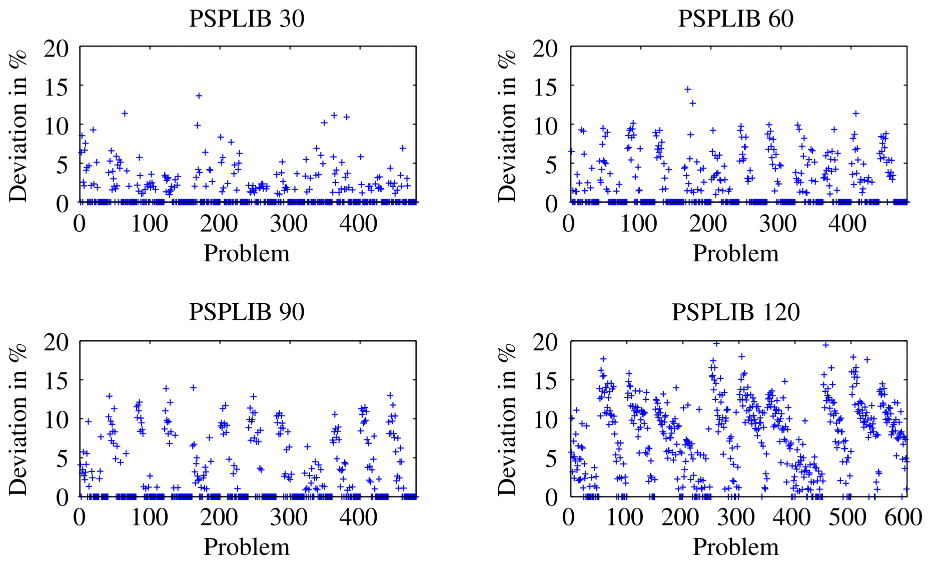


Figure B.1: Deviation of best of $500,000$ random schedules from best known solutions for PSPLIB problems.

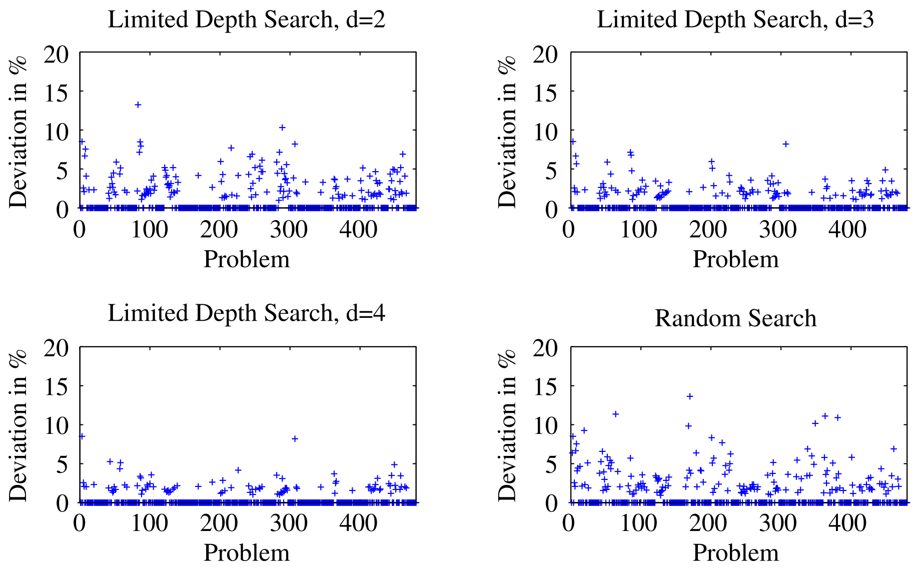


Figure B.2: Deviation of algorithms from optima for the PSPLIB $j30$ problem set

Algorithm	Parameters	r_{opt}	$\bar{x}$	t	n_f
HSS	$n = 500000$	99.38%	0.01%	$7,160ms$	$500,000$
RL	$s = RL123$	n/a	0.02%	$34,960ms$	n/a
WWO	$n_p = 10000, i = 1000$	97.71%	0.04%	$156,677ms$	$10,061,675$
WWO	$n_p = 1000, i = 1000$	96.25%	0.06%	$2,2735ms$	$1,006,492$
EA		93.54%	0.10%	$590ms$	n/a
ILDS	$d = 3, n = 100$	93.33%	0.14%	n/a	n/a
WWO	$n_p = 100, i = 10000$	91.04%	0.16%	$14,630ms$	$1,000,700$
WWO	$n_p = 100, i = 1000$	91.25%	0.17%	$2,286ms$	$100,682$
ILDS	$d = 2, n = 100$	90.21%	0.21%	n/a	n/a
GA	$o_c = TP\_FBC$	n/a	0.24%	n/a	$5,000$
ILDS	$d = 5, n = 10$	84.80%	0.34%	n/a	n/a
PSO	$n_i = 125$	96.40%	0.42%	n/a	$5,000$
ILDS	$d = 4, n = 10$	80.42%	0.44%	n/a	n/a
WWO	$n_p = 10, i = 10000$	79.58%	0.49%	$2,129ms$	$100,083$
PSO2		n/a	0.54%	n/a	n/a
ILDS	$d = 3, n = 10$	74.38%	0.65%	n/a	n/a
WWO	$n_p = 1000, i = 100$	70.83%	0.88%	$1,756ms$	$102,087$
ILDS	$d = 2, n = 10$	67.29%	1.10%	$2,784ms$	$43,402$
Random	$n = 500000$	65.42%	1.16%	$49,967ms$	$500,000$
WWO	$n_p = 10, i = 500$	67.92%	1.23%	$121ms$	$5,036$
WWO	$n_p = 100, i = 100$	63.33%	1.52%	$176ms$	$10,207$
WWO	$n_p = 50, i = 100$	60.42%	1.83%	$125ms$	$5,103$
ILDS	$d = 2$	53.33%	3.57%	$258ms$	$4,216$
ILDS	$d = 1$	42.92%	5.17%	$6.95ms$	77

Table B.1: Ratio of optima of the PSPLIB $j30$ problems found by algorithms and the respective mean deviations

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	93.13%	0.07%	$19,610ms$	$500,000$
EA		86.46%	0.17%	$1,870ms$	n/a
RL	$s = RL123$	n/a	0.51%	$62,300ms$	n/a
GA	$o_c = TP\_FBC$	n/a	0.59%	n/a	$5,000$
ILDS	$d = 2, n = 100$	70.83%	0.87%	$330,123ms$	$3,347,322$
WWO	$n_p = 1000, i = 1000$	73.75%	0.87%	$69,396ms$	$1,009,426$
WWO	$n_p = 100, i = 10000$	71.25%	1.29%	$44,261ms$	$1,001,044$
ILDS	$d = 3$	64.38%	1.31%	n/a	n/a
WWO	$n_p = 100, i = 1000$	70.00%	1.32%	$7,045ms$	$100,940$
ILDS	$d = 2, n = 10$	59.38%	2.01%	$32,381ms$	$302,723$
WWO	$n_p = 10, i = 10000$	64.38%	2.11%	$6,675ms$	$100,119$
Random	$n = 500000$	57.30%	2.14%	$218,751ms$	$500,000$
GH		68.33%	2.37%	n/a	n/a
WWO	$n_p = 1000, i = 100$	61.04%	2.78%	$5,141ms$	$102,042$
PSO2		n/a	3.71%	n/a	n/a
WWO	$n_p = 10, i = 500$	52.50%	4.05%	$364ms$	$5,029$
WWO	$n_p = 50, i = 100$	53.12%	4.06%	$379ms$	$5,102$
ILDS	$d = 2$	48.96%	4.97%	$3,096ms$	$30,581$
ILDS	$d = 1$	40.00%	6.72%	$45ms$	207

Table B.2: Ratio of optima of the PSPLIB $j60$ problems found by algorithms and the respective mean deviations

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	91.04%	0.07%	$38,870ms$	$500,000$
EA		88.75%	0.16%	$4,900ms$	n/a
RL	$s = RL123$	n/a	0.98%	$77,290ms$	n/a
WWO	$n_p = 1000, i = 1000$	71.67%	1.61%	$133,577ms$	$1,011,275$
ILDS	$d = 3, n = 10$	65.42%	1.62%	n/a	n/a
WWO	$n_p = 100, i = 1000$	68.12%	2.19%	$14,601ms$	$101,047$
ILDS	$d = 2, n = 10$	60.83%	2.31%	n/a	n/a
Random	$n = 500000$	57.71%	2.64%	$353,538ms$	$500,000$
WWO	$n_p = 10, i = 10000$	64.38%	2.99%	$13,930ms$	$100,148$
WWO	$n_p = 1000, i = 100$	60.42%	3.65%	$10,533ms$	$102,044$
PSO2		n/a	3.79%	n/a	n/a
WWO	$n_p = 50, i = 100$	51.04%	4.92%	$777ms$	$5,102$
WWO	$n_p = 10, i = 500$	48.12%	5.27%	$749ms$	$5,027$
ILDS	$d = 2$	47.08%	5.48%	$19,970ms$	$101,220$
ILDS	$d = 1$	37.71%	6.60%	$179ms$	369

Table B.3: Ratio of optima of the PSPLIB $j90$ problems found by algorithms and the respective mean deviations

Algorithm	Parameters	r_{best}	$\bar{x}$	t	n_f
HSS	$n = 500000$	72.33%	0.28%	$69,570ms$	$500,000$
EA		50.17%	0.75%	$35,000ms$	n/a
GA	$o_c = TP\_FBC$	n/a	1.36%	n/a	$5,000$
RL	$s = RL123$	n/a	2.91%	$202,410ms$	n/a
WWO	$n_p = 1000, i = 1000$	25.33%	5.48%	$259,471ms$	$1,020,095$
ILDS	$d = 2$	11.83%	6.18%	$186,940ms$	$541,750$
WWO	$n_p = 100, i = 1000$	20.00%	7.22%	$27,658ms$	$101,802$
Random	$n = 500000$	14.17%	7.41%	$482,920ms$	$500,000$
WWO	$n_p = 10, i = 10000$	13.83%	9.05%	$26,486ms$	$100,290$
PSO2		n/a	9.38%	n/a	n/a
WWO	$n_p = 1000, i = 100$	9.17%	10.52%	$19,820ms$	$102,101$
WWO	$n_p = 100, i = 100$	6.17%	12.41%	$1,956ms$	$10,209$
ILDS	$d = 1$	1.67%	14.68%	$866ms$	994
WWO	$n_p = 10, i = 100$	2.67%	14.98%	$288ms$	$1,021$

Table B.4: Ratio of optima of the PSPLIB $j120$ problems found by algorithms and the respective mean deviations

B.2 UPSPLIB Results

Result \ Objective	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.05	1.02	1.04	1.04	1.03	1.03	1.04	1.04
$\mathcal{O}_2$ (mean)	1.11	1.11	1.11	1.13	1.11	1.11	1.12	1.11
$\mathcal{O}_3$ (max)	1.16	1.16	1.16	1.21	1.16	1.18	1.20	1.18
$\mathcal{O}_4$ (stdev)	0.89	0.89	1.02	1.03	0.96	0.98	0.95	0.92

Table B.5: Best solutions of water wave optimization on the uncertainty-extended PSPLIB $j60u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Result \ Objective	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.21	1.21	1.22	1.54	1.22	1.22	1.22	1.22
$\mathcal{O}_2$ (mean)	1.33	1.32	1.32	1.68	1.32	1.32	1.32	1.32
$\mathcal{O}_3$ (max)	1.49	1.47	1.46	1.85	1.46	1.47	1.46	1.47
$\mathcal{O}_4$ (stdev)	2.82	2.41	2.09	2.47	2.11	2.17	2.09	2.09

Table B.6: Mean solutions of water wave optimization on the uncertainty-extended PSPLIB $j60u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Result \ Objective	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.46	1.46	1.49	2.36	1.48	1.53	1.56	1.49
$\mathcal{O}_2$ (mean)	1.68	1.59	1.60	2.62	1.62	1.67	1.66	1.62
$\mathcal{O}_3$ (max)	1.91	1.97	2.11	2.77	1.88	2.09	2.14	2.06
$\mathcal{O}_4$ (stdev)	12.79	9.44	7.04	8.30	5.72	9.87	8.01	6.24

Table B.7: Worst solutions of water wave optimization on the uncertainty-extended PSPLIB $j60u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.09	1.08	1.09	1.09	1.09	1.09	1.08	1.09
$\mathcal{O}_2$ (mean)	1.16	1.15	1.15	1.15	1.16	1.15	1.16	1.16
$\mathcal{O}_3$ (max)	1.23	1.21	1.20	1.22	1.22	1.21	1.21	1.21
$\mathcal{O}_4$ (stdev)	1.48	1.57	1.48	1.56	1.51	1.56	1.54	1.58

Table B.8: Best solutions of water wave optimization on the uncertainty-extended PSPLIB $j90u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.27	1.27	1.27	1.48	1.27	1.27	1.28	1.28
$\mathcal{O}_2$ (mean)	1.38	1.37	1.38	1.61	1.37	1.38	1.38	1.38
$\mathcal{O}_3$ (max)	1.53	1.51	1.52	1.79	1.51	1.52	1.52	1.52
$\mathcal{O}_4$ (stdev)	4.27	3.79	3.70	4.36	3.68	3.63	3.56	3.56

Table B.9: Mean solutions of water wave optimization on the uncertainty-extended PSPLIB $j90u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.62	1.55	1.62	2.03	1.61	1.61	1.64	1.64
$\mathcal{O}_2$ (mean)	1.81	1.73	1.80	2.28	1.75	1.79	1.84	1.80
$\mathcal{O}_3$ (max)	2.13	2.01	2.03	2.54	2.09	2.13	2.08	2.14
$\mathcal{O}_4$ (stdev)	17.38	9.85	10.16	14.18	10.44	11.50	11.11	12.47

Table B.10: Worst solutions of water wave optimization on the uncertainty-extended PSPLIB $j90u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Objective / Result	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.13	1.11	1.10	1.19	1.11	1.12	1.13	1.13
$\mathcal{O}_2$ (mean)	1.21	1.16	1.16	1.27	1.21	1.22	1.18	1.20
$\mathcal{O}_3$ (max)	1.31	1.33	1.32	1.37	1.33	1.31	1.27	1.32
$\mathcal{O}_4$ (stdev)	2.06	2.05	2.15	2.50	2.28	2.13	2.00	1.65

Table B.11: Best solutions of water wave optimization on the uncertainty-extended PSPLIB $j120u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Result \ Objective	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.39	1.39	1.40	1.63	1.39	1.40	1.41	1.42
$\mathcal{O}_2$ (mean)	1.54	1.52	1.52	1.79	1.52	1.53	1.53	1.54
$\mathcal{O}_3$ (max)	1.73	1.70	1.71	1.99	1.70	1.71	1.71	1.72
$\mathcal{O}_4$ (stdev)	6.81	6.01	5.66	6.34	5.77	5.69	5.58	5.57

Table B.12: Mean solutions of water wave optimization on the uncertainty-extended PSPLIB $j120u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.

Result \ Objective	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$
$\mathcal{O}_1$ (min)	1.68	1.59	1.65	2.08	1.68	1.63	1.68	1.67
$\mathcal{O}_2$ (mean)	1.87	1.79	1.85	2.30	1.83	1.81	1.84	1.83
$\mathcal{O}_3$ (max)	2.24	2.16	2.14	2.56	2.07	2.14	2.26	2.29
$\mathcal{O}_4$ (stdev)	26.36	16.09	15.35	15.07	14.15	13.66	13.69	14.83

Table B.13: Worst solutions of water wave optimization on the uncertainty-extended PSPLIB $j120u$ set for the different objective functions $\mathcal{O}_1$ to $\mathcal{O}_8$. The result measures have been obtained through 10,000 Monte Carlo runs of the final solution.