ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis in Analytics and Operations Research in Logistics

The Movie Scheduling Problem: Forecasting and Optimization

Name student: **Merel Groen**

Student ID number: **492835**

Supervisor Erasmus University Rotterdam: **prof. dr. Dennis Huisman**

Second assessor Erasmus University Rotterdam: **Mette Wagenvoort**

Supervisors Lynxx: **Kim de Groot** and **Kevin Mann**

September 14, 2023

Abstract

Multiplex cinemas have gained significant popularity in recent years. These cinemas feature a number of rooms of different sizes and screen types. The large size of these movie theatres, coupled with diverse scheduling requirements and preferences, gives rise to a complex scheduling problem. In this thesis, we therefore aim to develop a method to determine the optimal movie schedule for movie theatres.

The so-called Movie Scheduling Problem consists of two parts: session demand forecasting, and movie schedule optimization. According to our analysis, the expected demand for a session can best be modeled using a Gradient Tree Boosting model. The movie schedule can be represented as a network, and the problem is formulated as a Set Partitioning Problem. A comparative analysis is conducted between Column Generation and an Adaptive Large Neighborhood Search heuristic to solve the problem. While the former is generally able to obtain better solutions, the latter has a much shorter running time. Finally, a method is presented wherein the session demand forecast and the schedule optimization are combined. We use an iterative procedure in which the forecast is updated upon the acceptance of a new schedule, aiming to capture their mutual interdependence.

Contents

1 Introduction

In 1895 the Lumière brothers showed a short film in the first public screening (National Science and Media Museum, n.d.). Now, almost 130 years later, there are approximately 208,000 cinema screens around the world according to Motion Picture Association (2022). After a decline in the global box office market in 2020 due to the COVID-19 pandemic lockdowns, the market started increasing again in 2021 resulting in a total revenue of 21.3 billion USD.

The multiplex cinema is a very common type of movie theatre nowadays. Although there is no agreement on the exact definition of a multiplex cinema, it is generally described as a cinema that has multiple screening rooms (Hanson, 2019). The strength of this type of cinema is that it offers a wide range of movies in different genres and languages to serve a broad audience.

However, the rise of this cinema type has also introduced some challenges. The great number of different movies, screening room capacities and screen types can result in a large and complex schedule. Agreements with film distributors and management requests impose additional requirements on the schedule. Next to that, the movie exhibition business has recently undergone substantial changes, resulting in changes in the requirements of the schedule (M. Groen, personal communication, May 12, 2023). This thesis therefore aims to develop a method to find the optimal movie schedule for movie theatres. Optimality can be determined by identifying the key performance indicators of a movie theatre. To find an optimal schedule, we will also need to determine how to forecast ticket demand for both already showing movies, as well as new movies.

We use data from a large movie theatre company located in the Middle East. Ticket transaction data and schedule information from one of their locations form the basis of our case study. These data are especially relevant as the Middle East is expected to become one of the leading markets for global box office revenues, signified by the expected compound annual growth rate of the movies and entertainment market of 8.5% from 2021 to 2028 (Grand View Research, 2021). We complement this dataset with information on logistical and managerial requirements from other companies in the movie exhibition business, and movie characteristics retrieved from Internet Movie Database (IMDb).

The Movie Scheduling Problem can be represented by a network with multiple layers. To forecast the session demand, we formulate a both a linear regression model, as well as a Gradient Tree Boosting (GTB) model, and compare their performances. Given the forecast, we can formulate the problem as a Set Partitioning Problem. To find the optimal schedule, we compare the performance of Column Generation with the performance of an Adaptive Large Neighborhood Search (ALNS) heuristic. The session demand forecasting and schedule optimization are thereafter combined in an iterative procedure, where the forecast is updated when a new schedule is accepted.

We find that our GTB is best able to predict the demand for a session in the movie schedule. Moreover, Column Generation is able to find slightly better solutions than our ALNS heuristic, however at the expense of a much larger running time. The result of the combination of our forecasting and scheduling approach suggests to capture some interactions between the session demand and movie schedule, but the method struggles to find a good solution where all schedule requirements are satisfied.

This thesis firstly extends the existing literature on the Movie Scheduling Problem by comparing the performance of methods trying to find an optimal schedule. Moreover, to the best of our knowledge, we are the first to make the ticket demand forecast dependent on other showings in the current schedule. As this problem generalizes to more stochastic scheduling problems, our methods also contribute to other scheduling problems in the field of Operations Research. Next to that, this thesis helps to improve the operations of multiplex cinemas. Not only the movie theatre owner benefits from these improved performances, but also the customer as the schedule better satisfies their needs and preferences.

This thesis is part of an internship at data consultancy Lynxx. This company is mainly active in the analysis of data and optimization, and next to that provides data insights using visualizations.

The structure of this thesis is as follows. Chapter 2 presents the existing literature related to this thesis. Next, Chapter 3 discusses the problem and its requirements. The mathematical formulation and methods to solve the problem are presented in Chapter 4. The case study is introduced in Chapter 5 and the results are thereafter analyzed in Chapter 6. Chapter 7 discusses the outcomes, and finally Chapter 8 gives the limitations of this thesis and concludes.

2 Literature Review

This section will outline the existing literature on the Movie Scheduling Problem (Section 2.1), session demand forecasting (Section 2.2), other relevant Scheduling Problems (Section 2.3), and the integration of forecasting and optimization (Section 2.4). Finally, Section 2.5 will give the concluding remarks for the literature review.

2.1 Movie Scheduling Problem

Swami et al. (1999) were the first to formulate a model to create an optimal schedule for the movie screening in multiplex cinemas. They distinguish between two stages, the first being the selection of movies to include in the schedule, and the second being the creation of the schedule. Similarly, Iniestra et al. (2006) try to forecast movie demand and allocate movies to screens. They construct an Integer Programming model to create the weekly schedule. Note that these papers assume that each week only one movie can be shown at each screen, which was only the case before the digital film projector was introduced (see Appendix A for more details). Research thereafter continues to improve this approach and relaxes this assumption.

Similar to our approach, Eliashberg et al. (2009) formulate the Movie Scheduling Problem as a network. They subsequently model the problem as a Set Partitioning Problem, which they solve using Column Generation combined with Lagrangian Relaxation, and using an all-pair Shortest Path Problem as pricing problem. Next to that, they forecast the demand for a movie at a given time using Ordinary Least Squares. A different model is used for movies that have already been running than for new movies. Their forecast however does not take into account the effect of other movies that are shown during that time period. Note also that in this model movie theatres still try to limit the movie switching on one screen.

Table 1 summarizes the discussed papers considering the Movie Scheduling Problem.

2.2 Session Demand Forecasting

To create the movie schedule, the expected movie demand at a given time must be determined. Baranowski et al. (2020) forecast cinema attendance using several Linear Regression models. They show that cinema and movie specific variables in the model improve its predictive performance. Next, Lawitsanon et al. (2022) also formulate a predictive model for a Movie Scheduling Problem. They test the performance of Linear Regression models, Regression Trees and Neural Networks, and

Table 1. Overview literature Movie Scheduling Problem

Paper	**Forecasting Method**	**Scheduling Method**	**Output**
Swami et al. (1999)	Exponential function to forecast total gross box office sale	BILP exact solution	Weekly schedule with one movie per screen
Iniestra et al. (2006)	Ordinary Least Squares	BILP exact solution	Weekly schedule with one movie per screen
Eliashberg et al. (2009)	Ordinary Least Squares	Set Partitioning Problem with Column Generation and Lagrangian Relaxation	Daily schedule with preferably one movie per screen

conclude that the Neural Network has the best overall performance. Lee et al. (2018) also work with Machine Learning models. They construct a Cinema Ensemble Model by using seven different Machine Learning algorithms as candidate models, and evaluate their performance to select the best one for the specific prediction. They find that Gradient Tree Boosting most often has the most accurate results. Likewise, Leem et al. (2023) use a Gradient Boosting algorithm. However, next to that, they use k-means clustering to group movies with similar characteristics.

Different from this, Bardadym (1996) further investigates demand for movies by examining their life cycle. The author makes use of the Bass model (Bass, 1969), that is often used in forecasts of new products. More specifically, this model includes an innovation and imitation effect to forecast how accumulated movie demand is distributed over time. The paper highlights that sharing positive movie experiences has the largest impact on movie attendance in the early stage. Based on the Bass model, Zhang et al. (2017) construct a model with different seasonal dummies in an alternative seasonal structure. Their model allows for different seasonal effects of weekends and holidays and their distinct intertemporal demand shift pattern. The empirical results do show an improved performance of this model compared to the traditional Bass model. Tang and Dong (2021) also proposes a new method for forecasts of new short life-cycle products. Moreover, they apply their method on the box office market and therefore also show quantification of relevant variables. Their method, called Multi-Evidence Dynamic Weighted Combination Forecasting, has shown to perform better than other classical classification methods.

Table 2 summarizes the papers with session demand forecasting methods.

Table 2. Overview Session Demand Forecasting

Paper	**Forecasting method**
Baranowski et al. (2020)	Several Linear Regression models
Lawitsanon et al. (2022)	Linear Regression models, Regression Trees, **Neural Network**
Lee et al. (2018)	Ensemble model (Adaptive Tree Boosting, **Gradient Tree Boosting**, Linear Discriminant, Logistic Regression, Neural Network, Random Forest)
Leem et al. (2023)	Gradient Tree Boosting with k-means clustering
Bardadym (1996)	Bass model
Zhang et al. (2017)	Bass model with seasonal dummies
Tang and Dong (2021)	Multi-Evidence Dynamic Weighted Combination Forecasting

Note. For papers with multiple solving techniques, the method in bold is the best performing method.

2.3 Scheduling Problems

The Movie Scheduling Problem is a type of problem included in the large set of Scheduling Problems. As suggested by Swami et al. (1999), an analogy with our problem and the parallel Machine Scheduling Problem can be made. In this analogy, movies are represented by jobs, and screens can be seen as non-identical machines. Then, the problem is to decide when to schedule jobs, how often to schedule jobs of the same type, and on which machine. However, an important difference between this problem and ours is that for the Machine Scheduling Problem the ordering of the jobs only matters for their completion time, whereas in our problem the time interval that a machine is working on a job is of importance. Moreover, a substantial addition of our problem to the Machine Scheduling Problem is the uncertainty in revenue of the proposed schedule.

Another problem that could be compared to ours is the Classroom Scheduling Problem, which is proven to be $\mathcal{NP}$-hard (Bardadym, 1996). This is the problem of assigning lectures (movies) to rooms (screens) at a given time to build a schedule. The most important distinction is however that the lectures that need to be scheduled are fixed and their characteristics, such as group size, do not depend on the schedule. This problem can be modeled as a Binary Integer Linear Programming (BILP) model (Samiuddin & Haq, 2019). The problem in this paper is optimized in two stages using the simplex method, where the first stage schedules one type of classroom, and the second stage the other type. Yazdani et al. (2017) model the problem similarly, but thereafter investigates three solution algorithms. Out of Simulated Annealing, Genetic Algorithm and Artificial Immune Algorithm, the third performs best. Y. Chen et al. (2022) recently published a paper on the Classroom Scheduling Problem where student and teachers' preferences are considered in the model. In our problem we could similarly include customer and movie screening preferences to optimize the performance of a movie schedule. Similar to us, R. M. Chen and Shih (2013) also make the distinction between hard and soft constraints in the Course Timetabling Problem to include student and teacher preferences. They model the soft constraints by including a penalty in the objective function.

Finally, a relevant problem for us to examine is the Crew Scheduling Problem. This problem considers the scheduling of crew members (movies) to tasks (screens) at a specific time. Different from the Movie Scheduling Problem is however that all tasks must always have an assigned crew member during the entire time horizon. Moreover, the cost or revenue of assigning a crew member to a specific tasks needs to be determined differently than in the Movie Scheduling Problem. The problem is generally modelled as a Set Partitioning Problem (Wen et al., 2021). However, there exists a large number of feasible schedules for each crew member, and therefore solving techniques and heuristics are widely discussed in the literature. Importantly, Lavoie et al. (1988) was the first to introduce Column Generation on this specific problem, which is now a commonly used approach.

Table 3 summarizes the discussed papers in this section.

Table 3. Overview Scheduling Problems

Problem	**Paper**	**Solving technique**	**Constraints**
Classroom Scheduling	Samiuddin and Haq (2019)	Simplex Method	Hard
	Yazdani et al. (2017)	Simulated Annealing, Genetic Algorithm, **Artificial Immune Algorithm**	Hard
	Y. Chen et al. (2022)	Genetic Algorithm	Hard and soft
	R. M. Chen and Shih (2013)	Partical Swarm Optimization with Local Search	Hard and soft
Crew Scheduling	Lavoie et al. (1988)	Set Partitioning Problem with Column Generation	Hard

Note. For papers with multiple solving techniques, the method in bold is the best performing method.

2.4 Integrated Forecasting and Optimization

As highlighted before, the Movie Scheduling Problem consist of two parts: session demand forecasting and schedule optimization. Ding and Niu (2013) also try to optimize a schedule that uses forecasting input. They use forecasted parameters in their Mixed Integer Quadratic Constraint Programming Problem, and thus model the problem in two stages. Likewise, Zhou et al. (2022) apply the predict-then-optimize method to the reallocation in the bike-sharing network. They employ Distributionally Robust Optimization to predict the expected reallocation, and a branch-and-price algorithm is proposed to solve the reallocation problem. Different from our problem is that these forecasts only depend on external factors, and not on factors that are included in the optimization problem.

Carriere and Kariniotakis (2019) show an approach to simultaneously consider a forecasting model and a decision-making strategy. Their problems in the renewable energy market use forecast models as input into scheduling, reserves estimation and trading strategy decisions. Instead of optimizing the forecasting and decision-making steps separately, they use the Extreme Learning Machine variant of the Artificial Neural Network to solve the problems simultaneously. Similarly, Stratigakos et al. (2022) examine integrated forecasting and optimization in the renewable energy industry. They use Prescriptive Decision Trees where decisions are formulated using a Weighted Sample Average Approximation. Thus, the locally optimal split is determined for each tree node by directly optimizing based on the contextual information.

We also examine Bilevel Programming frameworks, or more generally Hierarchical Programming frameworks, where one optimization problem depends on the solutions obtained at the lower levels. Muñoz et al. (2022) provide a framework for solving Bilevel Optimization Problems, where also contextual information can be included in the models. In our problem there is however not a clear hierarchical structure when we assume that the forecasted demand for a session also depends on the rest of the schedule. Namely, it is the case that the optimal schedule depends on the optimal forecast and vice versa.

Table 4 summarizes the papers discussed in this section.

Table 4. Overview Integrated Forecasting and Optimization

Paper	**Method**	**Application**
Ding and Niu (2013)	Predict-then-optimize	Seawater Reverse Osmosis System
Zhou et al. (2022)	Predict-then-optimize	Bike sharing network
Carriere and Kariniotakis (2019)	Simultaneous optimization with Neural Networks	Renewable energy trading
Stratigakos et al. (2022)	Simultaneous optimization with Prescriptive Decision Trees with Weighed Sample Average Approximation	Renewable energy industry
Muñoz et al. (2022)	Bilevel Programming	Electricity industry

2.5 Conclusion of the Literature Review

We highlight the most important papers from the literature review to conclude this chapter. The paper by Eliashberg et al. (2009) serves as the starting point for this thesis. We use a similar network formulation and modeling approach to represent and solve the problem, whilst enhancing the problem requirements and specifications. To extend the problem formulation, we follow the approach of R. M. Chen and Shih (2013) to incorporate soft constraints in scheduling problems. Moreover, for predicting session demand, we make use of the findings of Lee et al. (2018) and Leem et al. (2023), which indicate that a Gradient Tree Boosting model often yields accurate forecasts in the movie demand context. Finally, inspiration for an approach to combine the session demand forecast and schedule optimization is mostly taken from the predict-then-optimize method outlined by Ding and Niu (2013) and Zhou et al. (2022).

3 Problem Description

Before examining the problem, we will first shortly discuss some more general insights in the movie exhibition business in Section 3.1. Additional background information about the sector is presented in Appendix A. Secondly, we will define the Movie Scheduling Problem in Section 3.2. Finally, the aspects relevant to forecast demand for a session will be outlined in Section 3.3. Note that information and the problem definition in this chapter is, among other things, based on a personal interview with an experienced professional in the movie exhibition business (M. Groen, personal communication, May 12, 2023).

3.1 Sector Information

Multiplex cinemas typically have a number of rooms of different sizes and screen types. Moreover, rooms can have different seat types to allow for distinct experiences. Note that each room and seat type can have different ticket prices. An important party in the movie exhibition business is the film distributors. Film distributors control the process of making a movie available to view by the audience after the production process has finished. This includes the marketing and distribution of the movie. Film distributors also form agreements with multiplex cinemas to show a movie in their cinema. In this agreement, the share of box office sales that the distributor receives is stated, and also some requirements can be specified which the multiplex cinema needs to satisfy.

3.2 Movie Scheduling

The Movie Scheduling Problem is the problem of assigning movies to screens at a specific time. The weekly schedule is released three days ahead of time and covers the period from Thursday to Wednesday. The list of movies that need to be scheduled each week is given. The goal is to create a schedule that yields an expected box office revenue that is as high as possible. To find an optimal schedule, a number of restrictions and requirements need to be taken into account. These restrictions can be split into three sets: logistical, distributor, and managerial constraints.

The first set of restrictions of the problem we call the logistical constraints. Firstly, the number of tickets sold of a specific seat type cannot exceed the number of seats of that type in the room. Second, there is the restriction that the room for the next showing can only open after the previous movie in that room has finished, all the guests have left the room, and the room is cleaned. Third, it is required that only a given number of movies can start during a given time interval. The same holds for the number of movies ending. Next to this, within every consecutive set of a given number of time periods, the total capacity of movie rooms starting or ending screenings during this interval cannot exceed a given number. This is to prevent these areas from getting overcrowded and also to make sure that enough personnel is available to for example scan tickets or prepare the room for the next showing.

The next set of restrictions we call the distributor constraints. These constraints arise from agreements between film distributors and multiplex cinema operators. The constraints differ significantly between movies and distributors, but generally have become less restricting after the COVID-19 pandemic. The requirements can be of the following form. The first requirement can be on a weekly basis, specifying a minimum number of showings in the evening or afternoon per week. Second, distributors can require to use a room exclusively for their movie in that week. Finally, an agreement on the minimum number of daily showings can be in place for movies.

The final set of restrictions we call the managerial constraints. These requirements are set by the management of the multiplex cinema, mostly to achieve a certain level of service. Note that this set of constraints should be modeled as soft constraints. The first of such restrictions specifies the time the first movie can start and the time the last movie must finish, since opening times of cinemas can be flexible. Second, movies in a given number of different languages or with different genres must start in a specified time interval to offer a wide range of movies. Third, movies should be shown on the same set of screens as much as possible. This is to limit unnecessary confusion and complication for visitors and staff. Finally, it is sometimes desired that at least one movie starts in every specified time interval.

3.3 Session Demand Forecasting

To make the schedule, we also need to make a forecast on the number of people to attend a movie at a given time. The number of ticket sales are dependent on a large number of variables. Firstly, some general information is important in the time specific forecast. Ticket sales are dependent on the moment of the screening, such as time of day, day of the week, public holidays, or other large events occurring. Secondly, the weather can be affecting the number of ticket sales. Moreover, characteristics of the movie influences tickets sold. Currently, target audience is mostly used in the demand forecasting of a movie. The important characteristics language, genre, sequel information, release date, cast, and directors of the movie together determine the target audience, and we therefore consider these characteristics separately. Other suitable characteristics are the movie budget, and the marketing plan and budget. Note that the demand forecast is currently also largely based on identifying equivalent movies and examining their past performance. If a movie is already showing, the rating is also relevant in the prediction. There are two types of ratings: those provided by movie professionals and those given by the audience. The rating provided by the audience indicates the extent to which a positive movie experience is shared. Furthermore, for already showing movies, historic ticket sales data can be used to improve the forecast. Finally, other sessions in the schedule can also affect the expected session demand. Namely, if other popular movies, movies in the same genre, or movies with the same release week show at the same time, the demand per session might decrease because customers now need to decide what movie or session to attend.

4 Methods

This chapter will present the methods to solve the different stages of the Movie Scheduling Problem. First, Section 4.1 shows how to model the movie demand at a given time. Next, Section 4.2 will introduce the approaches to optimize the movie schedule given the session demand forecast. Finally, Section 4.3 presents the approach to combine the scheduling and forecasting methods.

4.1 Forecasting Model

To model demand for movie m at time t, we first introduce a set of general variables that can affect the accumulative demand for the sessions. These variables are the first listed in Table 5. Next, we also introduce a set of movie specific variables. Finally, we also allow for interaction between sessions. For this, we need information of the current schedule. These variables are presented last in Table 5.

Table 5. Independent Variables

General Variables	
I_m	indicator for movie m
I_h	indicator for starting time in hour h
I_ω	indicator for showing day ω
$HOLI$	indicator for (public) holiday
$WEATHER$	indicator for favourable weather for movie attendance
Movie Specific Variables	
$METER$	IMDb MOVIEmeter position
$RATING$	IMDb audience rating [1]
$LANG_l$	indicator for language l
$GENRE_g$	indicator for genre g
$SEQUEL$	indicator whether movie is a sequel
$ACTOR_{100}$	indicator whether one of the maximum first three listed actors are in IMDb top 100
$ACTOR_{1000}$	indicator whether one of the maximum first three listed actors are in IMDb top 1000

[1] To measure audience experience.

$DIRECT_{1000}$	indicator whether one of the maximum first three listed directors are in IMDb top 1000
$DIRECT_{5000}$	indicator whether one of the maximum first three listed directors are in IMDb top 5000
$DISTR$	indicator for IMDb ranking of main distributor above 500
$PRODUCT$	indicator for movie production budget above 80,000,000 USD
$RELEASE$	number of weeks after release date
$TICKETS$	number of tickets sold last scheduled week (Thursday to Sunday)
Movie Schedule Variables	
$COUNT_g$	number of movies in the same genre starting maximum one hour before or after this movie
$COUNT_{release}$	number of movies with same release week starting maximum one hour before or after this movie
$COUNT_{pop}$	number of movies with top 5 MOVIEmeter position of the movies showing that week starting maximum one hour before or after this movie

We use Supervised Learning since we have a target variable, namely session demand, and predictors, namely the independent variables listed in Table 5. Note that the movie schedule variables are only included in the model when an initial schedule is known. We determine the demand for a specific movie on an hourly basis. Two different models are introduced, and their performance will be compared.

4.1.1 Linear Regression

We start with predicting the demand for sessions in a naive way with a linear regression model, estimated by Ordinary Least Squares (OLS). We will test the data for skewness by determining the Fisher-Pearson coefficient of skewness, and apply a logistic transformation of the dependent variable if needed (Changyong et al., 2014). If a logistic transformation is applied, the model for demand,

d_{mt}, of movie m showing at time t is given by the equation below.

$$\begin{aligned}\log d_{mt} = \quad & \alpha + \sum_{m \in M} \beta_m \cdot I_m + \sum_{\forall h} \beta_h \cdot I_h + \sum_{\omega \in W} \beta_\omega \cdot I_\omega + \beta_{HOLI} \cdot HOLI + \beta_{WEATHER} \cdot WEATHER \\ & + \beta_{METER} \cdot METER + \beta_{RATING} \cdot RATING + \sum_{l \in L} \beta_l \cdot LANG_l + \sum_{g \in \mathcal{G}} \beta_g \cdot GENRE_g \\ & + \beta_{SEQUEL} \cdot SEQUEL + \beta_{ACTOR_{100}} \cdot ACTOR_{100} + \beta_{ACTOR_{1000}} \cdot ACTOR_{1000} \\ & + \beta_{DIRECT_{1000}} \cdot DIRECT_{1000} + \beta_{DIRECT_{5000}} \cdot DIRECT_{5000} + \beta_{DISTR} \cdot DISTR \\ & + \beta_{PRODUCT} \cdot PRODUCT + \beta_{RELEASE} \cdot RELEASE + \beta_{TICKETS} \cdot TICKETS \\ & + \beta_{COUNT_g} \cdot COUNT_g + \beta_{COUNT_r} \cdot COUNT_{release} + \beta_{COUNT_{pop}} \cdot COUNT_{pop} + \varepsilon_{mt}\end{aligned}$$

Note that we assume $\varepsilon_{mt} \sim N(0, \sigma^2)$ such that the coefficients can be estimated by OLS.

4.1.2 Gradient Tree Boosting

We also model the session demand using Gradient Tree Boosting (GTB). GTB is a machine learning method that improves the performance of decision trees sequentially by combining weak classifiers. In our case, since the target variable is numerical, we use regression trees. The large number of independent variables, between which some interaction might be present, makes this method a good fit for our data. Namely, as the weights of the predictors are updated using the gradient of the loss function, the relevance of each independent variable is determined and used in the final prediction. Moreover, this ensures some robustness against irrelevant independent variables, limiting the degree of overfitting. We use the commonly used Mean Squared Error as a loss function. Moreover, all data points are used in the individual trees, since we want to limit the bias.

To correctly train the model and limit overfitting, we tune some hyperparameters. We need to tune both boosting parameters, as well as tree-specific parameters. Firstly, we tune the learning rate, which regulates the contribution of each tree in the final model. A high learning rate can lead to overfitting of the model, whereas a lower learning rate slows down the training of the model. Secondly, we tune the number of trees. A too high number of trees can again result in overfitting. Next, we also need to do some tuning of the tree-specific parameters. We tune the parameter for the minimum requirement on the number of samples to split a node. Furthermore, the maximum depth of a tree need to be specified. A higher depth makes the model more likely to overfit the data. Finally, we can also tune the maximum number of features considered at each split. This value should be around the squared root of the total number of features.

4.2 Scheduling Model

To formulate the problem, we first describe the main sets. Firstly, $\mathcal{T}$ contains the set of time periods t, ranging from 1 to T. The subset $\mathcal{T}^m$ contains the time periods in which movie m is allowed to start. The subset $\hat{\mathcal{T}}^m$ contains the time periods in which it is not desired, but allowed, to start movie m. Secondly, the set of movies is expressed by M. Movie m, $m \in M$, has a duration denoted by δ_m and a commercial time denoted by o_m. Moreover, the subset M^t contains the movies that can be shown during time period t and the subset M^s the movies that can be shown on screen s. Thirdly, the set of screens is expressed by S. Screen s, $s \in S$, has a capacity of each seat type denoted by c_{sk}, where k, $k \in K^s$, denotes the type of seat from a set of seat types of screen s. Moreover, it has a cleaning duration p_s. Subset S^m contains the screens on which movie m can be shown. Furthermore, subset S^e contains the screens with cinema type e.

To be able to introduce specific restrictions to the problem, we also introduce the following movie characteristics and sets. For each existing movie genre, which we denote by g ($g \in \mathcal{G}$), we create a subset of movies $M^g \subseteq M$ that contains all movies in that genre. At least movies with $\hat{g}$ different genres must preferably be shown in time interval τ_g. Moreover, for each language l that a movie is showed in ($l \in L$), we create a subset $M^l \subseteq M$ that contains all movies in that language. At least movies displayed in $\hat{l}$ different languages must preferably be shown in time interval τ_l. Moreover, we define the minimum amount of showings of movie m in the time periods in $\mathcal{T}^m_{req}$ as η_m.

Furthermore, the maximum amount of movies that can start in time interval τ_{start} is denoted by γ_{start}. Similarly, the maximum amount of movies that can end in time interval τ_{end} is denoted by γ_{end}. Moreover, we can group screens in area r in subset $S^r \subseteq S$. Then, for each time interval $[t, t+1]$, the total capacity of movie rooms in S^r ending a screening at time t and the total capacity of movie rooms in S^r starting screenings at time $t+1$ cannot exceed f_{rt}. Next to that, the minimum number of daily showings of movie m on cinema type e is denoted by $\underline{s}_{me}$. Similarly, the maximum number of daily showings of movie m on cinema type $e \in E$ is denoted by $\bar{s}_{me}$ Finally, at least one movie must start in time interval $\hat{\tau}$.

An overview with all sets and parameters that are discussed above can be found in Table B1.

4.2.1 Network Formulation

The problem can be formulated as a network. We denote the complete acyclic directed graph by $G = (V, A)$, where V denotes the set of nodes, and A the set of arcs. The graph contains a source node v_s and a sink node v_t. All other nodes v, $v \in V \setminus \{v_s, v_t\}$, correspond to a time period t, $t \in \mathcal{T}$, a screen s, $s \in S$, and a movie m, $m \in M$, which can be denoted by $(t,(s,m))$. The graph G can be divided into subgraphs G^s, $G^s = (V^s, A^s), \forall s \in S$, where each subgraph corresponds to a screen s. The subset $V^s \subset V$ contains only the nodes corresponding to movies m that can show on screen s ($m \in M^s$), and corresponding to time periods t that are allowed for the movie m ($t \in T^m$). Also note that there are no arcs between subgraphs. Consequently, each subgraph can be interpreted as a layer of the network that is only connected to the source node v_s and the sink node v_t. The arcs in the graph are weighted, such that the weight of the arc is defined as the negative weighted sum of the minimum of the capacity of the screen and the expected demand for the movie of each seat type of the screen, all of the destination node. If we denote the demand for the movie m starting at time period t by d_{mt}, then we can write the weight of arc $((t_1,(s,m_1)),(t_2,(s,m_2)))$ as $w_{((t_1,(s,m_1)),(t_2,(s,m_2)))} = \sum_{k \in K^s} \rho_k \min\{c_{sk}, d_{m_2 t_2 k}\}$, with ρ_k the weight of seat type k. Arcs exist between the source node v_s and all nodes that exist in the layer of the network. Note that also an arc exists between the source node v_s and the sink node v_t indicating no showings on that screen. Moreover, arcs exist between all nodes in subgraph and the sink node v_t. Finally, arcs $((t,(s,m_1)),(t+o_{m_1}+\delta_{m_1}+p_s+x,(s,m_2))), \forall x$ s.t. $t \in \mathcal{T}^{m_1}, t+o_{m_1}+\delta_{m_1}+p_s+x \in \mathcal{T}^{m_2}, \forall s \in S, \forall m_1, m_2 \in M^s$, exist in the subgraph G^s.

The maximum size of the network can be expressed by the maximum number of nodes in the graph. The maximum number of nodes is $|V| = |\mathcal{T}| * |S| * |M| + 2$, and is obtained if every movie can possibly be shown at every screen and starting each time period.

4.2.2 Set Partitioning Problem

We model the Movie Scheduling Problem as a Set Partitioning Problem, such that we select a schedule for every screen. A path represents a partial schedule of a screen, and a path from the source node v_s to the sink node v_t through subgraph G^s is a complete schedule for screen s. The weight of path p is defined as the sum of the weights of the selected arcs, denoted by w_p. We then try to select the paths with minimum costs, such that one complete path is selected for each screen and all other requirements are satisfied. Let decision variable x_p^s be one if path p is selected for screen s, and zero otherwise. Then the problem can be formulated as follows.

$$\min \sum_{s \in S} \sum_{p \in P^s} w_p x_p^s \tag{1}$$

$$\text{s.t.} \sum_{p \in P^s} x_p^s = 1, \qquad \forall s \in S \tag{2}$$

$$\sum_{s\in S}\sum_{p\in P^s}\sum_{m\in M^s}\sum_{j=t}^{t+\tau_{start}-1} b^s_{jmp}x^s_p \le \gamma_{start}, \qquad \forall t\in\mathcal{T}\setminus\{T-\tau_{start}+2,\dots,T\} \tag{3}$$

$$\sum_{s\in S}\sum_{p\in P^s}\sum_{m\in M^s}\sum_{j=t}^{t+\tau_{end}-1} b^s_{(j-\delta_m-o_m)mp}x^s_p \le \gamma_{end}, \qquad \forall t\in\mathcal{T}\setminus\{1,\dots,\max(\delta_m+o_m)\} \tag{4}$$

$$\sum_{s\in S^r}\sum_{p\in P^s}\sum_{m\in M^s}\sum_{k\in K^s} (b^s_{(t-\delta_m-o_m)mp}+b^s_{(t+1)mp})c_{sk}x^s_p \le f_{rt}, \qquad \forall r\in R, \forall t\in\mathcal{T}\setminus\{1,\dots,\max(\delta_m+o_m),T\} \tag{5}$$

$$\sum_{s\in S}\sum_{p\in P^s}\sum_{t\in\mathcal{T}^m_{req}} b^s_{tmp}x^s_p \ge \eta_m \qquad \forall m\in M \tag{6}$$

$$\sum_{s\in S^e}\sum_{p\in P^s}\sum_{t\in\mathcal{T}_\omega} b^s_{tmp}x^s_p \ge \underline{s}_{me}, \qquad \forall m\in M, \forall e\in E, \forall \omega\in W \tag{7}$$

$$\sum_{s\in S^e}\sum_{p\in P^s}\sum_{t\in\mathcal{T}_\omega} b^s_{tmp}x^s_p \le \bar{s}_{me}, \qquad \forall m\in M, \forall e\in E, \forall \omega\in W \tag{8}$$

$$x^s_p\in\{0,1\}, \qquad \forall s\in S, p\in P^s \tag{9}$$

Here P^s is the set of all paths in G^s. Moreover, b^s_{tmp} is one if movie m is on path p of screen s with time period t, and zero otherwise. Note that the set $\mathcal{T}_\omega$ contains the time periods belonging to day ω. Constraints (2) ensure that exactly one path is selected for every screen. Next, constraints (3) and (4) restrict the number of movies that can start or end in the same time interval, respectively. The rooms that start and end a session within a time interval within an area of the movie theatre is restricted by a capacity limit of the rooms specified in constraints (5). Furthermore, constraints (6) ensure a minimum amount of showings for movies within a specified time interval. In line with this, constraints (7) ensure a minimum amount of daily showings for movies in rooms of a certain type. Similarly, constraints (8) ensure a maximum amount of daily showings for movies in rooms of a certain type. Finally, (9) are the domain restrictions for our decision variables.

Next we introduce a set of soft constraints. If a soft constraint is violated, a penalty is added to the objective value. We have the following soft constraints.

$$\sum_{s\in S}\sum_{p\in P^s}\sum_{m\in M^s}\sum_{t\in\hat{\mathcal{T}}^m} b^s_{tmp}x^s_p = \sigma \tag{10}$$

$$\sum_{s\in S}\sum_{p\in P^s}\sum_{m\in M^s}\sum_{j=t}^{t+\hat{\tau}-1} b^s_{jmp}x^s_p \ge 1-\beta_t, \qquad \forall t\in\mathcal{T}\setminus\{T-\hat{\tau}+2,\dots,T\} \tag{11}$$

$$\sum_{p\in P^s}\sum_{t\in\mathcal{T}^\omega}\frac{1}{\mathcal{M}}b^s_{tmp}x^s_p \le \zeta^\omega_{ms}, \qquad \forall\omega\in W,\forall m\in M,\forall s\in S^m \tag{12}$$

$$\sum_{m\in M^g}\sum_{s\in S}\sum_{p\in P^s}\sum_{j=t}^{t+\tau_g-1} b^s_{jmp}x^s_p \ge a_{gt}, \qquad \forall g\in\mathcal{G},\forall t\in\mathcal{T}\setminus\{T-\tau_g+2,\dots,T\} \tag{13}$$

$$\sum_{g\in\mathcal{G}} a_{gt} \ge \hat{g}-\beta^g_t, \qquad \forall t\in\mathcal{T}\setminus\{T-\tau_g+2,\dots,T\} \tag{14}$$

$$\sum_{m\in M^l}\sum_{s\in S}\sum_{p\in P^s}\sum_{j=t}^{t+\tau_l-1} b^s_{jmp}x^s_p \ge a_{lt}, \qquad \forall l\in L,\forall t\in\mathcal{T}\setminus\{T-\tau_l+2,\dots,T\} \tag{15}$$

$$\sum_{l\in L} a_{lt} \ge \hat{l}-\beta^l_t, \qquad \forall t\in\mathcal{T}\setminus\{T-\tau_l+2,\dots,T\} \tag{16}$$

$$\sigma\in\mathbb{N} \tag{17}$$

$$\beta_t\in\{0,1\}, \qquad \forall t\in\mathcal{T}\setminus\{T-\hat{\tau}+2,\dots,T\} \tag{18}$$

$$\zeta^\omega_{ms}\in\{0,1\}, \qquad \forall\omega\in W,\forall m\in M,\forall s\in S^m \tag{19}$$

$$a_{gt}\in\{0,1\}, \qquad \forall g\in\mathcal{G},\forall t\in\mathcal{T}\setminus\{T-\tau_g+2,\dots,T\} \tag{20}$$

$$\beta^g_t\in\mathbb{N}, \qquad \forall t\in\mathcal{T}\setminus\{T-\tau_g+2,\dots,T\} \tag{21}$$

$$a_{lt}\in\{0,1\}, \qquad \forall\in L,\forall t\in\mathcal{T}\setminus\{T-\tau_l+2,\dots,T\} \tag{22}$$

$$\beta^l_t\in\mathbb{N}, \qquad \forall t\in\mathcal{T}\setminus\{T-\tau_l+2,\dots,T\} \tag{23}$$

Note that $\mathcal{M}$ is a very large number. More specifically, for $\mathcal{M}$ we can divide the time in a day by the shortest duration of a movie, commercial time, and time required after a session. Constraint (10) count the number of times a movie starts in an undesired (but allowed) time period. Next, constraints (11) determine for every time period whether at least one movie starts, such that $\beta_t = 1$ if no movie starts in time interval $[t, t-\hat{\tau}+1]$, and β_t will be zero otherwise. The constraints (12) counts, using auxiliary variable ζ^ω_{ms}, for each movie how many different screens are used during each day. Furthermore, constraints (13) and (15) check if in a given time interval a movie is showing in a genre and language, respectively. Note that if a movie is showing in time interval $[t, t+\tau_g-1]$, then a_{gt} is one, and zero otherwise. This works similarly for languages. Then, (14) and (16) determine if the desired number of different genres and languages is reached, respectively. Finally, (17) to (23) specify the domains of the auxiliary variables.

Consequently, we can add the following penalty to the objective function:

$$\lambda_1\sigma+\lambda_2\sum_{t\in\mathcal{T}\setminus\{T-\hat{\tau}+2,\dots,T\}}\beta_t+\lambda_3\sum_{\omega\in W}\sum_{m\in M}\sum_{s\in S^m}\zeta^\omega_{ms}+\lambda_4\sum_{t\in\mathcal{T}\setminus\{T-\tau_g+2,\dots,T\}}\beta^g_t+\lambda_5\sum_{t\in\mathcal{T}\setminus\{T-\tau_l+2,\dots,T\}}\beta^l_t.$$

Here, λ_i, $i = 1, \dots, 5$, specifies the weight of the penalties.

4.2.3 Column Generation

The number of variables in our Set Partitioning Problem ((1)-(23)) is very large, and therefore we will apply Column Generation to solve the problem (Dantzig & Wolfe, 1960). We call the linear relaxation of the Set Partitioning Problem in which we only use a limited number of variables x_p^s, the Restricted Master Problem (RMP). To add variables to the RMP, we solve the pricing problem. The pricing problem in our case can be approached by solving the Shortest Path Problem, where we update the network with the dual variables. More specifically, we update the arc weights by summing the dual values corresponding to those arcs. The reduced cost of the path depends on the dual variables corresponding to the RMP. The reduced cost of path p on screen s is defined as follows.

$$
\begin{aligned}
RC_s^p = w_p - \mu_s^{(2)} + \sum_{t=1}^{T-\tau_{start}+1} \mu_{st}^{(3)} \sum_{m\in M^s} \sum_{j=t}^{t+\tau_{start}-1} b_{jmp}^s + \sum_{t=\max(\delta_m+o_m)+1}^{T} \mu_{st}^{(4)} \sum_{m\in M^s} \sum_{j=t}^{t+\tau_{end}-1} b_{(j-\delta_m-o_m)mp}^s \\
+ \sum_{r\in R} \sum_{t=\delta_m+o_m+1}^{T-1} \mu_{srt}^{(5)} \sum_{m\in M^s} \sum_{k\in K^s} c_{sk}(b_{(t-\delta_m-o_m)mp}^s + b_{(t+1)mp}^s) - \sum_{m\in M} \mu_{sm}^{(6)} \sum_{t\in\mathcal{T}_{req}^m} b_{tmp}^s \\
- \sum_{m\in M} \sum_{e\in E} \sum_{\omega\in W} \mu_{sme\omega}^{(7)} \sum_{t\in\mathcal{T}^\omega} b_{tmp}^s + \sum_{m\in M} \sum_{e\in E} \sum_{\omega\in W} \mu_{sme\omega}^{(8)} \sum_{t\in\mathcal{T}^\omega} b_{tmp}^s - \mu_s^{(10)} \sum_{m\in M^s} \sum_{t\in\hat{\mathcal{T}}^m} b_{tmp}^s \\
- \sum_{t=1}^{T-\hat{\tau}+1} \mu_{st}^{(11)} \sum_{m\in M^s} \sum_{j=t}^{t+\hat{\tau}-1} b_{jmp}^s + \sum_{\omega\in W} \sum_{m\in M} \mu_{s\omega m}^{(12)} \sum_{t\in\mathcal{T}^\omega} \frac{1}{\mathcal{M}} b_{tmp}^s \\
- \sum_{g\in\mathcal{G}} \sum_{t=1}^{T-\tau_g+1} \mu_{sgt}^{(13)} \sum_{m\in M^g} \sum_{j=t}^{t+\tau_g-1} b_{jmp}^s - \sum_{l\in L} \sum_{t=1}^{T-\tau_l+1} \mu_{slt}^{(15)} \sum_{m\in M^l} \sum_{j=t}^{t+\tau_l-1} b_{jmp}^s
\end{aligned}
$$

Note that the dual variables corresponding to constraint i are denoted by $\mu_s^{(i)}$ in the equation above. We find the shortest path and its weight for screen s efficiently by making use of the characteristics of a Directed Acyclic Graph. Namely, for such a graph a Topological Sorting exists and can be found in time $\mathcal{O}(|A| + |V|)$. Thereafter, we can apply a dynamic programming algorithm with time complexity $\mathcal{O}(|A|)$, such that the total time complexity of finding the shortest path is $\mathcal{O}(|A| + |V|)$. Note however that we want to find k-shortest paths for each screen to explore possible path options. We therefore do not only need to keep track of the shortest path, but also explore and recall non-optimal paths, which increases the time complexity. We approximate the k-shortest paths by not remembering all possible paths, but dropping possible paths if their solution is worse than $k + p$

other paths.

To initialize the RMP we solve the Shortest Path Problem of the initial network for each screen iteratively. After finding a shortest path, we update the network such that paths resulting in infeasible combinations cannot be made.

We add the paths generated by the k-Shortest Path Problem and with a negative reduced cost to RMP. Thereafter, we repeat the steps of resolving the RMP and finding paths with negative reduced costs. The Column Generation procedure is stopped when n columns with negative reduced cost are added, or when no more columns with negative reduced cost can be found by our pricing heuristic.

After we terminate the Column Generation procedure, we need to construct a feasible solution. We can do so by using a MIP solver to solve problem (1)-(23) exactly with the generated columns. This solution will be an approximation of the optimal solution.

4.2.4 Adaptive Large Neighborhood Search

In this section we introduce an Adaptive Large Neighborhood Search (ALNS) heuristic to solve the problem. This approach solves the problem iteratively, and we present the pseudo-code of the algorithm in Algorithm 1. The initial solution, s_0, can be found similarly as in Column Generation (Section 4.2.3), or the solution after performing Column Generation can be used. To start the algorithm, also the weights of the destroy, repair and improvement operators are initialized. These weights are updated during the algorithm by taking into account their performance. Every iteration, a repair, destroy and improvement operator is selected and used to destroy and thereafter repair and improve the current solution. The new solution, s', is accepted to use in the next iteration if it has a better objective than the current solution, or if the objective is accepted by the criteria of simulated annealing. This criteria is derived from Kirkpatrick et al. (1983). It states that a new solution s', that is not an improvement on the current solution s, is still accepted as the new solution with probability $e^{\frac{f(s)-f(s')}{temp_{iter}}}$. Here, $f(s)$ and $f(s')$ denote the objective of solution s and s' respectively. Moreover, $temp_{iter}$ denotes the temperature of the iteration. This is determined by the formula $temp_{iter} = \alpha \cdot temp_{iter-1}$, where α is the parameter for the cooling rate. The temperature is initialized at $temp_0 = \frac{z(s_0)}{\ln(2)}$. Moreover, every θ iterations we try to improve the current solution by inserting feasible sessions into the current schedule that improve the objective function. Note that we start with inserting the session with the best contribution to the objective function, and continue until no more sessions that improve the schedule can be found. The process is stopped when a predefined number of iterations χ_1 has occurred, or when a predefined number of non-improving solutions χ_2 is reached. The best found solution is thereafter improved by removing sessions from the schedule that worsen the objective. Note that a session can only be removed if it is allowed to remove the session, such that the final solution is still feasible. Afterwards, the resulting solution is further improved by inserting feasible sessions into the current schedule that improve the objective function. A detailed explanation of the operators used in our ALNS algorithm can be found below.

Algorithm 1 ALNS algorithm

```
1: **Input:** Initial solution $s_0$
2: $s \leftarrow s_0$
3: $s_{best} \leftarrow s_0$
4: $\omega_h \leftarrow \frac{1}{|D|}, \forall h \in D$                    ▷ Initialize weight destroy operation
5: $\omega_h \leftarrow \frac{1}{|R|}, \forall h \in R$                    ▷ Initialize weight repair operation
6: $\omega_h \leftarrow \frac{1}{|I|}, \forall h \in I$                    ▷ Initialize weight improvement operation
7: $iter \leftarrow 1$
8: **while** Stopping criteria not satisfied **do**
9:     $s' \leftarrow s$
10:    **if** $iter \% \iota = 0$ **then**                                  ▷ Weight adjustment
11:        $w_D \leftarrow adjust(w_D)$
12:        $w_R \leftarrow adjust(w_R)$
13:        $w_I \leftarrow adjust(w_I)$
14:    **end if**
15:    **if** $iter \% \theta = 0$ **then**                                 ▷ Improve current solution
16:        $s' \leftarrow AddImprovingSessions(s')$
17:    **end if**
18:    $h_{destroy} \leftarrow select(\omega_D)$                           ▷ Select destroy operation
19:    $h_{repair} \leftarrow select(\omega_R)$                            ▷ Select repair operation
20:    $h_{improvement} \leftarrow select(\omega_I)$                       ▷ Select improvement operation
21:    $s' \leftarrow h_{improvement}(h_{repair}(h_{destroy}(s')))$        ▷ Perform operations
22:    **if** $f(s') \leq f(s_{best})$ **then**                            ▷ Set new best solution
23:        $s_{best} \leftarrow s'$
24:    **end if**
25:    **if** $accept(s', s)$ **then**                                     ▷ Set new solution for next iteration
26:        $s \leftarrow s'$
27:    **end if**
28:    $iter \leftarrow iter + 1$
29: **end while**
30: $s_{best} \leftarrow RemoveWorseningSessions(s_{best})$  ▷ Remove sessions that worsen schedule if allowed
31: $s_{best} \leftarrow AddImprovingSessions(s_{best})$          ▷ Add feasible sessions that improve schedule
32: **Output:** $s_{best}$
```

Destroy Operators

To break the current solution s, destroy operators are used. These operators destroy a solution by removing κ showings. We select the following destroy operators to use.

1. **Random Showing Removal:** This operator randomly selects κ showings and removes them from the solution.

2. **Random Screen Removal:** This operator randomly selects a screen and removes all showings on that screen. This process continues until κ showings are removed from the solution.

3. **Worst Removal:** This operator removes showings with the κ lowest contributions to the objective function.

Repair Operators

After breaking the current solution s, showings might need to be added to create a feasible solution. The following repair operators are used.

1. **Random Showing Insertion:** This operator randomly selects a movie or screen for which the minimum showing requirements (constraint (2), (6) and (7)) are not met. It then randomly inserts a screening for this movie or screen at a feasible position. This process is repeated until all minimum showing requirements are satisfied and the solution is feasible.

2. **Best Showing Insertion:** This operator again randomly selects a movie or screen for which the minimum showing requirements (constraint (2), (6) and (7)) are not met. Then, a screening for this movie or screen is inserted at a feasible position that yields the best improvement in the objective function. This process is repeated until all minimum showing requirements are satisfied and the solution is feasible.

3. **Worst Insertion:** This operator again randomly selects a movie or screen for which the minimum showing requirements (constraint (2), (6) and (7)) are not met. Then, a screening for this movie or screen is inserted at a feasible position that yields the worst improvement in the objective function. This process is repeated until all minimum showing requirements are satisfied and the solution is feasible.

Improvement Operators

Next to repairing the solution to a feasible one, also showings might be included in the schedule to improve the objective function. We therefore use the following insertion operators.

1. **Poor Screen Insertion:** This operator selects the screen that currently has the worst contribution to the objective function. If possible, it inserts the screening on this screen at a feasible position that yields the best improvement in the objective function. This process is repeated ψ times.

2. **Best Movie Insertion:** This operator selects the movie that currently has the best contribution to the objective function. If possible, it inserts the movie on a screen at a feasible position that yields the best improvement in the objective function. This process is repeated ψ times.

3. **Expensive Screen Insertion:** This operator selects the screen with the highest ticket price. If multiple screens have the same ticket price, it picks one of these screens randomly. If possible, it inserts a movie on this screen at a feasible position that yields the best improvement in the objective function. This process is repeated ψ times.

Selecting and Updating Operators

We use a roulette-wheel selection procedure to pick operators. The weight of each operation is directly proportional to its angle on the wheel. We denote ω_i as the weight of operator i. The weights are updated every ι operations by recalculating the weights according to the formula

$$\omega_i^{new} = \begin{cases} (1-\rho)\omega_i + \rho\frac{\xi_i}{\nu_i}, & \text{if}, \nu_i \neq 0, \\ (1-\rho)\omega_i, & \text{if}, \nu_i = 0. \end{cases} \tag{24}$$

Here, ρ represents the reaction factor that regulates the speed at which weight adjustments are made. Furthermore, ν_i counts the number of times operator i is used in the last ι operations, and ξ_i denotes the score of operator i. The score is calculated using the following rules. After the destroy, repair and improvement operators are performed on the solution, the score of the operators is updated as follows:

1. If the new solution s' improves the best solution s_{best}, we increase the scores of the operators that are used during the current iteration by ϵ_1.

2. If the new solution s' only improves the current solution s, we increase the scores of the operators that are used in the current iteration by ϵ_2.

3. If the new solution s' does not improve the current solution s and the best solution s_{best}, we increase the scores of the operators that are used in the current iteration by ϵ_3.

It's important to note that after the weights of all operators are updated, these weights are normalized within their respective sets, and both ν_i and ξ_i are reset to 0.

4.3 Combining Forecasting and Scheduling

As the session demand forecast and the schedule are dependent on each other, a strategy needs to be formulated how to combine the two. Firstly, a model to predict the demand for all sessions will be formulated without the movie schedule variables. This forecast is thereafter used in the optimization of the schedule as explained in Section 4.2. Note that this schedule thus does not take into account the effect of the schedule on the session demand. After the schedule is created, the demand forecast will be updated with the current movie schedule variables. Subsequently, we perform our ALNS heuristic on the current solution. After ALNS finished and a new solution has been found, the session demand forecast is updated and a new iteration of the method starts. This process is repeated until a stopping criterion is met. Firstly, the algorithm terminates when the schedule found by ALNS in the current iteration equals the schedule of the previous iteration, such that the forecast and solution will no longer change. Moreover, the algorithm terminates when a given number of iterations ($iter_{\max}$) is reached. An overview of the approach is presented in Algorithm 2.

Algorithm 2 Method combining forecasting and scheduling

```
1:  $d^0_{mt} \leftarrow forecastingModel()$                ▷ Initial session demand forecast
2:  $s_0 \leftarrow schedulingModel(d^0_{mt})$              ▷ Initial movie schedule
3:  $s \leftarrow s_0$
4:  $iter \leftarrow 1$
5:  while Stopping criteria not satisfied do
6:      $d^{iter}_{mt} \leftarrow forecastingModel(s)$     ▷ Update session demand forecast using previous solution
7:      $s \leftarrow schedulingModel(d^{iter}_{mt})$      ▷ Update movie schedule using updated demand forecast
8:      $iter \leftarrow iter + 1$
9:  end while
10: Output: $s$
```

5 Case Study

A case study is conducted using data from an exhibitor in the Middle East. The dataset contains transaction and ticket information on one of their locations, which has 24 screens. To clean the data we consider the returns by removing the corresponding sale and return transaction from the data, as we are interested in the number of people actually attending the movie. This results in dropping 16,268 out of 294,330 transactions. Moreover, we remove entries with highly unlikely or incomplete information. This includes removing the entries with a negative amount of seats sold or available (132 transactions dropped), removing the entries of films with no rating information (1 transaction dropped), and removing the entries of films with invalid starting times (2 transactions dropped). After cleaning the raw data, we obtain over a time interval of 2 months (30/06/2022 to 31/08/2022) a total of 273,537 transaction data points (one data point for each transaction).

From the cleaned set of transaction data, we create a dataset such that we have one entry per session. Note that a session is a unique combination of a movie, starting time and screen. The summary statistics of the (numerical) data entries of this dataset are presented in Table 6. An overview of the other characteristics of the data is presented in Table 7.

Table 6. Summary statistics sessions

Variable	**count**	**mean**	**std. dev.**	**minimum**	**25%**	**50%**	**75%**	**maximum**
Admissions	7742	35.33	37.49	1	10.25	24	44	340
Tickets available	7742	66.65	65.31	0	21	42	99	349

Table 7. Overview data

Variable	**Unique entries**	**Range**
Transaction ID	116368	-
Session ID	7742	-
Film ID	84	-
Film genre	12	action, adventure, . . . , war
Cinema type	5	IMAX, 4DX, KIDS, standard, theatre
Experience type	2	regular, premium
Movie start times	4633	[30/06/2022 23:45, 31/08/2022 23:30]
Movie language	11	Arabic, Chinese, . . . , Turkish
Screen	24	1, 2, . . . , 24

In this case study, we reconstruct the movie schedule for the last two weeks of our available data (18/08/2022 to 31/08/2022), considering each week separately. We provide a list of movies for these weeks in Table B2. Note that each movie listed must be screened at least once every day.

Next, the characteristics of each room are presented. The additional time needed after a movie to let people leave the room and to clean the room is dependent on the size and layout of the room. Note that the cleaning time of bigger and smaller rooms can be similar by scheduling more or less people to clean the room.[2] Since each room has only one seat type, the ticket price in a room is the weight of the seat type. The screen information is presented in Table 8

Table 8. Screen Characteristics

Screen	**Additional time afterwards**	**Type**	**Seat capacity**		**Ticket price**
			standard	**premium**	
1	15	IMAX	350	-	12.25
2	15	4DX	156	-	18.38
3	15	standard	173	-	8.58
4	15	standard	135	-	8.58
5	15	standard	135	-	8.58
6	15	standard	135	-	8.58
7	15	standard	44	-	8.58
8	15	standard	44	-	8.58
9	15	standard	44	-	8.58
10	15	standard	44	-	8.58
11	15	KIDS	-	43	9.80
12	15	KIDS	-	44	9.80
13	15	standard	103	-	8.58
14	15	standard	147	-	8.58
15	15	standard	105	-	8.58
16	15	standard	73	-	8.58
17	15	standard	182	-	8.58
18	15	KIDS	-	90	9.80
19	15	standard	87	-	8.58
20	15	standard	111	-	8.58
21	40	theatre	-	28	39.21
22	40	theatre	-	28	39.21
23	40	theatre	-	28	39.21
24	40	theatre	-	26	39.21

Note. Additional time afterwards is in minutes; ticket prices are in USD; only non-discounted ticket prices are considered.

Next to ticket and screen information, also some general information on the operation and logistics of the movie theatre are needed. Firstly, for the purpose of the case study, we set the time period t equal to one hour. Secondly, the preferred time the first movie can start is 10:00. Moreover, the latest time that a movie should preferably finish is 3:00. Thirdly, the time before the start of the

[2]The minimum additional time needed after a session has finished is determined by finding the median between the finish time and next starting time during peak hours (18:00 - 22:00).

movie to show commercials is either 15 or 20 minutes, depending on the movie. This information is displayed in Table B3 for the movies showing in the last two weeks of the available data.

Next, we specify the restriction on the total capacity of movie rooms starting and ending a screening at the same time interval. Specifically, for each time interval $[t, t+1]$, the total capacity of movie rooms ending a screening at time t and the total capacity of movie rooms starting screenings at time $t+1$ cannot exceed a certain number f_{rt}. An illustration of this restriction is given in Figure 2.

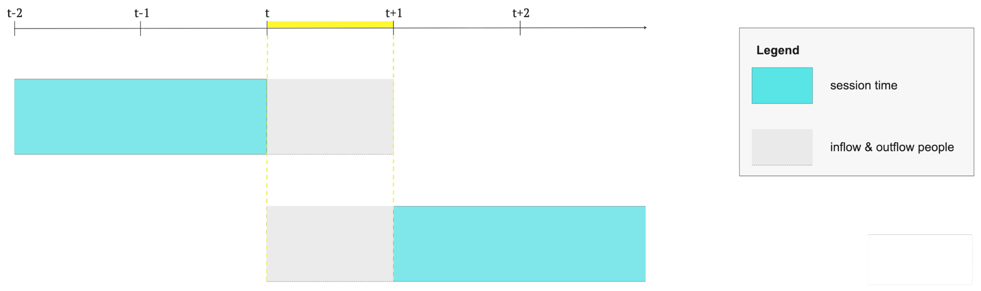


Figure 2. Inflow and outflow of people at time interval t to $t+1$

The value of f_{rt} is determined using the expected capacity utilization at t and the overall maximum allowed flow of people. This maximum flow is 1010 people in the entire cinema, 520 for rooms 2 to 12, and 840 for rooms 1 and 13 to 24. More specifically, to find f_{rt}, we divide the maximum allowed flow by the n^{th}-percentile of the capacity utilization of the time interval that t is included in. We use this division to account for the fact that sessions do not have to be fully booked. A distinct percentile value exists for each day and peak or non-peak hours. In this case study, we decide to use the 80^{th}-percentile. The capacity utilization statistics of the data are presented in Table 9. To exemplify, we allow for $[t, t+1]$ on a Monday during peak-hours the rooms starting and ending a screening at times as shown in Figure 2 to have a total maximum capacity of $1010/0.7483 \approx 1350$.

Table 9. Statistics Capacity Utilization using 80^{th}-percentile

Time	**Thursday**	**Friday**	**Saturday**	**Sunday**	**Monday**	**Tuesday**	**Wednesday**
Peak	83.43	92.86	95.43	96.21	74.83	62.39	66.06
Non-peak	39.29	60.37	78.57	84.09	36.36	32.97	30.94

Note. Values are the percentage of total capacity utilized; data between 31/06/2022 and 17/08/2022 is included in the calculations; peak hours are between 18:00 and 22:00.

Furthermore, at most fourteen movies can start a screening within every consecutive set of two time periods. The same holds for ending movies. Moreover, each movie can have specific constraints, for example that it must be shown on a specific set of screens, or it must be shown a given number of times in a given time interval. An overview of the movies with specific requirements specified by the distributor can be found in Table B4. Additionally, the management of the movie theatre requires that each day, there must be at least movies in four different languages screened. Similarly, it is required that each date at least movies in five different genres are offered. Finally, we include the desire of the management to start at least one movie every hour between 10:00 and midnight.

We currently only forecast the demand per movie at a specific time, and not per screen type, due to limited amount of data. Therefore, we also introduce a minimum or maximum requirement on the number of times per day that a movie must be shown at a specific screen type. This overview is presented in Table B5.

For the movie specific independent variables, specifically $METER$, $RATING$, $SEQUEL$, $ACTOR_{100}$, $ACTOR_{1000}$, $DIRECT_{1000}$, $DIRECT_{5000}$, $DISTR$, $PRODUCT$, $RELEASE$, we use the information available on IMDb, retrieved 30 May, 2023. The summary statistics of this data are presented in Table 10. The information for $HOLI$ is retrieved from the python package holiday, licensed by MIT. Moreover, for this case study we decide to exclude the variable $WEATHER$, since the climate in the location of the case study is characterised by negligible variation in weather over our time horizon. Finally, the variables $LANG_l$, $GENRE_g$ and $TICKETS$ are available in the dataset of the multiplex cinema.

Table 10. Summary statistics movie characteristics retrieved from IMDb

Variable	**count**	**mean**	**std. dev.**	**minimum**	**25%**	**50%**	**75%**	**maximum**
$METER$	71	27076.39	31157.86	132	5523.5	13585	43360	121914
$RATING$	71	5.83	1.27	2.80	5.00	5.80	6.80	8.80
$SEQUEL$	71	0.15	0.36	0	0	0	0	1
$ACTOR_{100}$	71	0.11	0.32	0	0	0	0	1
$ACTOR_{1000}$	71	0.23	0.42	0	0	0	0	1
$DIRECT_{1000}$	71	0.01	0.12	0	0	0	0	1
$DIRECT_{5000}$	71	0.10	0.30	0	0	0	0	1
$DISTR$	71	0.21	0.41	0	0	0	0	1
$PRODUCT$	71	0.20	0.40	0	0	0	0	1

Note. Only movies with release date before 17/08/2022 are included in the calculations.

6 Results

The results of the Movie Scheduling Problem for the case study introduced in Chapter 5 are presented in this section. Firstly, Section 6.1 discusses the results of the session demand forecast models. Secondly, the results of the schedule optimization methods given the forecast are presented in Section 6.2. Finally, Section 6.3 shows the result when the session demand forecasting and schedule optimization are combined.

6.1 Forecasting Results

To model the demand for movie m at time t, we tune and test two different models and compare their performance. The first set includes the available data from 30/06/2022 to 24/08/2022. The training set consist of 6087 sessions, which have a show time between 30/06/2022 and 17/08/2022. The testing set consist of 851 sessions, which have a show time in week 18/08/2022 to 24/08/2022. The second set includes the available data from 30/06/2022 to 31/08/2022. In this case, the training set contains all data of the first seven weeks (30/06/2022 to 24/08/2022), specifically 6939 sessions. The testing set consist of 803 sessions, which have a show time in the final week (25/08/2022 to 31/08/2022). In this section we show the results of including all independent variables in the model. However, note that if schedule information is not yet available, we use the model fitted without the movie schedule variables to forecast session demand.

6.1.1 Linear Regression

Before determining the estimates of the coefficients for our linear regression model, we first examine the degree of skewness of the data by determining the Fisher-Pearson coefficient of skewness. We find a coefficient of skewness of 2.492, signaling that the distribution is asymmetrical with more weight in its left tail. Figure 3 shows the histogram of the dependent variable, confirming this finding. We therefore do apply a logarithmic transformation of the dependent variable movie demand at time t. In this case, the coefficient of skewness is -0.386, and thus the symmetry is improved.

Next, we determine the estimates of coefficients of the independent variables by OLS using the equation presented in Section 4.1.1. The results are presented in Table B in Appendix B.

The models presented in Table B in Appendix B are used to predict the session demand in week 18/08/2022 to 24/08/2022 and week 25/08/2022 to 31/08/2022, respectively. Figure 4 compares the true values to the predicted values of the test set. For both weeks we observe a greater variation

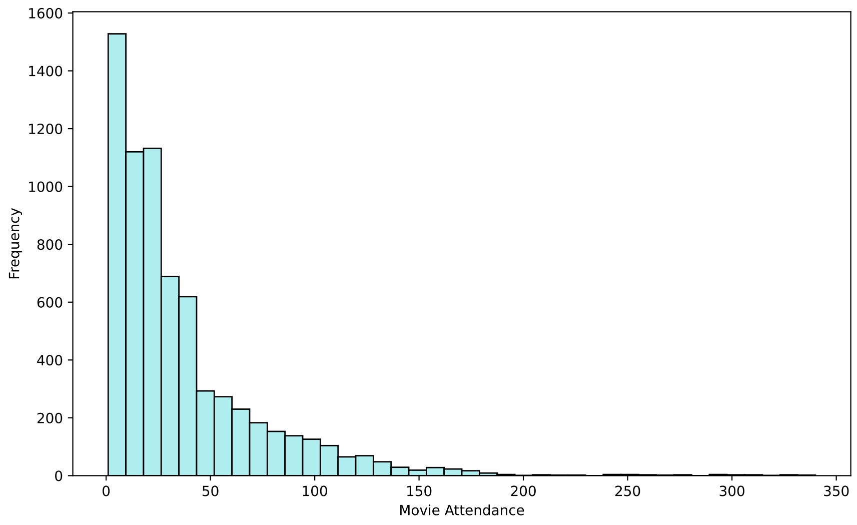
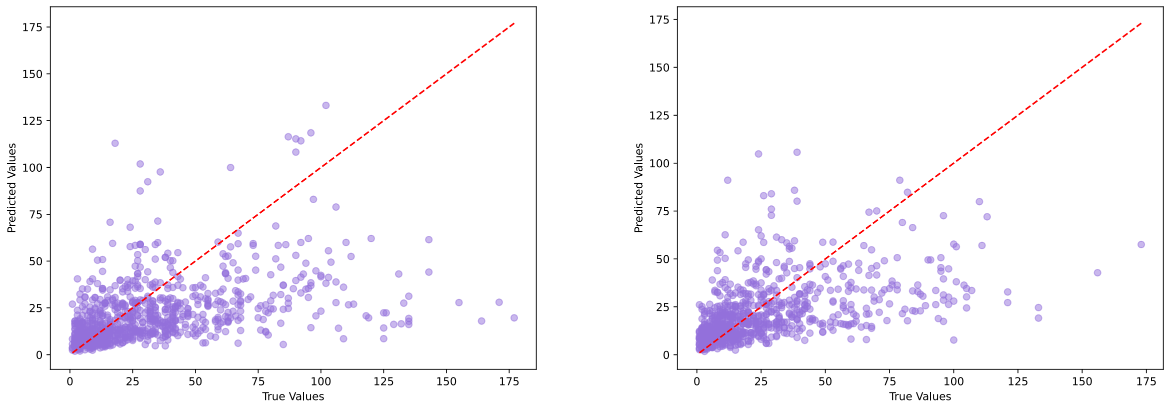


Figure 3. Histogram of attendance per session

from the 45 degree line for higher demand values. Furthermore, mainly the session demand for week 18/08/2022 to 24/08/2022 shows underpredicition for higher actual values.



(a) testing set week 18/08/2022 to 24/08/2022 *(b)* testing set week 25/08/2022 to 31/08/2022

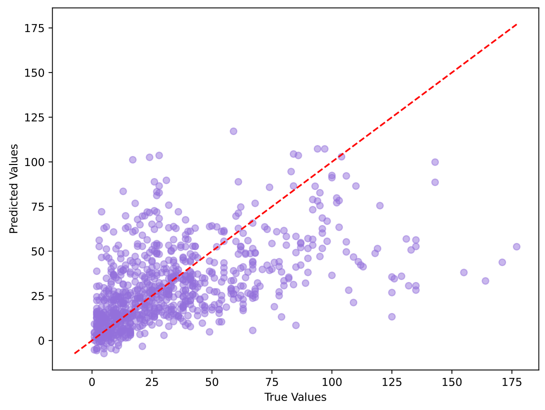
Figure 4. True values versus predicted values Linear Regression

6.1.2 Gradient Tree Boosting

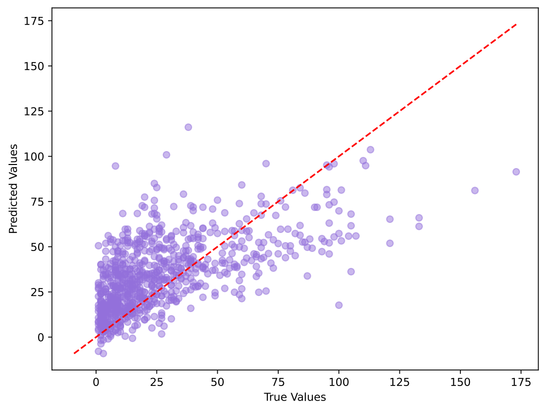
To create our Gradient Tree Boosting (GTB) model, we firstly need to tune the hyperparameters as described in Section 4.1.2. We tune the model using a training set consisting of weeks 30/06/2022 to 17/08/2022. The hyperparameters are tuned on this training set using a grid search with 5-fold cross validation. The best model is obtained by selecting the following values for the hyperparameters. Firstly, for the boosting parameters, the learning rate is set equal to 0.1, and the number of trees equal to 300. Next, for the tree-specific parameters, the minimum required samples to split a node is set equal to 30, the maximum depth of a tree equal to 8, and the maximum number of features considered at each split equal to 12.

To predict the session demand for week 18/08/2022 to 24/08/2022, we fit the model using the data of weeks 30/06/2022 to 17/08/2022, giving an R^2 of 0.725. To predict the session demand for week 25/08/2022 to 31/08/2022, we fit the model using the data of weeks 30/06/2022 to 24/08/2022 and similarly compare the prediction to the true attendance. The R^2 of the model is 0.708.

Figure 5 compares the true values to the predicted values of the test set. We observe that both weeks show a similar pattern. In general, the observations are reasonably well centered along the 45 degree line. However, the session demand tends to be underpredicted for high actual values. Furthermore, especially for week 25/08/2022 to 31/08/2022, the expected demand is oftentimes too high for low true values.



(a) testing set week 18/08/2022 to 24/08/2022

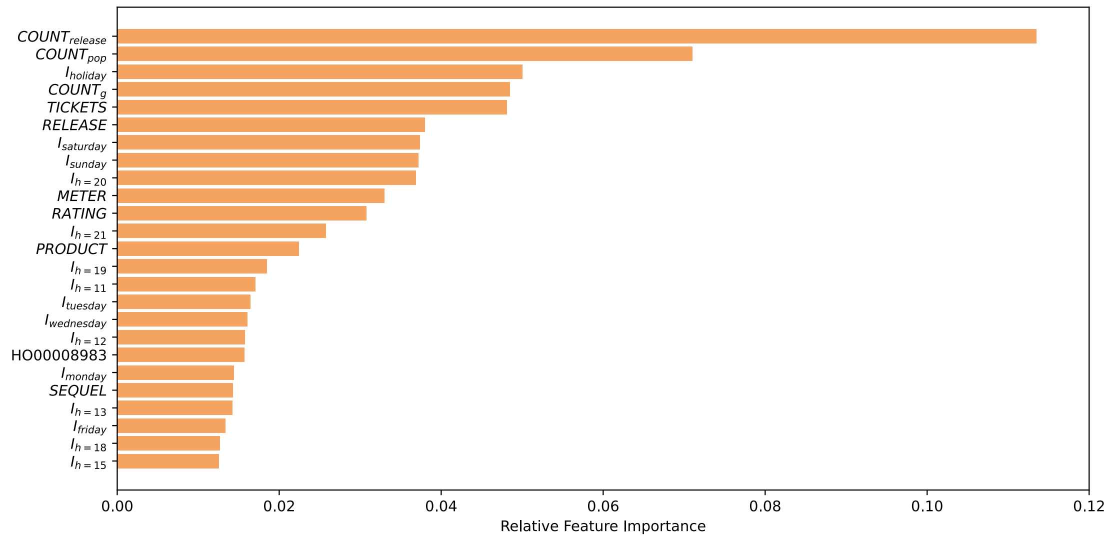


(b) testing set week 25/08/2022 to 31/08/2022

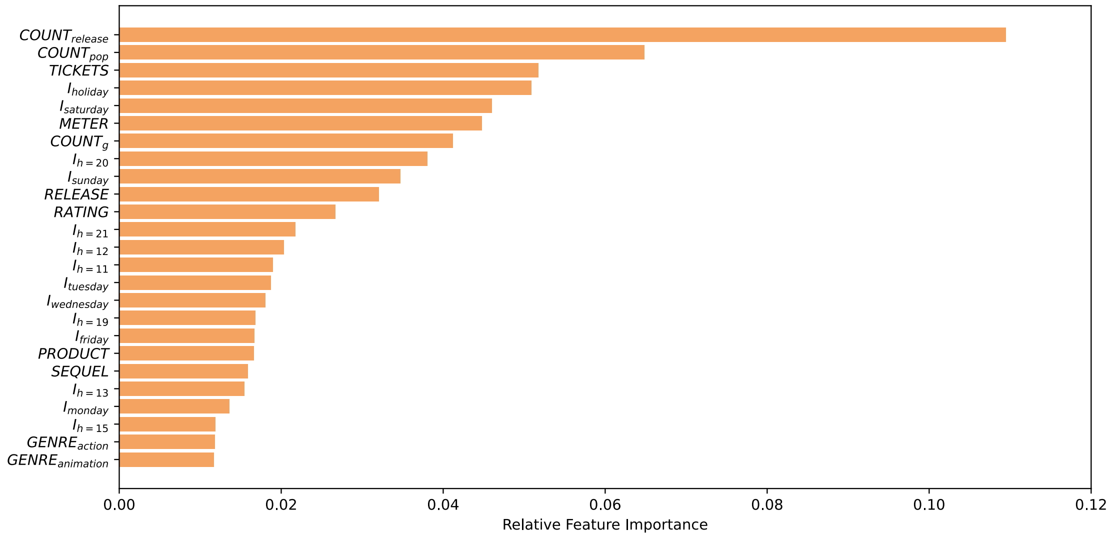
Figure 5. True values versus predicted values Linear Regression

Finally, we look at the independent variables. Figure 6 shows the top 25 most important features to forecast the session demand in week 18/08/2022 to 24/08/2022 and week 25/08/2022 to 31/08/2022 using the GTB model.

It should be noted that the set of 25 most important features comprises nearly identical features. However, the exact contribution to the importance does differ somewhat between the models. We observe that the movie schedule variables are highly important in our forecasting model. Note that this also highlights the relevance of this thesis, which integrates the session demand forecast and the schedule optimization. Other important variables are the number of tickets sold last week, the release week, and the public and professional rating of a movie. Note that also some of the day and hour indicators have a high feature importance.



(a) training set weeks 30/06/2022 to 17/08/2022



(b) training set weeks 30/06/2022 to 24/08/2022

Figure 6. Feature importance of top 25 most important features

6.1.3 Comparison Forecasting Methods

To analyse the performance of the models, we look at several evaluation metrics. Table 11 gives an overview of these metrics.

Table 11. Overview of evaluation metrics

	testing set week 18/08/2022 to 24/08/2022		**testing set week 25/08/2022 to 31/08/2022**	
Metric	**Linear Regression**	**GTB**	**Linear Regression**	**GTB**
Mean Squared Error	892.0170	720.2489	560.6710	490.1069
Root Mean Squared Error	29.8667	26.8375	23.6785	22.1384
Mean Absolute Error	19.5599	18.3297	15.5416	16.7949

Note. The metrics of Linear Regression are determined after transforming the logarithmic dependent value back to its linear value.

Firstly note that the evaluation metrics are better when using week 25/08/2022 to 31/08/2022 as a testing set. This can be explained by presence of additional data from week 18/08/2022 to 24/08/2022 in the training set of this model, in contrast to the model trained solely on weeks 30/06/2022 to 17/08/2022. Furthermore, we find that the evaluation metrics for the GTB models are in all cases better than those for the Linear Regression models, with the exception of the Mean Absolute Error of week 25/08/2022 to 31/08/2022. This confirms the overall observation when comparing Figure 4 to Figure 5. Hence, we decide to use our GTB models to forecast the session demand in our scheduling models.

6.2 Scheduling Results

This section presents the results of optimizing the movie schedule for the case study presented in Chapter 5, given the demand forecast from the GTB model. To do so, a network is constructed as described in Section 4.2.1. Due to the large size of the problem, the network and models are created for each day separately. Note that we do not follow the standard time-day notation, but instead we let a day start and end at 9:00 in the morning such that movies can be scheduled during midnight. An overview of the size of the network can be found in Table 12. Moreover, penalties for not meeting the soft constraints need to be specified. For this case study, we set the penalty parameters as follows: $\lambda_1 = 1100$, $\lambda_2 = 50$, $\lambda_3 = 100$, and $\lambda_4 = \lambda_5 = 10$. Finally, note that weight of a path is the negative of the expected total revenue obtained on a screen.

Table 12. Network size

Day	**Thursday**	**Friday**	**Saturday**	**Sunday**	**Monday**	**Tuesday**	**Wednesday**
			week: 18/08/2022 - 24/08/2022				
Number of nodes	9044	9344	9344	9344	9344	9344	9344
Number of arcs	1438062	1540602	1540602	1540602	1540602	1540602	1540602
			week: 25/08/2022 - 31/08/2022				
Number of nodes	8272	8272	8272	8272	8272	8272	8572
Number of arcs	1248868	1248868	1248868	1248868	1248868	1248868	1343698

Note. Days start and end at 9:00.

6.2.1 Column Generation

We firstly want to remark that in this section the mathematical program solver CPLEX version 20.1 for Java is used. We begin with tuning the parameters of the Column Generation procedure. We choose Sunday 28/08/2022 for this purpose, since we have most data available for this week, and Sunday has the highest total attendance.

We start with determining the number of shortest paths to find (k) for each screen, to add to the RMP if the path has a negative reduced cost. To tune this parameter, we set the termination parameter n equal to 5000 paths, where we do finish the current iteration. Moreover, for tuning purposes, the optimality gap when solving the final MIP ((1)-(23)) is set equal to 1%. Finally, we set our k-Shortest Path algorithm to remember the best 100 sub-paths during the algorithm. Note that although our pricing problem is thus solved using a pricing heuristic, due to the characteristics of our network and problem instances, the shortest path is often included in our solution. The results are shown in Table 13.

Table 13. Column Generation results tuning parameter k, Sunday 28/08/2022

k	**Objective value**	**Lower bound on objective value**	**Number of paths in the RMP**	**Running time Column Generation**	**Running time MIP**
10	-55755.48	-61321.97	2376	2893	32
20	-55485.99	-61321.97	3720	2757	429
30	-55170.58	-60671.12	5041	2797	1552
40	-54388.02	-58672.67	5322	1144	224
50	-52637.29	-57453.54	5320	813	36
60	-46419.63	-54120.51	5486	655	39
70	-48317.53	-55476.13	6408	709	34
80	-45003.83	-49771.00	5548	445	9
90	-45038.07	-49837.67	6234	476	33
100	-44972.79	-49927.08	6925	477	34

Note. The lower bound is found by solving the linear relaxation of the problem ((1)-(23)); running time is in seconds.

We firstly observe a decrease in running time for the Column Generation procedure as we increase k. Note that this increase mainly results from needing to do more iterations. Secondly, the objective value has an increasing trend as k increases. This indicates that reducing the number of columns generated in each iteration of the pricing problem typically improves the objective value whilst increasing the overall solution time. We decide to use the value for k that gives a reasonably good objective value for an average running time, and thus set k equal to 40.

Next, we tune the stopping criterion. To do so, we let termination of the Column Generation procedure only happen when no more columns with negative reduced cost can be found by the pricing heuristic. Figure 7 shows the decrease in objective function per iteration. We observe that after 7 iterations, the decrease in the objective value diminishes. At this point, there are 5645 paths added to the Restricted Master Problem (RMP), and we therefore decide to set the stopping criterion equal to $n = 5500$.

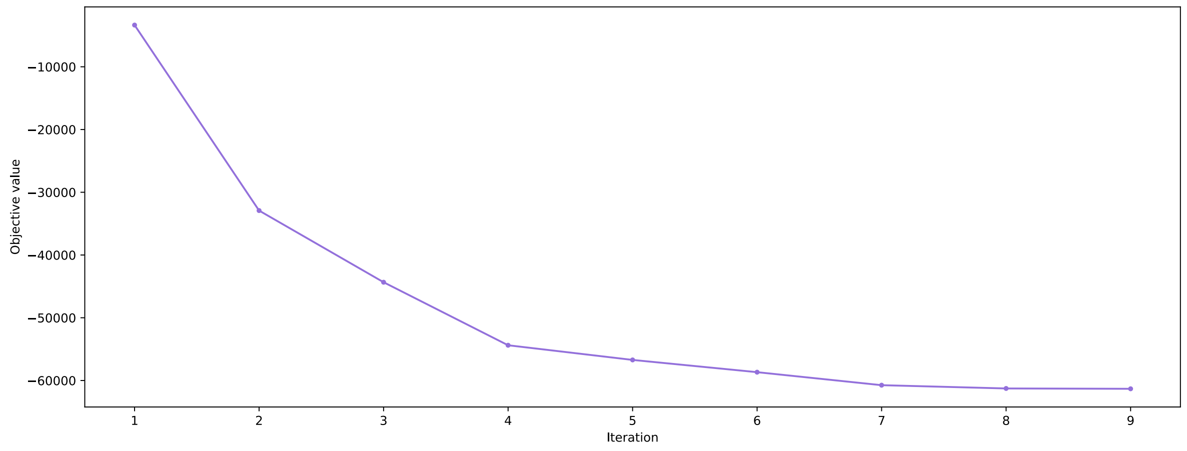


Figure 7. Column Generation result linear relaxation per iteration, Sunday 28/08/2022

We now optimize the schedule for all days using the Column Generation procedure with $n = 5500$, $k = 40$, and $p = 60$. The results are presented in Table B7 in Appendix B, and visualized in Figure 8.

We observe that both the objective value and the total weight of the paths is much better for week 25/08/2022 to 31/08/2022 than for week 18/08/2022 to 24/08/2022. This holds for both the values per day, as well as the aggregated result. Moreover, the total penalty and running time (Table B7) is considerably higher for week 18/08/2022 to 24/08/2022. Thus, week 25/08/2022 to 31/08/2022 is, according to this method, a much better performing week.

Next, the best objective value is obtained for the weekend days, and we notice that for week 25/08/2022 to 31/08/2022 the objective keeps improving from Thursday to Sunday. No clear trend

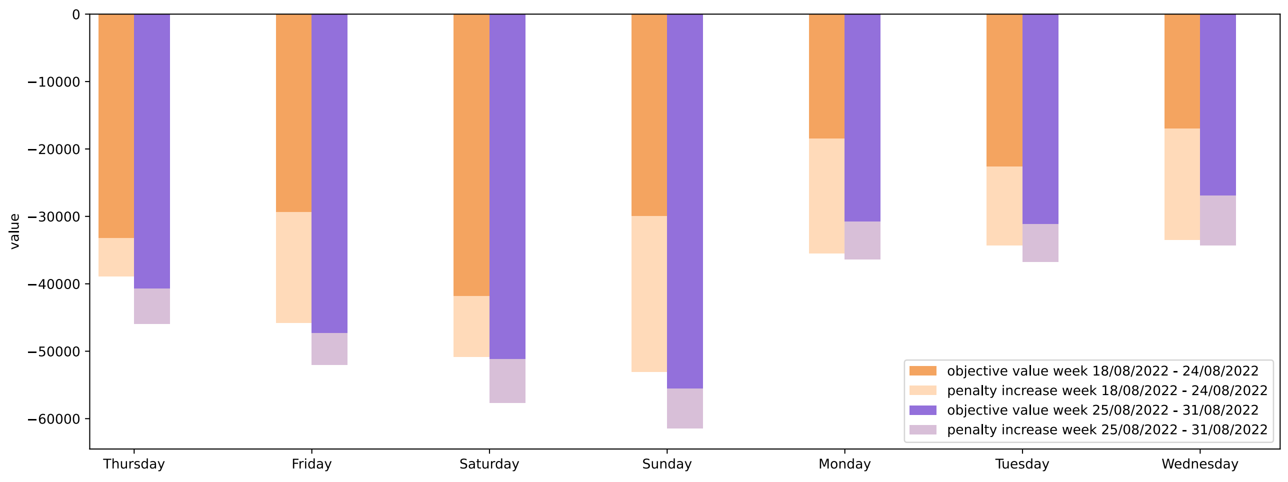


Figure 8. Result Column generation

in the total penalty per day can be identified. From Table B7, we do however see a slight increase in the running time as the objective value improves.

The detailed schedule generated for Sunday 28/08/2022 can be found in Figure C1 in Appendix C. Note that the different colours in the schedule represent the different genres. We observe that some movies are scheduled very often, whereas others are only scheduled once or twice. The movies that are scheduled often have a high expected demand, and since here the expected demand is still independent of the rest of the schedule, many sessions are scheduled to obtain a good objective value. Moreover, almost all sessions are scheduled between 10:00 and 3:00 to avoid penalty costs of sessions in undesired time periods.

6.2.2 Adaptive Large Neighborhood Search

We start by setting the parameters of the Adaptive Large Neighborhood Search (ALNS). We set the parameter values as $\epsilon_1 = 50$, $\epsilon_2 = 20$, $\epsilon_3 = 12$, $\alpha = 0.999$, $\rho = 0.1$, $\iota = 50$, $\theta = 200$, and we further tune κ, ψ, and the stopping criteria. Similar as for the previous method, we use Sunday 28/08/2022 for this purpose.

We start with determining the number of sessions to destroy, κ, and the number of sessions to add for improvement ψ. Note that for this purpose we have the stopping criteria set as $\chi_1 = 20000$ and $\chi_2 = 12000$. Table 14 shows the tuning results.

Table 14. Objective value for tuning parameters κ and ψ, Sunday 28/08/2022 (ALNS)

$\kappa \setminus \psi$	**4**	**6**	**8**	**10**	**12**	**14**
4	-41375.16	-48912.13	-43564.39	-45732.72	-44942.88	-45486.93
6	-47436.66	-45759.17	-46289.56	-41009.13	-48792.22	-45281.22
8	-48256.57	-46934.99	-45566.46	-45144.64	-46667.17	-46441.19
10	-47001.30	-45137.79	-40555.53	-46948.80	-45820.28	-43539.49
12	-46527.87	-46705.29	-46721.01	-45861.74	-45408.74	-45486.18
14	-40993.69	-46294.41	-46229.93	-44605.97	-47643.78	-46126.21

The results do not show an obvious relation between the objective value and the parameters. However, if we look closely, we observe that the values for $\kappa = 8$ are generally good. Similarly, $\psi = 6$ shows good and stable resutls for different values of κ. We therefore decide to set $\kappa = 8$ and $\psi = 6$.

Subsequently, we determine the stopping criteria. We therefore run the algorithm for a large number of iterations. the result is shown in Figure 9.

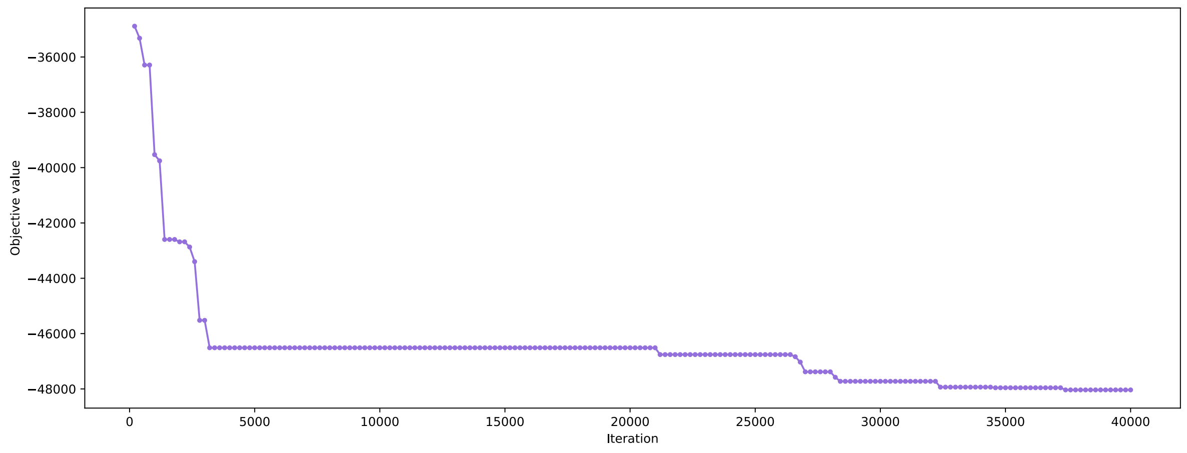


Figure 9. ALNS objective value per iteration, Sunday 28/08/2022

We observe that after around 3500 iterations we get stuck in a local optimum for a long time. After escaping this local optimum, the objective value only improves with approximately 3%. Note that the total running time increases with the total number of iterations. However, the number of iterations performed before being trapped in a local optimum differs per instance, and we should therefore also not stop the algorithm too early. Taking all this into account, we decide to stop the algorithm after 10000 iterations ($\chi_1 = 10000$) or 6000 non-improving iterations ($\chi_2 = 6000$).

The movie schedule for all days is now optimized using ALNS with parameters $\kappa = 8$, $\psi = 6$, $\chi_1 = 10000$, and $\chi_2 = 6000$. The results are presented in Table B8 in Appendix B, and displayed in Figure 10.

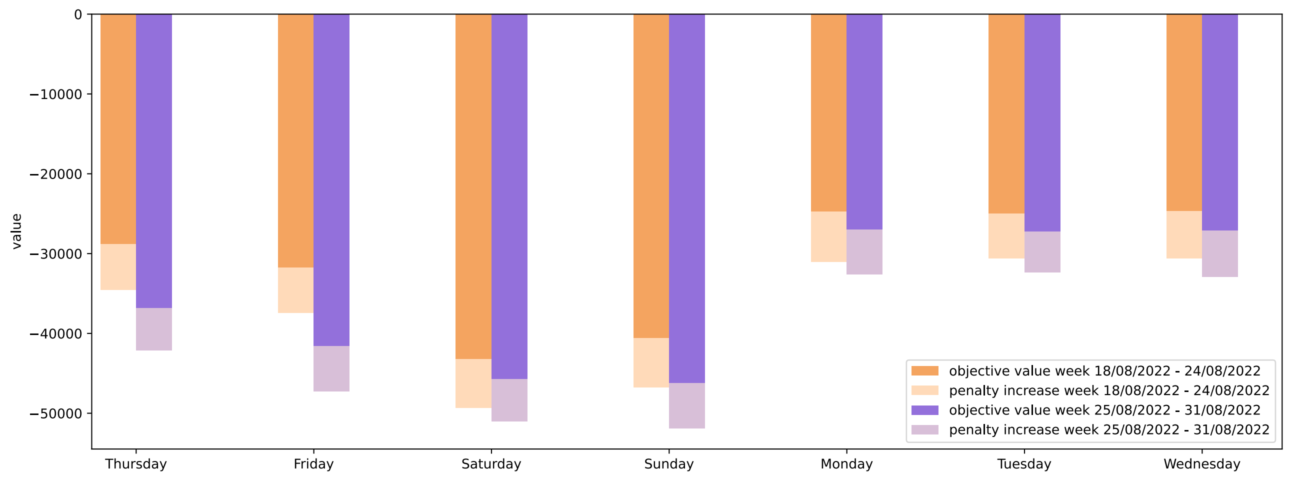


Figure 10. Result ALNS

The objective value and total weight of paths is better for week 25/08/2022 to 31/08/2022 than for week 18/08/2022 to 24/08/2022, for both the individual as well as the aggregated results. The weekend days have the best objective value and total path weight. Furthermore, the total penalty is slightly higher for week 18/08/2022 to 24/08/2022 than for week 25/08/2022 to 31/08/2022. Note from Table B8 that the running times are constant over all instances.

The detailed schedule generated for Sunday 28/08/2022 can be found in Figure C2 in Appendix C. We observe that no movies are scheduled in undesired time periods (3:00 to 10:00). Moreover, some movies are again scheduled often, whereas other movies are only in the schedule once or twice.

Finally, we have a closer look at some penalty parameters using ALNS. We again use the schedule of Sunday 28/08/2022 for this purpose. Firstly, the sensitivity of parameter λ_1 is analyzed. This penalty parameter corresponds to constraint (10), which penalizes the use of undesired time periods. Note that we leave all other parameters unchanged. The results are shown in Table 15.

Table 15. Sensitivity analysis λ_1, Sunday 28/08/2022 (ALNS)

λ_1	Objective value	Total weight paths	Total number of undesired time periods used (σ)
0	-60229.64	-66229.64	33
400	-50480.95	-65180.95	22
800	-42661.90	-57061.90	11
1200	-45485.76	-51185.76	0
1600	-44622.52	-49922.52	0
∞	-44679.01	-50179.01	0

We notice that as λ_1 increases up to a value of 1200, both the objective value and the total weight of all paths increase. This signals that as constraint (10) becomes more restricting, the resulting solutions show a decreasing trend in the total expected revenue. From λ_1 equal to 1200 onward, an increase in the value of λ_1 does not significantly affect the solution. Hence, from this value, constraint (10) becomes a hard constraint.

Secondly, we analyze the sensitivity of penalty parameter λ_3. This parameter corresponds to constraint (12), which makes the use of multiple screens for each movie less attractive. Table 16 presents the results.

Table 16. Sensitivity analysis λ_3, Sunday 28/08/2022 (ALNS)

$\boldsymbol{\lambda_3}$	**Objective value**	**Total weight paths**	**Total number of screens used for each movie** $(\sum_{\omega \in W} \sum_{m \in M} \sum_{s \in S^m} \zeta^{\omega}_{ms})$
0	-47661.77	-53161.77	63
100	-46186.86	-51886.86	57
200	-42079.68	-51679.68	48
300	-34837.29	-49237.29	48
400	-31618.31	-48418.31	42
500	-28417.26	-48417.26	40
600	-26531.77	-47531.77	35
700	-20835.53	-48135.53	39
800	-18980.67	-46180.67	34
900	-15107.07	-45707.07	34
1000	-12025.49	-44025.49	32

We observe that the objective value and total weight of all paths increase as the penalty parameter λ_3 increases. Moreover, we note that as the value of λ_3 increases, also the total number of screens used for each movie decreases. This signals that increasing penalty parameter λ_3 results in an improved fulfillment of the preference for movies to be shown on the same set of screens. However, this has a worsening effect on the total weight of the paths that can be achieved.

6.2.3 Comparison Scheduling Methods

We compare the performance of Column Generation and ALNS heuristic, by firstly looking at the results for all days presented in Table B7 and Table B8. Looking at aggregate result of week 18/08/2022 to 24/08/2022, we observe that ALNS obtains solutions with a better total objective value, whereas the aggregated path weight is better for the solution obtained by Column Generation. This difference is caused by the much higher aggregate penalty value for the solution of Column Generation. Next, the solutions obtained by Column Generation for week 25/08/2022 to

31/08/2022 have a better objective value and total paths weight than the solutions obtained by ALNS. Specifically, the solutions obtained by Column Generation have an approximately 9% better objective value and total path weight, with respect to the solution obtained by ALNS. The total penalty lies around the same value for the two methods for this week. Finally, we observe that for all instances the total running time is much higher for Column Generation than the ALNS heuristic.

Comparing the detailed schedule of Sunday 28/08/2022 presented in Figure C1 and Figure C2, we observe that the schedule obtained by Column Generation shows less variation than the schedule obtained by ALNS. In other words, the amount of times each movie is scheduled is more evenly distributed in the schedule obtained by ALNS than in the schedule obtained by Column Generation. Namely, in the schedule obtained by Column Generation, movies HO00009116 and HO0009294 are scheduled very often, and most other movies are scheduled only once or twice. In the schedule obtained by ALNS these two movies are still scheduled regularly, however less times than in the schedule for Column Generation, and other movies are also scheduled more often.

Increasing Instance Size

The results shown so far are for time periods equal to one hour. To make a more precise schedule, we need to make the time periods smaller. In this section we present the results obtained when setting the time period equal to 15 minutes. Note that we only change the stopping criteria of the methods, and leave all other instance and method parameters unchanged.

Making the time periods smaller results in a larger network. Specifically, the amount of nodes in our network formulation increases to around four times its original count, and the amount of arcs to approximately twenty times its original count. The stopping criterion of Column Generation is set equal to 10000 paths in the Restricted Master Problem. Figure B1 in Appendix B displays the trajectory of the objective value per iteration, showing some convergence towards the end. The stopping criteria of ALNS is set to $\chi_1 = 20000$ and $\chi_2 = 12000$. Figure B2 presents the objective value per iteration, showing clear convergence from around 6000 iterations.

Using these stopping criteria, we compare the two different methods in Figure 11 for the instance of Sunday 28/08/2022. In Figure B3 in Appendix B, we display the first 2000 seconds of this figure. Moreover, Table B9 shows the final results.

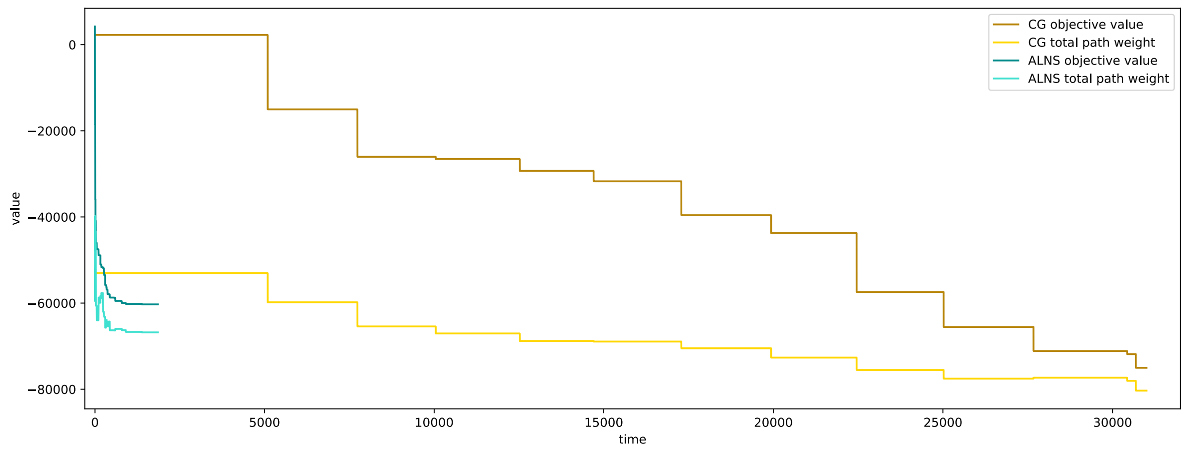


Figure 11. Comparison Column Generation and ALNS over time in seconds, Sunday 28/08/2022

Column Generation obtains a solution with a better objective value and total weight of paths than ALNS. Note that the penalty is approximately equal between the two solutions. However, the running time of Column Generation to obtain a solution better than that of ALNS is much longer than the total running time of ALNS. More specifically, the total running time of Column Generation is around 17 times as high as the running time of the ALNS heuristic to obtain the final solution.

6.3 Combining Forecasting and Scheduling

In this section, the forecasting and optimization methods are combined as described in Section 4.3. The initial session demand forecast is made using the GTB model without movie schedule variables. We use the Column Generation methods with parameters $k = 40$, $p = 60$, and $n = 5500$ to perform the first movie schedule optimization. We thereafter iteratively update the demand forecast using the GTB model with movie schedule variables, and re-optimize the movie schedule from the current solution using the ALNS heuristic with parameters $\epsilon_1 = 50$, $\epsilon_2 = 20$, $\epsilon_3 = 12$, $\alpha = 0.999$, $\rho = 0.1$, $\iota = 50$, $\theta = 200$, $\kappa = 8$, $\psi = 6$, $\chi_1 = 10000$, and $\chi_2 = 6000$.

We run the method for Sunday 28/08/2022. Firstly, we determine the number of iterations to perform. Figure 12 displays the objective value per iteration of the method combining forecasting and scheduling (Algorithm 2).

The figure shows some under- and overshooting pattern in the first few iterations, but appears to be stabilizing from around iteration 7 at an objective value of approximately -38000. We therefore decide on 8 iterations ($iter_{\max} = 8$) for this instance.

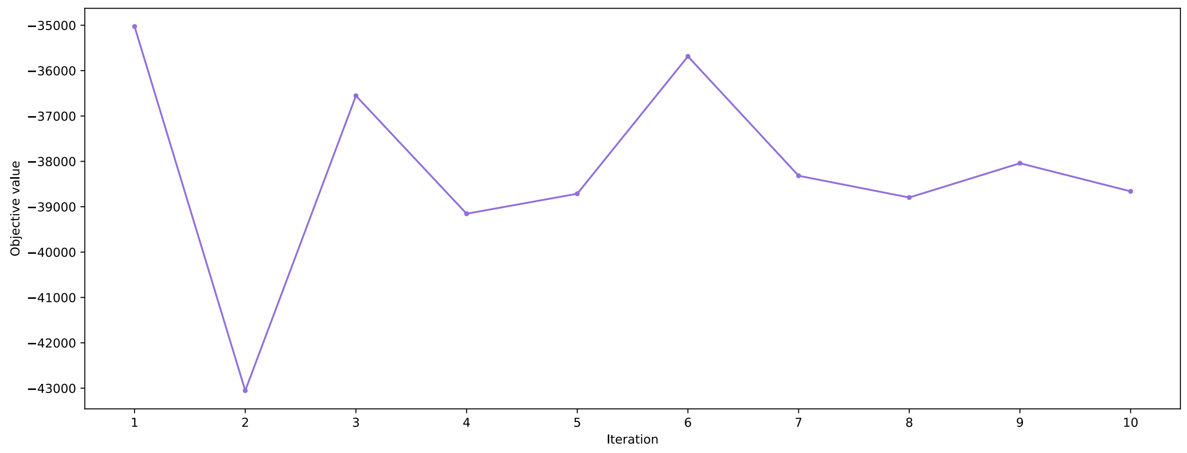


Figure 12. Objective value per iteration combining forecasting and scheduling, Sunday 28/08/2022

Setting $iter_{\max} = 8$ gives a final solution for Sunday 28/08/2022 with objective value -38796.63 and total path weights of -49496.63. The penalty is thus 10700. Taking the initial solution of Sunday 28/08/2022 as presented in Figure C1, the total running time of this method is 1863 seconds. The detailed schedule is presented in Figure 13. Note that the sessions that changed with respect to the initial schedule (Figure C1) have a striped, red border. We observe that the resulting schedule shows more variation than the initial solution, such that the amount of times each movie is scheduled is more evenly distributed. Moreover, five sessions are scheduled during undesired time periods, whereas in the initial schedule only one. This mostly explains the higher penalty score of the final solution.

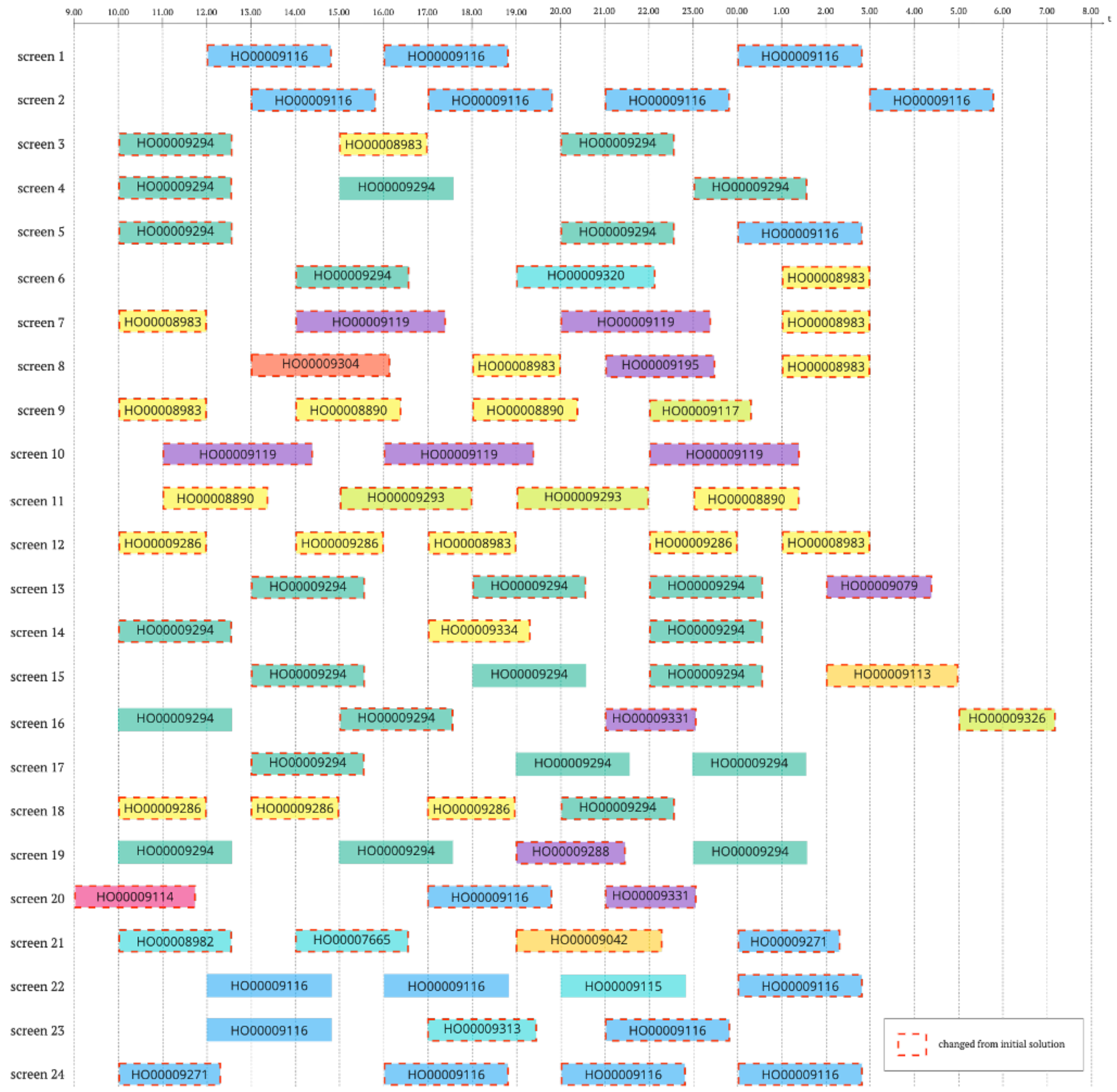


Figure 13. Schedule combining forecasting and scheduling Sunday 28/08/2022

7 Conclusion

This thesis examined the Movie Scheduling Problem for movie theatres. The recent changes in the movie exhibition business and the many preferences for and restrictions on the schedule create a large and complex scheduling problem. In this thesis, we therefore aimed to develop a method to find the optimal movie schedule for movie theatres.

To answer the research question, logistical, distributor, and managerial requirements are specified. The optimal movie schedule is defined as the schedule with the highest total ticket revenue with as little violation of the soft (managerial) requirements as possible. We analyze the problem in two stages, using a case study from a multiplex cinema in the Middle East to test the methods. Firstly, we formulated models to forecast the demand for a session. General, movie specific, and schedule specific variables are determined, and using these variables the performance of a linear regression model and a Gradient Tree Boosting (GTB) model are compared. The evaluation metrics signal that the GTB model captures the interactions in the data best, and the GTB model is therefore our preferred model.

Secondly, methods to optimize the movie schedule given the demand for the sessions are defined and their performance is evaluated. The problem is represented as a network with multiple layers, and a Set Partitioning Problem is formulated. To solve the model, the performance of Column Generation with a k-Shortest Path Problem as pricing heuristic is compared to an Adaptive Large Neighborhood Search (ALNS) heuristic. The solutions obtained by Column Generation have an approximately 9% better objective value and total ticket revenue than the solutions obtained by ALNS for all instances. The total running time of Column Generation is however 9 to 45 times larger than that of ALNS. This larger running time can especially create problems when the problem instance size increases. Moreover, the total penalty of the solution obtained by ALNS is rather constant, whereas total penalty of the solution obtained Column Generation shows much more variation. A high penalty possibly occurs if insufficient well performing paths are created by Column Generation when the stopping criterion is satisfied. This signals that the performance of Column Generation is more dependent on the tuning of the parameters. Considering all this, the preferred scheduling method is thus largely dependent on the available solving time, desired quality of the solution, and the problem instance size.

Finally, the session demand forecasting and schedule optimization methods are iteratively combined to determine the final optimal schedule. This approach tried to capture the dependency of the movie schedule on the expected demand for a session, and vice versa. We chose to use the solution obtained by Column Generation as initial solution, used the GTB model to update the demand forecast, and used our ALNS heuristic to update the schedule. The resulting final schedule is much different from the initial schedule. The main differences are that the amount of times each movie is scheduled is more evenly distributed in the final solution compared to the initial solution, and that the final solution uses more undesired time periods to schedule movies. From these observations we conclude that interactions between the expected demand for a session and schedule are to a certain extend captured by this iterative method, but the method struggles to find an optimal solution with little violation of the soft constraints.

8 Discussion

The methods presented in this thesis are able to obtain desirable movie schedules, but also have some limitations. To improve, the following limitations can be further analyzed and the methods can be extended.

Looking at the session demand forecast, we firstly note that the forecast is based on the data of 6087 or 6939 sessions for week 18/08/2022 to 24/08/2022 or week 25/08/2022 to 31/08/2022, respectively. This is a relatively small amount of data, especially when considering the amount of possible sessions for which the demand needs to be predicted. To improve, we should collect more data such that we obtain a larger set of training data. Next to that, other session information could be recorded to analyze whether inclusion of these other explanatory variables in the forecasting model improves its performance. To improve the forecast without much additional data, we could research the success of a clustering approach, where for example similar movies are clustered and a demand forecast is done per cluster of movies. Contrary to making the forecast less specific by grouping data, the forecast can be made more detailed by predicting the movie demand per session for a certain ticket price or room type. This allows the scheduling method to have demand predictions differing per experience or seat type. Furthermore, our current Gradient Tree Boosting (GTB) model can be further developed by additional tuning of the parameters and design choices. Moreover, we currently use the observed attendance data to predict session demand. However, these data do not take into account demand for sessions that are sold out or sessions that do not exist in the past schedule. Modeling techniques that do take this into account can be investigated to overcome this limitation. Finally, the performance of other non-linear (machine learning) methods can be analyzed and compared to our current model of choice.

Furthermore, our movie schedule optimization methods exhibit some limitations, pointing out promising suggestions for further research. First of all, we note that none of the introduced methods are able to guarantee optimality of the obtained solution. Another important limitation of our scheduling methods is the size of the problem representation. Specifically, adding more movies or screens to the problem instance, or making the size of the time periods smaller, quickly increases the total size of the network and therefore the number of possible paths. Furthermore, to further improve the performance of Column Generation, we could continue tuning the parameters and analyze different stopping criteria. Moreover, an exact pricing problem such as the Shortest Path Problem can be implemented after our current pricing heuristic is not able to find any paths with reduced cost. Another suggestion for further research is to extend the method to a branch-and-price framework, also analyzing different branching rules. As well as for the other methods, also the performance of the Adaptive Large Neighborhood Search (ALNS) heuristic could possibly be improved by further tuning of its parameters. Moreover, the introduction of other destroy, repair, or improvement operations could lead to the construction of better solutions. In line with this, the exploration of different neighborhoods in the local search can help improving the solution.

Finally, a start in the integration of session demand forecasting and schedule optimization has been made. A possible first research extension in this area could be to analyze the trajectory of the optimal solution over many more iterations. Subsequently, the effect of updating the demand forecast more frequently can be examined. By improving this method, the relation between the session demand and the movie schedule could be captured better, resulting in movie schedules with a higher actual revenue.

References

Baranowski, P., Korczak, K., & Zając, J. (2020). Forecasting cinema attendance at the movie show level: Evidence from poland. *Business Systems Research: International journal of the Society for Advancing Innovation and Research in Economy*, *11*(1), 73–88.

Bardadym, V. A. (1996). Computer-aided school and university timetabling: The new wave. In E. Burke & P. Ross (Eds.), *Practice and theory of automated timetabling* (pp. 22–45). Springer Berlin Heidelberg.

Bass, F. M. (1969). A new product growth for model consumer durables. *Management Science*, *15*(5), 215–227.

Carriere, T., & Kariniotakis, G. (2019). An integrated approach for value-oriented energy forecasting and data-driven decision-making application to renewable energy trading. *IEEE transactions on smart grid*, *10*(6), 6933–6944.

Changyong, F., Hongyue, W., Naiji, L., Tian, C., Hua, H., Ying, L., et al. (2014). Log-transformation and its implications for data analysis. *Shanghai archives of psychiatry*, *26*(2), 105.

Chen, R. M., & Shih, H. F. (2013). Solving university course timetabling problems using constriction particle swarm optimization with local search. *Algorithms*, *6*(2), 227–244.

Chen, Y., Bayanati, M., Ebrahimi, M., Khalijian, S., et al. (2022). A novel optimization approach for educational class scheduling with considering the students and teachers' preferences. *Discrete Dynamics in Nature and Society*, *2022*.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*(1), 101–111.

Ding, Q., & Niu, Z. (2013). Optimizing and scheduling of super large-scale seawater reverse osmosis desalination system. *2013 10th IEEE International Conference on Control and Automation (ICCA)*, 705–711.

Eliashberg, J., Hegie, Q., Ho, J., Huisman, D., Miller, S. J., Swami, S., Weinberg, C. B., & Wierenga, B. (2009). Demand-driven scheduling of movies in a multiplex. *International Journal of Research in Marketing*, *26*(2), 75–88.

Grand View Research. (2021). *Middle East Movies Entertainment Market Size, Share Trends Analysis Report By Product (Movies, Music Videos), By Country, And Segment Forecasts, 2021 - 2028.*

Hanson, S. (2019). *Screening the World*. Springer.

Iniestra, J. G., López, E. A., María del Pilar, & Gorina, N. V. (2006). Optimal allocation of movies to screens in movie theaters. *IIE Annual Conference.Proceedings*, 1–5.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Lavoie, S., Minoux, M., & Odier, E. (1988). A new approach for crew pairing problems by column generation with an application to air transportation. *European Journal of Operational Research*, *35*(1), 45–58.

Lawitsanon, P., Hanthanunchai, K., Chanachanchai, N., Mahanin, S., & Polvichai, J. (2022). Improving the movie showtime scheduling problem by integrated artificial intelligence techniques. *2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 1–6.

Lee, K., Park, J., Kim, I., & Choi, Y. (2018). Predicting movie success with machine learning techniques: Ways to improve accuracy. *Information Systems Frontiers*, *20*, 577–588.

Leem, S., Oh, J., & Moon, J. (2023). Towards an effective over-the-top platform service: A machine learning approach for box office analysis. *2023 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 413–416.

Motion Picture Association. (2022). *2021 THEME Report*. https://www.motionpictures.org/wp-content/uploads/2022/03/MPA-2021-THEME-Report-FINAL.pdf

Muñoz, M., Pineda, S., & Morales, J. (2022). A bilevel framework for decision-making under uncertainty with contextual information. *Omega*, *108*, 102575.

National Science and Media Museum. (n.d.). A very short history of cinema. https://www.scienceandmediamuseum.org.uk/objects-and-stories/very-short-history-of-cinema

Samiuddin, J., & Haq, M. A. (2019). A novel two-stage optimization scheme for solving university class scheduling problem using binary integer linear programming. *Operations Management Research*, *12*(3-4), 173–181.

Stratigakos, A., Camal, S., Michiorri, A., & Kariniotakis, G. (2022). Prescriptive trees for integrated forecasting and optimization applied in trading of renewable energy. *IEEE Transactions on Power Systems*, *37*(6), 4696–4708.

Swami, S., Eliashberg, J., & Weinberg, C. B. (1999). Silverscreener: A modeling approach to movie screens management. *Marketing Science*, *18*(3), 352–372.

Tang, Z., & Dong, S. (2021). A total sales forecasting method for a new short life-cycle product in the pre-market period based on an improved evidence theory: Application to the film industry. *International Journal of Production Research*, *59*(22), 6776–6790.

Wen, X., Sun, X., Sun, Y., & Yue, X. (2021). Airline crew scheduling: Models and algorithms. *Transportation Research Part E: Logistics and Transportation Review*, *149*, 102304.

Yazdani, M., Naderi, B., & Zeinali, E. (2017). Algorithms for university course scheduling problems. *Tehnicki vjesnik/Technical Gazette*, *24*.

Zhang, X., Hou, G., & Dong, W. (2017). Modelling movie attendance with seasonality: Evidence from china. *Applied Economics Letters*, *24*(19), 1351–1357.

Zhou, Y., Li, Q., Yue, X., Nie, J., & Guo, Q. (2022). A novel predict-then-optimize method for sustainable bike-sharing management: A data-driven study in china. *Annals of Operations Research*, 1–33.

Appendix

A Background Information

A.1 Changes in the movie exhibition business

As explained by a professional in the movie exhibition business, the sector has undergone some substantial changes during the last 15 years (M. Groen, personal communication, May 12, 2023). Roughly 10 years ago, movie theatres started updating their 35mm film projectors to digital film projectors. When using the 35mm film projector, each screen would (preferably) only show one movie because these 35mm films were heavy, fragile, and expensive. Distributors would pay the costs of these 35mm films. To smoothen the transition to digital film projectors, distributors had to pay a virtual print fee (VPF) to movie theatres for every digital copy they distributed to them. This way the distributors helped paying for these much more expensive digital projectors. This however still meant that movie theatres most often only had one digital copy of a movie available, such that unique movies were still not shown with overlap. In the recent years, this VPF is no longer in use.

The COVID-19 pandemic created another big change in the business. Before, movies were typically shown in the cinema exclusively for around four months. This period is called the window of a movie. However, due to the closer of movie theatres because of governmental restrictions, movies became available on streaming platforms much sooner. As a lasting result, the window of a movie is nowadays around two to three months. Another consequence is that the agreements between movie theatres and distributors have become less strict or long lasting, allowing the movie theatres to have more scheduling freedom.

B Tables and Figures

Table B1. Sets, parameters and variables

Sets	
$\mathcal{T}$	set of time periods
$\mathcal{T}^m$	set of time periods that movie m can start
$\hat{\mathcal{T}}^m$	set of time periods that movie m is not desired to start
$\mathcal{T}^m_{req}$	set of time periods that movie m is required to have minimum η_m showings
$\mathcal{T}^\omega$	set of time periods in day ω
M	set of movies
M^t	set of movies that can be shown during time period t
M^s	set of movies that can be shown on screen s
M^g	set of movies that have genre g
M^l	set of movies that are shown in language l
S	set of screens
S^m	set of screens on which movie m can be shown
S^e	set of screens with cinema type e
S^r	set of screens in area r
K^s	set of seat types in room s
$\mathcal{G}$	set of movie genres
L	set of movie languages
R	set of areas in the movie theatre
W	set of days in the week
E	set of cinema types
V	set of nodes in network G
A	set of arcs in network G
V^s	set of nodes in network G^s corresponding to screen s
A^s	set of arcs in network G^s corresponding to screen s
P^s	set of paths on network G^s corresponding to screen s
Parameters	
δ_m	duration of movie m

o_m	commercial time of movie m
c_{sk}	capacity of seat type k in room s
p_s	duration of cleaning room s
$\hat{g}$	minimum number of different genres that must be shown in time interval of length τ_g
$\hat{l}$	minimum number of different languages that must be shown in time interval of length τ_l
γ_{start}	maximum amount of movies that can start in time interval of length τ_{start}
γ_{end}	maximum amount of movies that can end in time interval of length τ_{end}
f_{rt}	maximum on the total capacity of movie rooms in S^r ending a screening at time t and the total capacity of movie rooms in S^r starting screenings at time $t+1$
$\bar{s}_{me}$	maximum number of daily showings of movie m on cinema type e
$\hat{\tau}$	time interval in which at least one movie must start
d_{mtk}	expected demand for movie m starting in time period t and seat type k
w_a	weight of arc a
w_p	weight of path p
ρ_k	weight of seat type k
λ_i	weight of penalty i
Decision variables	
x_p^s	whether path p is selected for screen s
Auxiliary variables	
σ	number of sessions starting in undesired starting time periods
β_t	whether in time interval $[t, t+\hat{\tau}-1]$ are no movies starting
ζ_{ms}^{ω}	whether movie m is showing on screen s during day ω
a_{gt}	whether a movie with genre g is showing in time interval $[t, t+\tau_g-1]$
β_t^g	counting the number of genres in time interval $[t, t+\tau_g-1]$ below the desired level
a_{lt}	whether a movie with language l is showing in time interval $[t, t+\tau_l-1]$
β_t^l	counting the number of languages in time interval $[t, t+\tau_l-1]$ below the desired level

Table B2. Overview Movies per week

Movie	week 18/08/2022 - 24/08/2022	week 25/08/2022 - 31/08/2022
HO00009079	x	x
HO00009195	x	x
HO00009117	x	x
HO00009113	x	x
HO00009115	x	x
HO00009339		x
HO00008890	x	x
HO00009334	x	x
HO00009042		x
HO00009279	x	
HO00009332	x	
HO00009274	x	
HO00009293	x	x
HO00008989	x	
HO00009331		x
HO00009119	x	x
HO00009320		x
HO00009294	x	x
HO00008983	x	x
HO00009304	x	x
HO00009116		x
HO00009338	x	
HO00009271	x	x
HO00009326		x
HO00009284	x	
HO00009286		x
HO00009288	x	x
HO00009253	x	
HO00009238	x	
HO00008779	x	
HO00009292	x	
HO00009313		x
HO00009316	x	
HO00008982	x	x
HO00007665	x	x
HO00009287	x	
HO00009114		x

Table B3. Overview commercial time

15 minutes	**20 minutes**
HO00008983	HO00009079
HO00009286	HO00009195
HO00008779	HO00009117
HO00007665	HO00009113
	HO00009115
	HO00009339
	HO00008890
	HO00009334
	HO00009042
	HO00009279
	HO00009332
	HO00009274
	HO00009293
	HO00008989
	HO00009331
	HO00009119
	HO00009320
	HO00009294
	HO00009304
	HO00009116
	HO00009338
	HO00009271
	HO00009326
	HO00009284
	HO00009288
	HO00009253
	HO00009238
	HO00009292
	HO00009313
	HO00009316
	HO00008982
	HO00009287
	HO00009114

Note. Only the movies that show between 18/08/2022 and 31/08/2022 are displayed.

Table B4. Overview distributor restrictions

Movie	**Minimum afternoons**	**Minimum evenings**	**Screen requirements**	**Minimum daily showings**
week: 18/08/2022 - 24/08/2022				
HO00009334	1	1	must at least show on IMAX	-
HO00009294	-	-	-	10
week: 25/08/2022 - 31/08/2022				
HO00009331	-	2	-	-
HO00009116	-	-	exclusive on screen 1 and 2	18
HO00009286	-	-	KIDS only	3

Note. Only the movies that show between 18/08/2022 and 31/08/2022 and have distributor restrictions are displayed; the minimum afternoons and evenings are per week; afternoon time is between 12:00 and 18:00; evening time is between 18:00 and 00:00.

Table B5. Overview daily requirements on number of screenings per room type

Movie	$\mathbf{max_{KIDS}}$	$\mathbf{max_{IMAX}}$	$\mathbf{max_{4DX}}$	$\mathbf{max_{theatre}}$
week: 18/08/2022 - 24/08/2022				
HO00009117	-	-	3	6
HO00009115	-	5	2	7
HO00008890	9	-	-	-
HO00009334	-	3	-	-
HO00009293	-	-	-	1
HO00008983	7	-	-	-
HO00009284	1	-	-	-
HO00008982	1	-	-	4
HO00007665	1	-	-	1
HO00009287	1	-	-	-
week: 25/08/2022 - 31/08/2022				
HO00009113	1	-	-	-
HO00009115	-	-	-	2
HO00008890	5	-	-	-
HO00009042	1	-	-	1
HO00009293	2	-	-	-
HO00009294	1	-	-	-
HO00008983	6	-	-	-
HO00009116	-	5	4	8
HO00009271	-	-	-	3
HO00009286	6	-	-	-
HO00009313	-	-	-	1
HO00008982	-	-	-	1
HO00007665	-	-	-	1

Note. We do not provide a minimum or maximum if the room type is not applicable to a movie.

Table B6. OLS regression results

	weeks 30/06/2022 to 17/08/2022		**weeks 30/06/2022 to 24/08/2022**	
Variable	**coefficient**	**P > |t|**	**coefficient**	**P > |t|**
Intercept	0.2633	0.019	0.3248	0.004
HO00008779	0.750	0.000	0.625	0.000
HO00009211	0.179	0.358	0.146	0.456
HO00009215	-0.437	0.076	-0.644	0.010
HO00008989	0.178	0.025	0.118	0.132
HO00007665	0.241	0.001	0.151	0.023
HO00008983	-0.208	0.072	-0.155	0.148
HO00009156	-0.079	0.607	0.042	0.782
HO00008998	0.169	0.342	0.092	0.612
HO00009042	0.355	0.000	0.305	0.000
HO00009189	0.999	0.000	1.111	0.000
HO00008995	-0.280	0.136	-0.244	0.197
HO00009045	2.049	0.000	2.197	0.000
HO00009113	-0.070	0.089	-0.061	0.124
HO00008951	1.049	0.000	0.979	0.000
HO00009109	0.062	0.764	-0.044	0.829
HO00009016	0.230	0.450	0.197	0.525
HO00008777	0.550	0.011	0.459	0.035
HO00009073	-0.327	0.075	-0.327	0.079
HO00009191	-0.392	0.042	-0.392	0.045
HO00009208	-0.627	0.004	-0.675	0.002
HO00009200	-0.925	0.000	-0.967	0.000
HO00009145	-0.747	0.000	-0.662	0.000
HO00009209	0.852	0.003	0.888	0.002
HO00008961	0.251	0.342	0.244	0.371
HO00008982	0.031	0.345	0.032	0.308
HO00009190	0.250	0.018	0.341	0.001
HO00009192	-0.525	0.000	-0.406	0.000
HO00009195	1.495	0.000	1.575	0.000
HO00009203	-1.934	0.000	-1.673	0.000
HO00009205	-0.738	0.015	-0.747	0.014
HO00009120	0.060	0.784	0.012	0.957
HO00009202	-0.033	0.869	-0.019	0.928
HO00009140	-0.234	0.034	-0.172	0.099
HO00009161	-0.338	0.000	-0.385	0.000
HO00009224	-0.235	0.117	-0.132	0.379
HO00009197	-0.028	0.892	-0.078	0.707
HO00009079	0.729	0.000	0.853	0.000
HO00009255	-0.438	0.032	-0.380	0.064
HO00009273	-0.419	0.021	-0.453	0.012

Continued on next page

Table B6 – *Continued from previous page*

	weeks 30/06/2022 to 17/08/2022		**weeks 30/06/2022 to 24/08/2022**	
Variable	**coefficient**	**P > |t|**	**coefficient**	**P > |t|**
HO00009272	0.116	0.437	0.085	0.572
HO00009246	-1.196	0.000	-1.199	0.000
HO00009008	0.663	0.001	0.569	0.005
HO00009267	-0.191	0.122	-0.151	0.221
HO00009223	0.805	0.000	0.864	0.000
HO00008890	-0.374	0.000	-0.266	0.003
HO00009242	-0.017	0.881	0.040	0.730
HO00009282	-0.483	0.001	-0.567	0.000
HO00008890	0.000	0.000	0.000	0.000
HO00009214	0.540	0.000	0.437	0.000
HO00009274	-0.156	0.383	-0.277	0.134
HO00009025	1.481	0.000	1.296	0.000
HO00009262	-0.110	0.740	-0.050	0.880
HO00009254	0.375	0.054	0.292	0.136
HO00009115	-0.292	0.001	-0.339	0.000
HO00009275	-0.575	0.000	-0.490	0.000
HO00009288	0.591	0.000	0.764	0.000
HO00009281	-0.671	0.000	-0.638	0.000
HO00009247	0.365	0.000	0.228	0.001
HO00009119	0.316	0.011	0.473	0.000
HO00009292	-0.456	0.000	-0.421	0.000
HO00009284	0.119	0.100	-0.023	0.794
HO00009287	0.289	0.098	0.247	0.106
HO00009117	0.080	0.246	0.122	0.122
HO00009302	-2.064	0.000	-1.999	0.000
HO00009279	-0.403	0.001	-0.165	0.112
HO00009305	-0.221	0.199	-0.278	0.107
HO00009253	0.524	0.000	0.628	0.000
HO00009316	0.000	0.305	-0.740	0.000
HO00009304	0.000	0.024	-0.249	0.001
HO00009293	0.000	0.002	0.184	0.060
HO00009334	0.000	0.060	-0.458	0.011
HO00009271	0.000	0.018	0.532	0.000
HO00009332	0.000	0.261	0.106	0.610
HO00009338	0.000	0.000	-0.106	0.492
HO00009238	0.000	0.143	0.151	0.388
HO00009294	0.000	0.001	0.321	0.002
HO00009286	0.000	0.002	0.000	0.000
HO00009116	0.000	0.078	0.000	0.000
HO00009114	0.000	0.038	0.000	0.000
HO00009320	0.000	0.001	0.000	0.232
HO00009313	0.000	0.001	0.000	0.406

Continued on next page

Table B6 – *Continued from previous page*

	weeks 30/06/2022 to 17/08/2022		**weeks 30/06/2022 to 24/08/2022**	
Variable	**coefficient**	**P > |t|**	**coefficient**	**P > |t|**
HO00009326	0.000	0.025	0.000	0.153
HO00009331	0.000	0.038	0.000	0.777
HO00009339	0.000	0.163	0.000	0.000
$I_{h=0}$	0.170	0.012	0.218	0.001
$I_{h=1}$	-0.315	0.032	-0.306	0.037
$I_{h=2}$	-0.327	0.343	-0.357	0.299
$I_{h=3}$	0.000	0.000	0.000	0.129
$I_{h=4}$	0.000	0.406	0.000	0.040
$I_{h=5}$	0.000	0.023	0.000	0.549
$I_{h=6}$	0.000	0.026	0.000	0.011
$I_{h=7}$	0.000	0.096	0.000	0.000
$I_{h=8}$	0.000	0.194	0.000	0.318
$I_{h=9}$	-1.346	0.001	-1.358	0.001
$I_{h=10}$	-0.913	0.000	-0.946	0.000
$I_{h=11}$	-0.654	0.000	-0.664	0.000
$I_{h=12}$	-0.564	0.000	-0.568	0.000
$I_{h=13}$	-0.272	0.000	-0.272	0.000
$I_{h=14}$	0.060	0.236	0.073	0.135
$I_{h=15}$	0.142	0.006	0.150	0.002
$I_{h=16}$	0.290	0.000	0.284	0.000
$I_{h=17}$	0.347	0.000	0.382	0.000
$I_{h=18}$	0.619	0.000	0.634	0.000
$I_{h=19}$	0.745	0.000	0.758	0.000
$I_{h=20}$	0.831	0.000	0.859	0.000
$I_{h=21}$	0.821	0.000	0.812	0.000
$I_{h=22}$	0.485	0.000	0.549	0.000
$I_{h=23}$	0.396	0.000	0.428	0.000
I_{monday}	-0.196	0.000	-0.197	0.000
$I_{tuesday}$	-0.275	0.000	-0.256	0.000
$I_{wednesday}$	-0.278	0.000	-0.256	0.000
$I_{thursday}$	0.003	0.938	0.016	0.709
I_{friday}	0.203	0.000	0.232	0.000
$I_{saturday}$	0.535	0.000	0.581	0.000
I_{sunday}	0.523	0.000	0.556	0.000
$I_{holiday}$	0.276	0.000	0.274	0.000
$METER$	0.000	0.064	0.000	0.000
$RATING$	0.347	0.000	0.334	0.000
$LANG_{turkish}$	-0.070	0.089	-0.061	0.124
$LANG_{russian}$	0.540	0.000	0.437	0.000
$LANG_{japanese}$	0.115	0.369	0.066	0.560
$LANG_{arabic}$	-0.941	0.000	-1.024	0.000
$LANG_{india}$	0.332	0.000	0.243	0.000

Continued on next page

Table B6 – *Continued from previous page*

	weeks 30/06/2022 to 17/08/2022		weeks 30/06/2022 to 24/08/2022	
Variable	**coefficient**	**P > |t|**	**coefficient**	**P > |t|**
$LANG_{tagalog}$	0.000	0.000	0.427	0.002
$LANG_{chinese}$	0.000	0.000	0.151	0.388
$LANG_{english}$	0.631	0.000	0.630	0.000
$LANG_{germany}$	0.365	0.000	0.228	0.001
$LANG_{korean}$	-0.456	0.000	-0.421	0.000
$GENRE_{horror}$	-0.088	0.271	-0.024	0.751
$GENRE_{thriller}$	-0.302	0.000	-0.236	0.002
$GENRE_{action}$	0.047	0.497	0.103	0.103
$GENRE_{animation}$	0.210	0.039	0.142	0.110
$GENRE_{comedy}$	0.547	0.000	0.527	0.000
$GENRE_{biographical}$	0.284	0.000	0.244	0.000
$GENRE_{drama}$	-0.747	0.000	-0.662	0.000
$GENRE_{adventure}$	0.445	0.000	0.534	0.000
$GENRE_{family}$	0.119	0.100	0.298	0.000
$GENRE_{crime}$	0.000	0.000	-0.249	0.001
$GENRE_{romance}$	0.000	0.000	0.106	0.610
$GENRE_{war}$	0.000	0.000	-0.106	0.492
$SEQUEL$	0.277	0.000	0.205	0.001
$ACTOR_{100}$	-0.141	0.064	-0.119	0.122
$ACTOR_{1000}$	-0.011	0.880	0.027	0.708
$DIRECT_{1000}$	0.031	0.345	0.032	0.308
$DIRECT_{5000}$	-0.467	0.000	-0.489	0.000
$DISTR$	-0.448	0.000	-0.407	0.000
$PRODUCT$	0.545	0.000	0.631	0.000
$RELEASE$	-0.109	0.000	-0.099	0.000
$TICKETS$	0.000	0.027	0.000	0.002
$COUNT_g$	-0.012	0.041	-0.018	0.002
$COUNT_{release}$	0.007	0.347	0.003	0.657
$COUNT_{pop}$	0.027	0.000	0.025	0.000
R^2	0.473		0.469	

Note. Columns 2 and 3 show the regression results for the data of weeks 30/06/2022 to 17/08/2022; columns 4 and 5 show the regression results for the data of weeks 30/06/2022 to 24/08/2022.

Table B7. Results Column Generation

	Objective value	**Total weight paths**	**Total penalty**	**Running time**
		week: 18/08/2022 - 24/08/2022		
Thursday	-33203.70	-38903.70	5700	1210
Friday	-29379.92	-45779.92	16400	1363
Saturday	-41820.80	-50820.80	9000	3599
Sunday	-29931.33	-53031.33	23100	3331
Monday	-18446.46	-35446.46	17000	1308
Tuesday	-22638.50	-34338.50	11700	1333
Wednesday	-16968.02	-33468.02	16500	1339
Aggregated	**-192388.73**	**-291788.73**	**99400**	**13483**
		week: 25/08/2022 - 31/08/2022		
Thursday	-40762.31	-45962.31	5200	950
Friday	-47353.12	-52053.12	4700	982
Saturday	-51143.01	-57643.01	6500	1013
Sunday	-55531.72	-61431.72	5900	1106
Monday	-30772.94	-36372.94	5600	1006
Tuesday	-31179.95	-36779.95	5600	1019
Wednesday	-26943.43	-34343.43	7400	948
Aggregated	**-283686.48**	**-324586.48**	**40900**	**7024**

Note. Running time is in seconds.

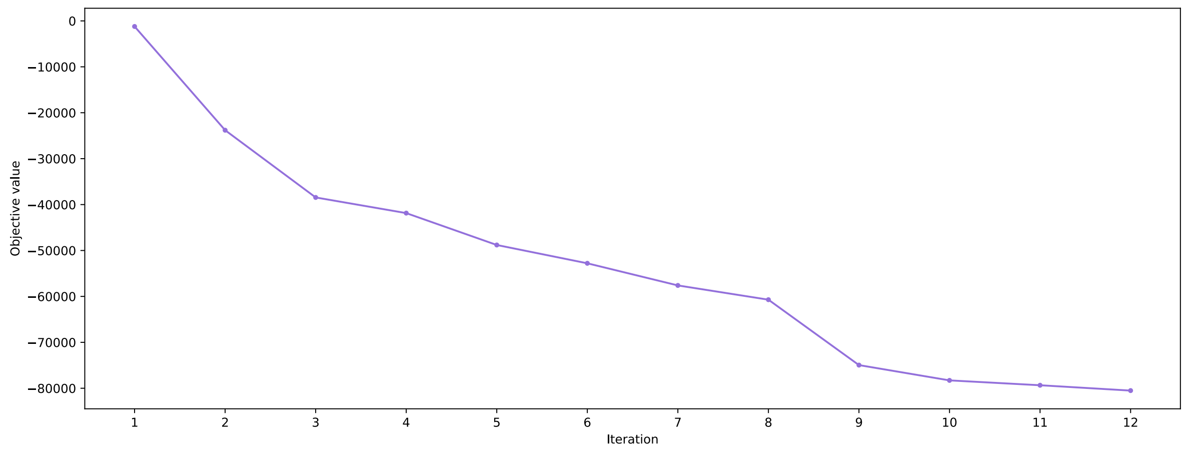


Figure B1. Large Instance Column Generation result linear relaxation per iteration, Sunday 28/08/2022

Table B8. Results ALNS

	Objective value	Total weight paths	Total penalty	Running time
		week: 18/08/2022 - 24/08/2022		
Thursday	-28832.62	-34532.62	5700	78
Friday	-31733.03	-37433.03	5700	76
Saturday	-43240.08	-49340.08	6100	80
Sunday	-40542.98	-46742.98	6200	75
Monday	-24759.50	-31059.50	6300	87
Tuesday	-25011.72	-30611.72	5600	76
Wednesday	-24695.48	-30595.48	5900	82
Aggregated	**-218815.41**	**-260315.41**	**41500**	**554**
		week: 25/08/2022 - 31/08/2022		
Thursday	-36820.64	-42120.64	5300	103
Friday	-41579.11	-47279.11	5700	82
Saturday	-45725.66	-51025.66	5300	73
Sunday	-46186.86	-51886.86	5700	70
Monday	-26994.59	-32594.59	5600	86
Tuesday	-27262.07	-32362.07	5100	74
Wednesday	-27104.98	-32904.98	5800	79
Aggregated	**-251673.91**	**-290173.91**	**38500**	**567**

Note. Running time is in seconds.

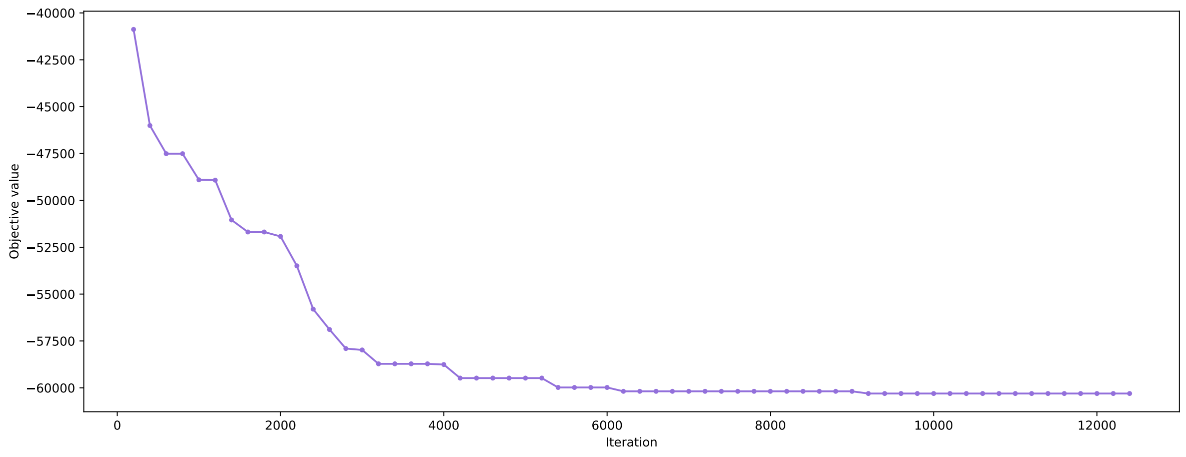


Figure B2. Large Instance ALNS objective value per iteration, Sunday 28/08/2022

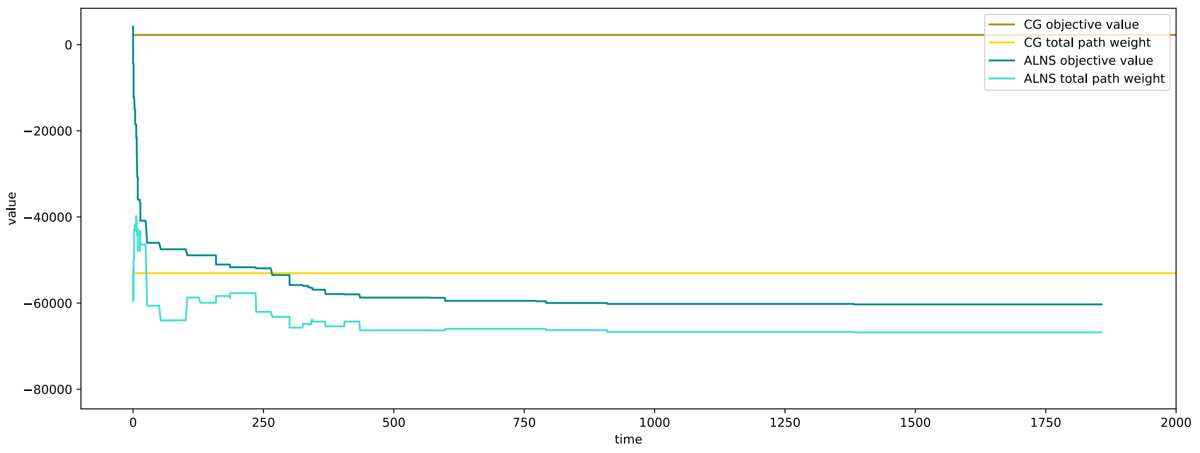


Figure B3. Comparison Column Generation and ALNS over time in seconds (partial), Sunday 28/08/2022

Table B9. Results large instance, Sunday 28/08/2022

	Objective value	**Total weight paths**	**Total penalty**	**Running time**
Column Generation	-75026.71	-80326.71	5300	30687
Adaptive Large Neighborhood Search	-60673.83	-66073.83	5400	1857

Note. Running time is in seconds.

C Schedules

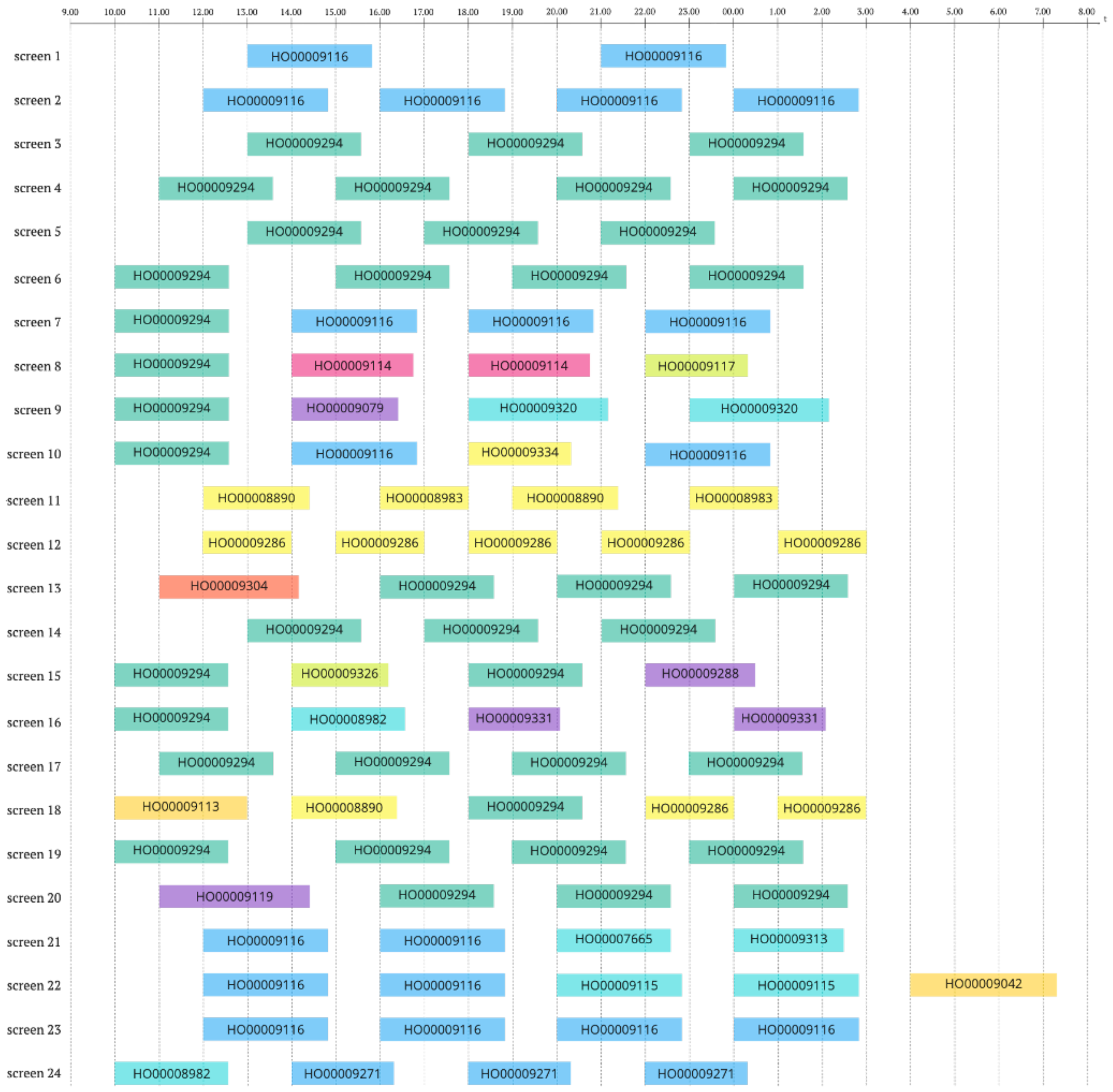


Figure C1. Schedule Column Generation Sunday 28/08/2022

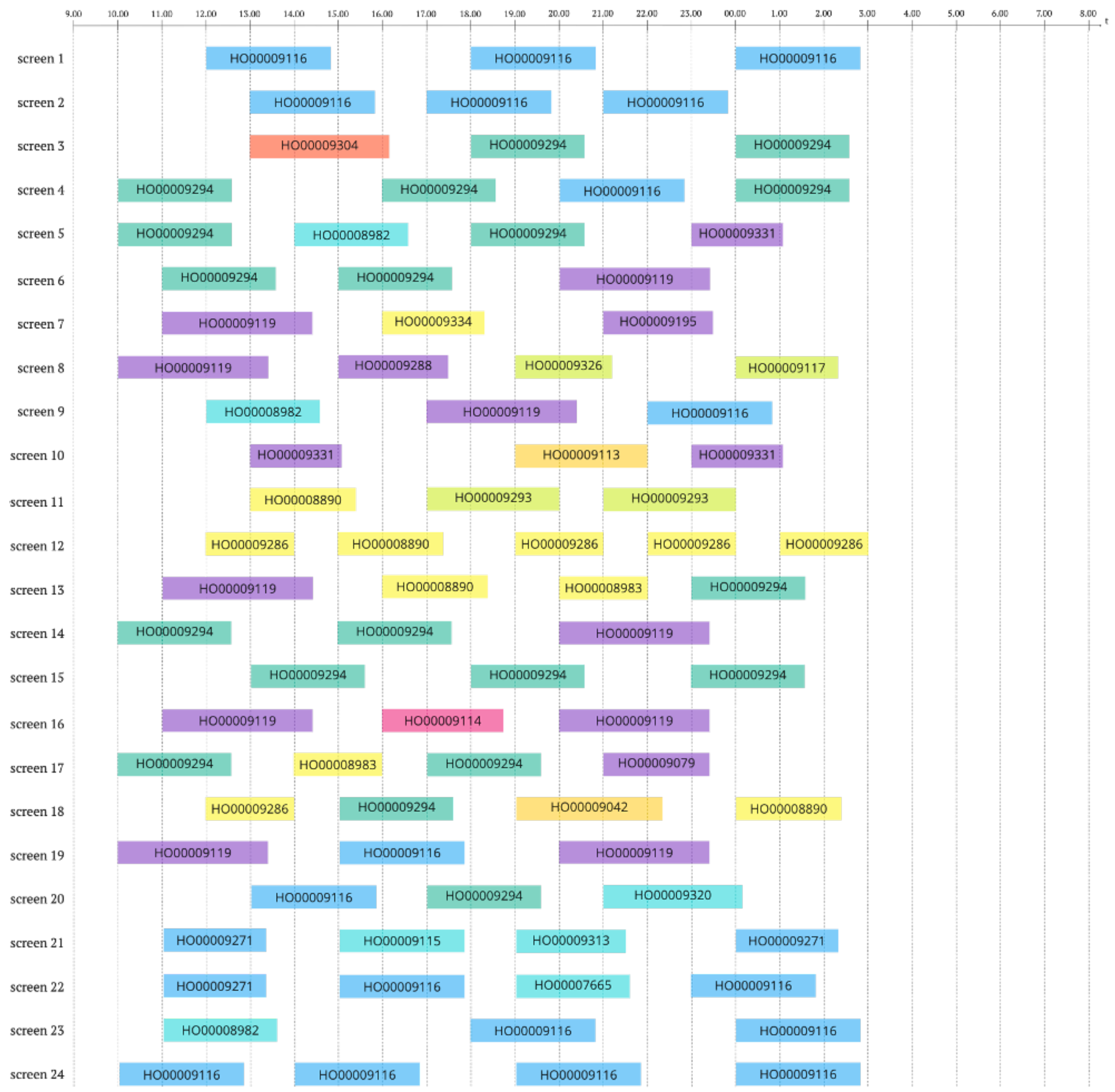


Figure C2. Schedule ALNS Sunday 28/08/2022

D Description programming code

In this section, a brief overview of the Python and Java code used in this thesis will be presented. The code adheres to the case study outlined in Chapter 5 and methodology outlined in Chapter 4. The Python Project contains three directories collecting the data and making the demand forecast. The Java Project contains four Packages importing instances and executing the described methods.

D.1 Python code

The directories *Data_collection* and *Data_preprocess* collect and prepare the data for our case study, respectively. The directory *Data_collection* contains files scraping websites such as IMDb to collection real time movie information and ratings. The directory *Data_preprocess* contains files analyzing the data and transforming the transaction dataset into a clean set of session data.

The directory *Attendance_forecast* tunes, trains and analyzes the Linear Regression model and Gradient Tree Boosting (GTB) model described in Section 4.1. It also contains the file *GTB_call_java* with the trained GTB models that is called from our Java code to update the demand forecast.

D.2 Java code

The package *movieNetwork* contains classes to construct the network presented in Section 4.2.1. Classes represent movies, screens and time periods, and other characteristics of the problem such as areas of the movie theatre, genres, languages, and screen types. Moreover, the classes representing nodes, arcs, paths and the network are expressed in this package.

Moreover, the package *SPP* contains classes constructing the Set Partitioning Problem, the Column Generation framework, the k-Shortest Path Problem, and the feasible initial solution. In the class generating the Set Partitioning Problem, the software package CPLEX is used.

Furthermore, classes to run the ALNS algorithm can be found in the package *ALNS*. The class *Framework* contains all steps of Algorithm 1. The classes containing all destroy, repair, and improvement operators are called from the class *Framework* to perform the operations on the current solution.

Lastly, the package *CaseStudy* contains classes to import the data, run the different methods, and analyze the solution. The instances described in Chapter 5 can be imported with time periods of 5 minutes, 15 minutes, or 1 hour. The classes *Forecast* and *ForecastAndSchedule* call the Python code to update the demand forecast based on the current schedule.