Optimizing the planning strategy of the judiciary using Approximate Dynamic Programming

A case study for commercial and administrative law
within the teams of Overijssel and Rotterdam

Irene C. H. Kuin
A thesis presented for the degree of
Master of Science in Applied Mathematics

Department of Stochastic Operations Research
University of Twente

December 2020

Preface

This report is the result of my Final Project within the master of Applied Mathematics at the University of Twente (specialization: Operations Research). The research was performed for the 'Raad voor de Rechtspraak' from April 2020 until December 2020. Due to the COVID-19 pandemic we (have) live(d) in, this research was executed from home.

First, I would like to thank my supervisor Richard Boucherie for providing me with all the feedback and new ideas during the project. Since I did not have all the correct prior knowledge and I had no one to ask my questions to from my home-office, I often got stuck and I really value that you were always there to help me out, despite your busy schedule. Besides, I would like to thank Aleida Braaksma for thinking along with the modeling of the Markov Decision Program; I really appreciated that you were willing to think along, even though you were not my first supervisor. Also, I would like to thank Richard as well as Erwin for giving me the opportunity to do this research for the 'Raad voor de Rechtspraak' and Anouk, Aleida, Clara and Erwin for taking place in the graduation committee and reading and evaluating my work.

Then, I would like to thank my supervisors at the 'Raad voor de Rechtspraak' Marleen Oele (until June 2020) and Anouk Bergmans (from July 2020) for their useful input as well. Anouk, I really value that you were always willing to think along with my research and helped me in my search for people that could contribute to my research (by sharing their knowledge). Likewise, I would like to thank everyone else from the 'Raad voor de Rechtspraak' that has helped me during this research, with a special thanks to the teams commercial law and administrative law of the courts situated in Rotterdam and Overijssel, as well as the project team 'Tijdige Rechtspraak'. It was a difficult period to get familiarized me with the world of the jurisdiction, but you helped me out with the needed Skype sessions along the way.

Last but not least, I would like to thank my family and friends for the support during this research. The combination of executing most of the research from home and the intensive rehabilitation after my knee surgery was sometimes a bit much for me. My closest friends and family supported me through the hard times and I am very grateful for that!

Irene Kuin,
Breda, December 2020

Abstract

In this study, we propose a method to develop a planning strategy for the planning of cases within the judiciary, within commercial law. For these cases, we schedule the hearing as well as the preparation of the hearing and the writing of the verdict, taking into account stochastic elements in the arrival of the cases as well as their duration. Our method is developed in an Approximate Dynamic Programming (ADP) framework that makes use of post-decision states and a rolling horizon. In this strategy we take into account the availability of the judges and legal assistants, as well as the specialisms and difficulties they can handle. Besides, we schedule one fold cases (that only need one judge) as well as multiple cases (for which three judges have to be scheduled). The results show that we are able to find solutions within a reasonable time for a toy-sized problem instance. Moreover, the results indicate that we are able to find better solutions using our planning strategy compared to a myopic policy. The results have been verified using our own data and therefore, the algorithm needs to be revised when the Rechtspraak can deliver adequate data, mainly on the duration of the preparation of a hearing and writing of the verdict. What is more, increasing the efficiency of the implementation of the algorithm such that larger problem instances can be solved, remains a topic for further research.

Keywords: jurisdiction, planning lawsuits, Approximate Dynamic Programming (ADP), Value Function Approximation (VFA)

Acronyms

RSP Resident Scheduling Problem. 6, 7

SAA Sample Average Approximation. 8, 29, 66

SDP Stochastic Dynamic Program. 3, 5, 9, 29, 30, 56, 57

SIP Stochastic Integer Program. 7, 29

SRCPSP Stochastic Resource-Constrained Project scheduling Problem. 5, 9

Contents

1 Introduction

This research is carried out in the name of the Rechtspraak. In Section 1.1 we discuss the emergence of the problem the Rechtspraak deals with as well as the scope of this research. In Section 1.2 we discuss the current planning- and scheduling process and in Section 1.3 we elaborate on how this report is structured.

1.1 Background

The Rechtspraak has an ongoing project called “Programma Tijdige Rechtspraak”. The objective of this project is to contribute to high customer satisfaction and access to justice. In order to achieve this goal, the program has been divided into three sub programs, namely:

1. Backlogs: The court has been dealing with backlogs for years. The exact size of the backlogs is unknown. Research should be done in order to determine the exact size of these backlogs and to determine how these backlogs can be eliminated.

2. Scheduling and planning: This sub-program researches the optimal way to schedule and plan, in order to save time.

3. Predictability: Many stakeholders in lawsuits are dissatisfied with the process in which they were involved. This dissatisfaction mainly involves: ignorance of the lead time of the process, lack of transparency about what happens during this lead time and the perception that the process elapses inefficiently.

The initial idea was to implement this program three years after it was released. This would be at the end of 2023. This planning will probably change due to the COVID-pandemid we live in. Due to this crisis, the courts and tribunals have been closed for a while. This meant that thousands of criminal cases are delayed [35]. This closure of the courts has led to the increase of cases in which a hearing has to be held. On the other hand, stocks of cases that could be finished by writing have shrunk enormously. Particularly in the latter case, it was be a challenge to ensure that no new stocks were created in the short term during the restart.

Since the schedulers and planners at the Rechtspraak do not use a way of planning/scheduling that is proven to be effective, different parties agree that problems can be solved by using a more efficient way of planning. First of all, time can be saved by creating and using a more efficient planning. Secondly, with a more accurate planning, realistic lead times can be established, which enlarges the transparency of the process, and thereby the satisfaction of the lawsuit stakeholders.

In the project “Programma Tijdige Rechtspraak” norms have been established for the different lead times one faces when entering into a lawsuit. Using a new planning- and scheduling algorithm, one could review these standards in order to determine the feasibility. The prosecutor prefers receiving a realistic lead time communicated *a priori*, to receiving a short one. In this project we limit ourselves to the Rechtbank and do not take the courts of justice into account. Within the Rechtbank, we focus on the jurisdictions administrative law (in Dutch: ‘bestuursrecht’) and commercial law (in Dutch: ‘handelsrecht’). When testing these models, we use the data from the courts situated in the province Overijssel as well as the court situated in Rotterdam and Dordrecht (which together form the team Rotterdam).

1.2 Current situation

Scheduling is defined as 'the allocation of judges and clerks at hearings, taking into account the allocated capacity in courtrooms'. With *planning* we refer to the planning of cases at these hearings. Within administrative law these two are clearly separated. The teams within commercial law allocate the clerks and judges at the same time as planning the cases. They do not use such an initial schedule.

For the *scheduling* part within administrative law, one has to classify the cases according to different components, such as:

1. Whether it is a one-fold or multiple case. A one-fold case only needs one judge, whereas for a multiple case the dossier must be read by three judges who all need to be present at the hearing.
2. Which team and/or specialism the hearing belongs to. Within a team, hearings can be handled in one or more areas of law or attention, such as finance and intellectual property. Judges can be part of more than one team.
3. Whether the case is media-sensitive. When one has to deal with media-sensitive cases, the assigned judge has to be able to handle the additional pressure from the outside.

In administrative law a schedule is made half a year in advance, based on historical data. This schedule states which hearings have to be handled at what time, based on the above mentioned components. In the current situation, 'hard to schedule'-hearings are scheduled first. The 'hard to schedule'-hearings involve: multiple hearings, training matters (i.e. a hearing in which a trainee judge is involved), detention cases (i.e. a hearing about someone in pre-trial detention) and summary proceedings. Afterwards, the other hearings are scheduled and subsequently by *planning*, cases are classified onto these hearings.

For the *planning* part of administrative law, the judges give their impediment dates. When the prosecutor(s) are not available on one of these days, the impediment days of the judges further in the future will be given based on which a new date for the hearing can be established. When the judge is not able to find a spot in his/her agenda, the case is sent back to the president of the certain team and a new judge will be assigned to the case. In both jurisdictions -administrative as well as commercial law- apart from the judge(s), a legal assistant (junior, average or senior, dependent on the case) also has to be assigned to a case. It is important to make the best possible combination of the judge(s) and legal assistant, taking into account which team they both belong to and what their availability is.

In commercial law, the team chairman determines the qualified judges for each case. Subsequently, the team chairman assigns the case to a judge based on the properties of a case (given the points above) and the available time in the agendas of the judges and legal assistants. Subsequently, the planners determine a date and time for the hearing based on this data. An additional property used for commercial law is the complexity of the case. This complexity can be classified into different categories, ranging from most complex to the least complex case.

1.3 Thesis outline

In the beginning of this thesis, we have given the abbreviations we use throughout this report. In Section 2 we give a literature review of both the planning and scheduling problem. Thereafter, in

Section 3 we give the context analysis, containing the steps in the current planning and scheduling process as well as the goal and scope of this research. In Section 4 we give the model design of our Integer Linear Program (ILP) for commercial law. We have programmed this model in Advanced Interactive Multidimensional Modeling System (AIMMS) and give an illustration on the results of this model in Section 5. Then, in Section 6 we give the model formulation of the Stochastic Dynamic Program (SDP) for commercial law. We discuss the results of the SDP in Section 7. Finally, we give the limitations of our research as well as recommendations for further research in Section 8 and we end with the conclusions and general recommendations in Section 9.

In Appendix A we give the original Dutch words for the law-related terms used in this report. In Appendix B we give an overview of the ILP used for commercial law. Some of the constraints of this ILP had to be rewritten in order to program them in AIMMS which we discuss in Appendix C. The results of that AIMMS model are subsequently given in Appendix D. Thereafter, in Appendix E we give the additions that have to be made to the ILP for commercial law in order to be valid for the planning problem of administrative law. Unfortunately, due to time restrictions, these additions have not been implemented and tested. We end with a couple of appendixes related to our Markov Decision Process (MDP). The pseudocodes we have given in the report in order to calculate the values for the basis functions had to be rewritten in constraints in order to calculate them using Python's Mixed Integer Programming (MIP)-package. Appendix F elaborates on these constraints. Thereafter, Appendix G gives the output of the first ten iterations (when executing 50 iterations) of the toy-sized problem. We end with the exact availability of the judges and legal assistants we have used when increasing the state space (and therefore dealing with a larger time horizon or more judges and legal assistants) in Appendix H.

2 Literature review

In this section we discuss how *similar* problems in literature related to planning and scheduling are addressed. To date (this section was written in May 2020) no literature exists on the optimization of planning and scheduling in the jurisdiction. In Section 2.1 we give the mathematical model for the classic assignment problem as well as the related Resource-Constrained Project scheduling Problem (RCPSP) and several variations. Moreover, other scheduling problem formulations are given in Section 2.1. In Section 2.2 we give a short overview of *similar* planning problem formulations taken from literature. Here, we focus on multi-appointment scheduling problems. We mostly discuss similar problem statements applied in healthcare, since we see various similarities between the two environments, such as the uncertainty in the duration of the hearing versus the uncertainty in the duration of the treatment, the nurse versus the judge, and the arrival of a patient versus the arrival of a case. Finally, in Section 2.3 we give a short conclusion of the literature review.

2.1 Scheduling problem

The scheduling problem can be seen as a variation of the well-known Assignment Problem (AP): the one-to-one problem of finding a matching between n tasks and n agents, the objective being to minimize the total costs of the assignments [38]. The mathematical model for the General Assignment Problem (GAP) in which each agent is assigned multiple tasks, is given in equation (2.1). Here, $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, c_{ij} is the cost of assigning agent i to task j, a_{ij} is the amount of agent i's capacity used if that agent is assigned to task j and b_i is the available capacity of agent i [57]. In our problem, n would represent the cases that have to be divided over m courts. The standard AP deals with deterministic durations, while we are dealing with uncertainty. Uncertainty can have two different sources, namely when durations are not known in advance or when there is uncertainty about the presence or absence of individual jobs. The second one is implemented in the model of Albareda-Sambola et al. [1]. In our problem however, we deal with both sources of uncertainty, where the second one does not occur that often. In literature, the GAP is solved by different approaches: from state-of-the-art metaheuristics to variable neighborhood search algorithms and from exact solution procedures to simple heuristic algorithms [57].

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{m} x_{ij} = 1, \quad j = 1, ..., n \\ & \sum_{j=1}^{n} a_{ij}x_{ij} \leq 1, i = 1, ..., m \\ & \qquad\qquad x_{ij} \in \{0, 1\} \end{aligned} \tag{2.1}$$

Multiple variations of this problem exist. We go deeper into the RCPSP by which Roland et al. [44] have formulated the Operating Room (OR) scheduling problem. The RCPSP then questions what date, operating room and starting time indication should be assigned to a set of surgeries [11]. Here, both the planning over several days and the scheduling on a single day are integrated in one model. The RCPSP can be defined as a combinatorial optimization problem. A combinatorial optimization problem is defined by a solution space and a subset of feasible solutions with an objective function. The RCPSP is Nondeterministic Polynomial time

(NP)-hard, which means the first results are time-consuming to obtain. Hence, one has to treat the problem meta-heuristically, in particular towards a Genetic Algorithm (GA) approach [44]. The RCPSP assumes each task is completed in time by applying a fixed amount of resource in each time period, while we are dealing with stochastic activity durations [42]. The Stochastic Resource-Constrained Project scheduling Problem (SRCPSP) is a variation on the RCPSP that uses stochastic activity durations [16]. In Section 2.1.1 we go into detail about the SRCPSP and in Section 2.1.2 we give variations of this as could be applied to our problem.

2.1.1 SRCPSP

For the problem definition of the SRCPSP we refer to the article by Creemers [15] in which he minimizes the expectation makespan over a given class of policies. In contrast to the basic type of the RCPSP, the SRCPSP has received only little attention in literature [15]. In this review we discuss the stochastic models reviewed in the paper of Habibi et al. [27]. Generally, a project can be described by a directed acyclic graph $G = (V, E)$ that has a set of nodes $V = \{0, 1, 2, .., n\}$ and a set of arcs $E = \{(i, j)|i, j \in V\}$ [14]. The nodes represent the project activities, whereas the arcs connecting the nodes represent precedence relations [14]. In the SRCPSP the deterministic durations of the RCPSP are replaced by random durations $c_{ij} : (i, j) \in A$. Here, c_{ij} is the stochastic duration time of the activity represented by (i, j) in A [34].

Creemers et al. [45] present three different ways to handle the uncertainty in the SRCPSP:

1. Robust or proactive scheduling, in which the decision maker may try to find a schedule that can tolerate minor deviations from the predicted values for the activity durations, for example as proposed by Artigues et al [2]. Furthermore, Vonder et al. [52] reschedule the remaining activities such that the sum of deviations of the new finishing times from the original ones is minimized.

2. Reactive scheduling, which iteratively changes an initial schedule in order to adjust it to the realizations of the underlying stochastic variables, for example as proposed by Deblaere et al. [18].

3. Stochastic scheduling, where no initial schedule is built before the execution of the project, for example as proposed by Raifiee et al. [42].

Creemers [15] and Creemers [14] use acyclic Phase-Type (PH) distributions to model activity durations and to match the first two moments of the activity duration distributions [15]. Since in most cases the *true* distribution of the activities is unknown, it is often assumed that the duration of an activity follows a beta, uniform or Gaussian distribution. These distributions only allow to match the first two moments of the true duration distribution, whereas PH distributions can match almost any distribution with arbitrary precision [15].

Because the SRCPSP is known to be NP-hard, most researchers have focused on developing heuristic solution methods. Rostami et al. [45] developed a two-phase metaheuristic that consists of a Greedy Randomized Adaptive Search Procedure (GRASP) and a GA to find the optimal procedure [15]. Ke and Liu [34] also used the GA to settle the SRCPSP. The aim of their research was to maximize the probability of the total cost not exceeding the budget. They had to find a solution under the constraint that the probability of finishing the project before the due date should be larger than or equal to a predetermined confidence level. Apart from these heuristic methods, Creemers et al. [14] [17] used backward SDP recursion to solve different kinds of SRCPSPs exactly .

2.1.2 Variations

The standard problem cannot cover all situations that may occur in practice, which is why many researchers have developed more general project scheduling problems using the standard RCPSP as starting point [30] [29].

A variation is the Multi-Mode Resource-Constrained Project scheduling Problem (MRCPSP). Here, multiple modes exist and each one shows a feasible way to combine a duration and resource request that allows to accomplish the underlying activity. This may come in handy when different workers work at a different speed, as we also encounter at the Rechtspraak. In particular, to implement the multi-mode extension, the following assumptions have to be made [24]:

1. The durations of the activities depend on the mode, i.e. we have $D_i = D_{im}(m \in M_i)$ where M_i is the finite set containing execution modes;
2. To each mode $m \in M_i$ a cost C_{im} is assigned;
3. The resource consumptions r_{ik} can depend on the mode: $r_{ik} = r_{ik}(m)$ for $m \in M_i$.

Moreover, due dates can be added to the RCPSP, which can only be exceeded at some penalty cost [30]. These due dates are important when multiple projects are planned simultaneously, which could also be the case in our problem. Another addition to the RCPSP are the time-switch constraints of Yang and Cheng [55]; the planning horizon is divided into cycles of work and rest time windows in which an activity can only start in a work window. This way working times, such as Monday through Friday, can be captured. When multiple projects have to be scheduled, several different objectives can be used. Since every project is associated with a due date, the minimization of weighted tardiness can be used as objective [30].

Lastly, Vanhoucke [53] has defined a set of time windows for each activity. In this set, the objective function minimizes penalties that are caused by executing activities outside their time windows. As we have seen in the "Programma Tijdige Rechtspraak", not only the speed of the process is seen as important, but the knowledge of a representative lead time is also important. Hence, when making a schedule that not only reduces the waiting time, but also endures the stated norms for time intervals, the clients will be more satisfied by the process.

2.1.3 Other problem formulations

In OR-scheduling and planning literature, a lot of different methodologies use a specific analysis and solution technique [26]. Mixed integer programming, simulation and heuristic algorithms are examples of frequently used methodologies. Furthermore, researchers can leverage the power of Constraint Programming (CP) in order to create mathematical representations of existing constraints in the problem [26]. By using CP, different values in the solution process can be evaluated. When one aims at obtaining satisfactory results in a short time, heuristic methods can be used.

Besides the RCPSP, another problem with similar constraints is the Resident Scheduling Problem (RSP). This is a special case of multi-period staff assignment problem where individuals are assigned to a task over multiple periods of time [21]. An example of this from healthcare is the problem of scheduling the staff to Outpatient clinic (OC)s in a physical medicine and rehabilitation department. The scheduling of outpatient clinics addresses:

a) determining the number of OCs that are in service during the weekdays;

b) assigning each OC to one of the senior academic staff;

c) assigning each resident to one of the OCs.

In order to solve this RSP, researches apply several techniques and methods, like MIP, CP, Goal Programming (GP) and meta-heuristics [25].

2.2 Planning problem

When a schedule has been made, the planning problem remains to be solved. In this section we discuss planning problem formulations with stochastic durations and multiple dependent stages. Different methods are used to solve these problems. Linear programming methods as well as MDPs are used in literature, where MDPs can be considered as a standard method for planning under uncertainty [43]. Since normal linear programming methods do not incorporate uncertainty, one needs to use an optimization algorithm that incorporates sampling of scenarios from the scenario tree. Singh et al. [49] do this by integrating the Dantzig-Wolfe decomposition and Homem-de-Mollo et al. [31] implement Stochastic Dual Dynamic Programming (SDDP). Escudero et al. [19] explicitly split the problem in different stages, for which they also used a decomposition algorithm in order to solve the problem, this time a Lagrangean-based one. Song and Huang [50] and Halman and Nannicini [28] propose a different model to formulate a stochastic planning problem with multiple stages. In Section 2.2.1 we examine the Multi-Appointment Scheduling Problem in Healthcare (MASPH). In the subsequent sections we focus on the used mathematical formulations and solvers: in Section 2.2.2 we discuss the two-stage problem formulation and in Section 2.2.3 we discuss the Markov Decision Process problem formulation.

2.2.1 Multi-appointment scheduling problem in healthcare

The MASPH is defined as the problem of scheduling patients who need appointments on a subset of hospital resources by bringing together all stakeholders in the scheduling process and subsequently optimizing on specific resources from a centralized perspective [37]. This problem occurs in different contexts that have multi-day fixed patterns: rehabilitation, examination and radiotherapy [48] [13]. Despite its importance, the subject of scheduling multiple (recurring) appointments is not thoroughly studied in any of the fields; existing studies on appointment scheduling problems mainly focus on single-stage service systems [56] [12].

Pérez et al. [41] formulate the problem of scheduling patients and resources in nuclear medicine clinics using following three approaches:

1. Offline, where the problem is solved on a day-to-day basis and it is assumed that all requests for the day are known in advance. This problem is modelled using Integer Programming (IP).

2. Online, where requests arrive sequentially one at a time and scheduling decisions are made when the requests arrive. To be able to solve this problem, the IP for offline scheduling is modified in order to schedule procedure requests one at a time. For this modification, the objective function now minimizes the waiting time for the patient instead of the number of patients scheduled on a given day. The decision variables and constraints are not modified.

3. Stochastic online, in which possible future requests are taken into account. This problem is formulated as a two-stage Stochastic Integer Program (SIP). In the first stage it is decided when to schedule the request at hand and which resources to use. This model

is similar to the model used for the online problem, in which, again, only the objective function differs. This time, the objective function not only contains the waiting time for the current patient, but also the expected value of the model's second stage objective function. The second stage creates the patient schedule by generating scenarios using Monte-Carlo simulations. These scenarios are defined as groups of possible requests that could arrive after the current patient request and that also share the preferred day for an appointment. This problem is solved every time a request arrives at the clinic by using the Nuclear Medicine Stochastic Online Scheduling (NMSOS) algorithm. Using this algorithm more people can be served compared to the number that can be helped using the Fixed Resource (FR) scheduling algorithm, especially under high demand.

2.2.2 Two-stage problem

The approach of dividing the stochastic online problem in two stages is used often to solve MASPHs [48] [56] [20]. The first step in this method is dividing the daily service capacity of the doctor into appointment slots and the second is to determine the number of patients that can fit into each slot [20]. The objective is to determine a job allowance for each customer in the first stage, so as to minimize the total expected weighted costs for customers' waiting times and service providers' idle times over multiple stages [56]. The constraints of the model can then be categorized into various subsections: resource constraints, sequencing and optional activities, linking constraints, recurring activities, stable activity starting times and general treatment starting time constraints [54]. Subsequently, the Sample Average Approximation (SAA) approach can be applied in order to transform the stochastic program into a two-stage program [56]. Because of the complexity, exact methods are rarely used in order to solve these two-stage problems. The popular methodologies are metaheuristics and multi-agent methods [37]. For example, Fan et al. [20] use local search to find the optimal solution since the objective function is submodular. Zhou and Yue [56] use an L-shaped algorithm in order to solve their two-stage model. For comprehensive information on this algorithm, we refer to their article.

2.2.3 Markov Decision Processes

Another method that is proposed in literature is the method of formulating MDPs [46] [4] [32]. The MDPs are mostly solved by Approximate Dynamic Programming (ADP) [46] [4]. Sauré et al. [46] claim that even in the deterministic case, the size of the state space and the size of the corresponding action sets require to approximately solve the MDP model via ADP. To that end, Sauré et al. [46] assume that the value function can be adequately represented by an affine approximation architecture. Once the approximation is substituted into the linear programming formulation of the discounted MDP, an Approximate Linear Program (ALP) is constructed. The ALP is still intractable, but because of the existence of a constraint for every state-action pair, its dual can be solved via column generation. For the formulations of the MDP, ADP and ALP we refer to the paper of Sauré et al. [46]. Whereas Barz and Rajaram [4] also start with formulating an MDP and ADP, they further used techniques from ADP to derive an upper bound. Afterwards, they simplified this upper bound problem to obtain an optimization problem that is easily solvable and yields approximated marginal values of one unit of capacity of the constraining resources. Finally, Barz and Rajaram [4] use heuristic methods in order to solve the patient admission and scheduling problem.

2.3 Conclusion

Because of the fact that no literature on the optimization of planning and scheduling in the jurisdiction is available, we had to search for *similar* problems addressed in literature. The scheduling problem can be seen as a variation of the well-known AP, but since multiple tasks have to be assigned to multiple people, a more extended problem description is needed. The SRCPSP addresses a scheduling problem that uses stochastic durations. Despite this initial problem being NP-hard, some researchers have used this problem formulation in order to formulate scheduling problems *similar* to our problem. It could subsequently be solved by SDP or heuristic methods (such as meta-heuristics) could be used in order to find a solution. Since this problem formulation has not been used that often in practice, we also have investigated other problem formulations, such as mixed integer programming in which the durations are assumed to be deterministic.

For the planning problem, the MASPH is discussed using solution methods applied in rehabilitation and radiotherapy. For the stochastic online formulation of the planning problem, dividing the planning problem into two stages is an often used solution method. MDP is an often used problem formulation which is mostly solved by ADP. In conclusion, the methods discussed could be appropriate for solving the planning and scheduling problem addressed in this thesis, but one has to make an informed decision after analyzing the context in Section 3.

3 Context analysis

This chapter gives a more elaborate description of the processes one encounters during the planning- and scheduling in administrative law as well as commercial law. In Section 3.1 we elaborate on the general current planning- and scheduling process, where we elaborate on the planning- and scheduling process per jurisdiction in Section 3.2. The process regulations, ‘Code Zaaktoedeling’ as well as conversations with employees of the Rechtbank have been used to create this analysis [1]. In Section 3.3 we state the goal and scope of the research.

3.1 General planning- and scheduling process

Every court plans and schedules separately for each jurisdiction. In our thesis we focus on the jurisdictions administrative law (in Dutch: ‘bestuursrecht’) and commercial law (in Dutch: ‘handelsrecht’) for the locations Rotterdam (including Rotterdam and Dordrecht) and Overijssel (including Zwolle, Almelo and Enschede).

As for the *scheduling* part, some of the teams work with a scheduling program, called ‘Zitting Rooster Planning’ (ZRP), whereas some still make their schedule by hand in Excel. Schedulers deal with:

1. The annual plan filled with the number of expected cases expected per category;
2. The professional standards giving the employability by jurisdiction;
3. The classification of cases:
    a) Whether it is one-fold or multiple case;
    b) Which team the case belongs to;
    c) Whether the case is media-sensitive;
    d) Whether the case is a so-called ‘summary proceedings’;
    e) Only in commercial law: The complexity of the hearing (expressed in the letters A, B, C or D). For hearings of a higher complexity more time has to be scheduled, for preparation, hearing and writing of the verdict.
4. The availability of the courtrooms. Besides unavailability of earlier scheduled hearings, one also deals with the so-called ‘zaakverdelingsregelingen’, describing which types of hearings can be handled at which locations.

As for the *planning* part, every jurisdiction has its own planning method. Planners deal with:

1. The prevent data of parties, lawyers and other litigants;
2. The duration of a hearing;
3. The classification of cases:
    a) Whether it is one-fold or multiple case;
    b) Which team the case belongs to;
    c) Whether the case is media-sensitive;
    d) Whether the case is a so-called ‘summary proceedings’;

[1] The process regulations as well as the ‘Code Zaaktoedeling’ can be found on the site of the Rechtspraak

e) Only in commercial law: The complexity of the hearing (expressed in the letters A, B, C or D). For hearings of a higher complexity more time has to be scheduled, for preparation, hearing and writing of the verdict.

3.2 Planning- and scheduling process per jurisdiction

3.2.1 Administrative law

Administrative law deals with claims of a party appealing against government decisions, within fields of social security, the non-award of subsidies, spatial planning and so on. If the government organization has rejected the objection that the party raised against their decision, the prosecutor can submit this objection to the court.

In administrative law a schedule is made half a year in advance. This schedule shows which hearings have to be handled at what time, including the classification of hearings, which are listed above. The schedule must then be accomplished in accordance to the following rules:

1. The hearings must be practicable within the capacity of court rooms, that has been made available centrally;
2. Hearings must be scheduled such that the expected production for the team per year can be met;
3. The contract durations of the workers must be taken into account such that the number of scheduled hearings could be executed. Full-time judges are employable for hearings 40 weeks a year, which translates to one hearing a week or two small hearings per week. Part-timers are scheduled in proportion to their contract. Legal assistants are employable for hearings 20 weeks a year, which translates to two hearings a month. Moreover, requested permission for vacation has to be taken into account;
4. Specific scheduling wishes could be taken into account, for example when a judge prefers to have two smaller hearings in a week instead of one bigger hearing. Initially, preferences such as morning or afternoon hearings or hearings on a certain day will not be granted;
5. The last restriction is created by the fact that only a certain number of hearings could be scheduled each week.

Based on the availability of the judges and legal assistants the planning can be made, taking into account the following points:

1. All durations (preparation hearing, hearing and writing of the verdict) have been processed into a number of points. Subsequently, it has been decided how many points a judge and a legal assistant have to carry out in a week. A judge has to fulfill fourteen points for hearings a week and a legal assistant has to fulfill these points within two weeks. Part-timers have to fulfill a certain number of points, in proportion to their contract.
2. Employees should not be scheduled for a hearing a week in advance of a big vacation. A big vacation is defined as a vacation that lasts three weeks or longer.
3. In the week after a so-called big vacation a one-fold hearing could be scheduled, at least three working days after returning to work, with a maximum of ten points.

4. The availability of the workers has to be taken into account. Apart from the labels “available” and “non-available”, one can also register the so-called label *free*, which indicates that one is available for a hearing if necessary. One could use this label when one has scheduled an appointment that day that could be rescheduled if that is necessary, such as a dentist appointment.

3.2.2 Commercial law

Commercial law deals with matters such as insurance law, personal injury and professional liability. Not all hearings have to take place physically and hearings do not contain multiple cases. Nowadays, when a case arrives, the team chairman assigns this case to a judge and thereafter, it will be planned. Sometimes, legal assistants are already allocated, but most of the time, they are only allocated after the case is planned. When the case is assigned to a judge, the planners will look for a date for the hearing that is at least four weeks ahead (in order for the judge to have enough preparation time) and at the latest thirteen weeks ahead (such that it meets the standards). Furthermore, the planners adhere to the guideline that judges can only handle two hearings a week and therefore planners do not schedule a hearing in a week that is already filled with two hearings for a certain judge. When a planner does not see any possibility to schedule a case, it returns it to the team chairman such that he/she can assign a new judge to the case or he/she can determine that the judge should be able to handle more cases in a certain week. For instance, if a week is filled with cases that are not that heavy, more than two cases can be scheduled. When assigning a case to a judge, one has to balance the expertise of the judge with the heaviness of the case. In addition, when assigning cases to judges, the taxability and experience is taken into account. Since the team chairmen do not have a complete overview of the agendas of the judges, as the planners do have, one cannot ascertain optimality of the alignment of cases to judges. The location of the hearing is only allocated after the date and time are established; hence we do not include any location restrictions in our research.

3.3 Goal and scope of the research

The main goal of this research is to find the optimal way to determine the moment of the hearing in such a way that there is enough time for the preparation of the hearing and such that the judgement can take place within the predetermined norms. In order to do this, we divide the planning- and scheduling process into three sub processes, namely:

1. Preparation hearing: the time the judge(s)/legal assistant are working on a case in the time between the application of lawsuit until the hearing;
2. The hearing itself;
3. Writing verdict: the time the judge(s)/legal assistant are working on with a case in the time from the hearing until the judgement.

In all three sub-processes we are dealing with some uncertainties. First of all, the duration of the hearing is uncertain, since one does not know exactly how long people have to be interrogated for when a lawsuit is submitted. Furthermore, it is uncertain whether the hearing will take place at all. Parties can decide, for instance, to resolve the problem among themselves before the scheduled hearing has taken place. This frees up time for the judges, as they no longer have to prepare the hearing. Parties can also decide during the hearing to enroll into mediation or settlement, which also frees up time for the judges, as they no longer have to write the verdict. Lastly, the durations of the different appointments -preparation of the hearing and writing of the verdict- contain stochastic elements.

In the prequel, the research problem is defined as the planning and scheduling problem for administrative law and the planning problem for commercial law. Because of time restrictions, the scheduling problem for administrative law remains a topic for further research. Because of the fact that the literature review provides us several possible solution methods for the planning problem of commercial law, we start with formulating an ILP using input from the different discussed problem formulations. The ILP gives us insight into the planning problem and can also act as a starting point for the other methods discussed above, such as the MDP. After we have constructed the ILP, we aim to extend the model in order to fit better to reality step by step, for example by including the aforementioned uncertainties.

4 Integer Linear Program commercial law

In this chapter we elaborate on the ILP for commercial law. In Section 4.1 we give the formulation of the ILP for commercial law, after which we give the decision variables in Section 4.2, constraints in Section 4.3 and the objective in Section 4.4. Then, we give the assumptions/simplifications we have made to model the problem in Section 4.5. Appendix B gives an overview of the model for commercial law, including the parameters, variables, objective and constraints. A few parameters and constraints have to be added in order to be valid for the planning problem of administrative law. Appendix E gives an overview of the additions one has to make to the model in order to be valid for administrative law (including the new parameter, variables and constraints). Unfortunately, due to time restrictions, these additions have not been implemented and tested.

4.1 Model formulation

Assuming deterministic durations, the model has to generate a planning proposal consisting of assigned judge(s), a legal assistant and a starting time for each appointment. We model this scheduling problem as an ILP that is intended for scheduling a series of appointments for one case at a time. A Linear Program (LP) is a mathematical optimization problem in the following form [36]: "minimize $c^T x$ subject to $Ax \geq b$ and $x \geq 0$", where we have the additional requirement that $x \in \mathbb{Z}^d$ in an ILP. Although this process may not produce the best overall schedules, it enables a direct response to a case that needs to be scheduled [9]. This gives direct clarity to the prosecutor as desired, which we have deduced from the "Programma Tijdige Rechtspraak". After we have constructed the ILP, we want to extend the model in order to fit better to reality step by step. This could for example be done by including uncertainty and possible future requests, where possible future requests can be included by transposing the offline scheduling problem into a (stochastic) online version.

4.2 Decision variables

For each appointment, we have to decide upon which judge(s) and legal assistant are assigned to the appointment and the starting timeslot. To model the problem, we use index a for appointments, j for judges, l for legal assistants and t for timeslots. Hence, the decision variables are as follows:

$$x_{ajt} = \begin{cases} 1, & \text{if appointment } a \text{ is assigned to judge } j \text{ and starts at the beginning of timeslot } t \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{alt} = \begin{cases} 1, & \text{if appointment } a \text{ is assigned to legal assistant } l \text{ and starts at the beginning of time slot } t \\ 0, & \text{otherwise.} \end{cases}$$

Each day has T timeslots, where $T \in \mathbb{N}_0$. So, $t = 0$ is the first timeslot on day one, $t = T$ is the first timeslot on day two, and so on. For each appointment within a series (of preparation, hearing and writing of the verdict) we must select the judge and legal assistant to which the case is assigned, as well as the starting timeslot. Each case has three corresponding appointment numbers of which one is included in the set preparation $\mathcal{P}$, one in the set hearing $\mathcal{H}$ and one in the set writing of the verdict $\mathcal{W}$. When there are R appointments that must be scheduled, all three sets have a cardinality of R. In the case of R appointments that have to be scheduled and appointment $\{a\} \in \mathcal{P}$ corresponds to the preparation of the same case as the hearing in appointment $\{a+R\} \in \mathcal{H}$ and the preparation of writing the verdict in appointment $\{a+2\cdot R\} \in \mathcal{W}$. We denote the cardinality of a set A as $|A|$, with a vertical bar on each side. This same notation is used for absolute value; the meaning depends on its context. We only create the variables x_{ajt} and y_{alt} if the last timeslot in which the appointment is executed is not larger than the number of timeslots. Hence, $x_{ajt} = 0$ for all $(a, j, t)|t + M(a, j) < |Timeslots|$, where $M(a, j)$ is the duration of appointment a for judge j. Also, $y_{alt} = 0$ for all $(a, l, t)|t + N(a, l) < |Timeslots|$, where $N(a, l)$ is the duration of appointment a for legal assistant l. A limitation to this way of writing the decision variables could be that the preparation and writing of the verdict could not be split up into multiple periods. An overview of the indices and sets can be found in Table 1.

Table 1: Indices and sets ILP

Index	**Description**	**Set**	**Description**
$t, \widetilde{t}$	Timeslots	$\mathcal{P}$	Appointments with regard to preparation of hearing
$a, \widetilde{a}$	Appointments	$\mathcal{H}$	Appointments with regard to the hearing itself
$j, \widetilde{j}$	Judges	$\mathcal{W}$	Appointments with regard to writing of the verdict
$l, \widetilde{l}$	Legal assistants	$\mathcal{A}$	All appointments
s	Specialisms		
d	Difficulties		

4.3 Constraints

4.3.1 Basic planning

Since only one assignment can be scheduled per timeslot for both judges and legal assistants, we introduce the constraints in equation (4.1) and (4.2).

$$\sum_{a} x_{ajt} \leq 1, \quad \forall \quad j, t \tag{4.1}$$

$$\sum_{a} y_{alt} \leq 1, \quad \forall \quad l, t \tag{4.2}$$

We introduce parameter M_{aj} that states the number of timeslots judge j needs to fulfill appointment a and parameter N_{al} that states the number of timeslots legal assistant l needs to fulfill appointment a. Here, $M_{aj} = N_{al}$ for all $a \in \mathcal{H}$, since the hearing has the same duration for the judge as well as the legal assistant. If judge j is assigned to an appointment a starting in timeslot t (and hence $x_{ajt} = 1$), he/she should not be scheduled in the timeslots afterwards where he/she is still executing appointment a. This is modelled in equation (4.3). The same holds for the legal assistant, which we model in equation (4.4).

$$\sum_{\widetilde{a},\widetilde{t}|t<\widetilde{t}\leq t+M_{aj}-1} x_{\widetilde{a}j\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a, j, t \tag{4.3}$$

$$\sum_{\widetilde{a},\widetilde{t}|t<\widetilde{t}\leq t+N_{al}-1} y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a, l, t \tag{4.4}$$

All scheduled appointments have to be assigned one legal assistant. We model these constraints in a way that $\sum_{l} y_{alt}$ has to equal 1 if appointment a will be executed by legal assistant l and starts at timeslot t (and thus $y_{alt} = 1$). We model these constraints in equation (4.5). Each scheduled multiple case appointment has to be assigned three judges and each single appointment has to be assigned only one judge. In order to model these constraints, we introduce the binary parameter τ_a that is 1 if appointment a is a multiple case appointment and 0 otherwise. Subsequently, these constraints are modelled in equations (4.6), (4.7) and (4.8). Equations (4.9) and (4.10) model the constraints implicating that the hearing of a multiple case must be scheduled at the same time for all three judges assigned to that multiple case.

$$\sum_{\widetilde{l}\widetilde{t}} y_{a\widetilde{l}\widetilde{t}} \leq 2 - y_{alt} \quad \forall \quad a, l, t \tag{4.5}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a, j, t \tag{4.6}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a, j, t \tag{4.7}$$

$$(1 - \tau_a) \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq (1 - \tau_a) \cdot (2 - x_{ajt}) \quad \forall \quad a, j, t \tag{4.8}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a \in \mathcal{H}, j, t \tag{4.9}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{4.10}$$

The hearing of a certain case must be handled by the same judge(s) and legal assistant as the preparation of that hearing and the writing of the verdict corresponding to that hearing. If a certain preparation (or hearing) is scheduled for a judge and legal assistant, the corresponding hearing (or writing of the verdict) has to be treated by the same judge and legal assistant. These constraints are modelled in equations (4.11)-(4.14).

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{P}, j, t \tag{4.11}$$

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{4.12}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{P}, l, t \tag{4.13}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{H}, l, t \tag{4.14}$$

The timeslots scheduled for one hearing must be scheduled consecutively within the same day, where T is the number of timeslots in a day. Since the hearing must take place at the same time for the judges and legal assistants, we make use of x_{ajt} and M_{aj} to model this constraint without loss of generality. This constraint is captured in equation (4.15).

$$\frac{x_{ajt} \cdot t - \{x_{ajt} \cdot t\} \mod T}{T} = \frac{x_{ajt} \cdot t - x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T}{T} \quad \forall \quad a \in \mathcal{H}, j, t \tag{4.15}$$

Besides, sequence should also be taken into account. For each case, the preparation should take place before the hearing and the hearing should take place before writing the verdict. We let

$C_{a,\widetilde{a}}$ be 1 if appointment a should take place before $\widetilde{a}$ and 0 otherwise. $C_{a\widetilde{a}}$ is thus 1 if ($a \in \mathcal{P}$ and $\widetilde{a} \in \mathcal{H}$) or ($a \in \mathcal{H}$ and $\widetilde{a} \in \mathcal{W}$). When $|\mathcal{P}| = 3$, the $C_{a\widetilde{a}}$ matrix thus looks as follows with a on the y-axis and $\widetilde{a}$ on the x-axis:

$$C_{a\widetilde{a}} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

The constraints on the precedence relations are modelled in equations (4.16)-(4.19).

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}j\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, j, t \tag{4.16}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}j\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{W}, j, t \tag{4.17}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, l, t \tag{4.18}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{P}, l, t \tag{4.19}$$

4.3.2 Appointment judge(s) and legal assistant

The legal assistant, judge(s), prosecutor and defendant have to be available during the hearing. We introduce the binary parameters BJ_{jt}, BL_{lt} and BP_{at}. Let BJ_{jt} be 1 if judge j is available in timeslot t, let BL_{lt} be 1 if legal assistant l is available in timeslot t and let BP_{ta} be 1 if the other parties (prosecutor and defendant) corresponding to appointment $a \in \mathcal{H}$ are available in timeslot t. Furthermore, we use the parameter M_{aj} that states the number of timeslots that have to be scheduled for judge j to execute appointment $a \in \mathcal{H}$, without loss of generality. We thus get the following constraint when modeling that the legal assistant, judge(s) and prosecutor have to be scheduled for the hearing:

$$x_{ajt} \cdot y_{alt} \leq BL_{l\widetilde{t}} \cdot BJ_{j\widetilde{t}} \cdot BP_{a\widetilde{t}} \quad \forall \quad a \in \mathcal{H}, t \leq \widetilde{t} \leq t + M_{aj} - 1$$

Since this constraint contains a multiplication of two binary variables, it is not linear. Because of this, we use a modeling trick to make this constraint linear. The modeling trick works as follows: "*Computing* $x_1 \cdot x_2$ with both being binary variables corresponds to an *and* of x_1 and x_2. This can be modelled by using a binary variable y and adding the constraints $y \leq x_1$, $y \leq x_2$ and $y \geq x_1 + x_2 - 1$" [36]. By using this trick, we obtain the linear constraints given in equation (4.20) up to and including (4.23).

$$q_{ajlt} \leq x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{4.20}$$

$$q_{ajlt} \leq y_{alt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{4.21}$$

$$q_{ajlt} \geq x_{ajt} + y_{alt} - 1 \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{4.22}$$

$$q_{ajlt} \leq BL_{l\tilde{t}} \cdot BJ_{j\tilde{t}} \cdot BP_{a\tilde{t}} \quad \forall \quad a \in \mathcal{H}, l, j, t \leq \tilde{t} \leq t + M_{aj} - 1 \tag{4.23}$$

Since the judge(s) and legal assistant have to be available for the preparation and writing of the verdict as well, we include the constraints in equation (4.24) and (4.25) for the judge(s) and legal assistant, respectively.

$$x_{ajt} \leq BJ_{j\tilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, j, t \leq \tilde{t} \leq t + M_{aj} - 1 \tag{4.24}$$

$$y_{alt} \leq BL_{l\tilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, l, t \leq \tilde{t} \leq t + N_{al} - 1 \tag{4.25}$$

Beyond the fact that all parties have to be available during the hearing, the hearing also has to take place on the same timeslot for all parties. This is regulated in equation (4.26).

$$\tau_a \cdot \sum_j x_{ajt} + (1 - \tau_a) \cdot \sum_j x_{ajt} = 3 \cdot \tau_a \cdot \sum_l y_{alt} + (1 - \tau_a) \cdot \sum_l y_{alt} \quad \forall \quad a \in \mathcal{H}, t \tag{4.26}$$

Besides the availability, the specialisms are also important for scheduling the judge(s) and legal assistant. Appointments that have a certain specialism should be allocated to a legal assistant and judge(s) of that specialism. Each appointment also has a difficulty that determines which legal assistant is suitable to handle that appointment. We introduce the following binary parameters: I_{js} is 1 if judge j is suitable to handle cases of specialism s and J_{lsd} is 1 if legal assistant l is suitable to handle cases of specialism s and difficulty d. Furthermore, we introduce the binary parameter θ_{ads} which is 1 if appointment a has difficulty d and specialism s. We give the corresponding constraints in equation (4.27) and equation (4.28).

$$I_{js} \cdot \sum_d \theta_{ads} \leq 2 - x_{ajt} \quad \forall \quad a, j, t, s \tag{4.27}$$

$$J_{lsd} \cdot \theta_{ads} \leq 2 - y_{alt} \quad \forall \quad a, l, t, s, d \tag{4.28}$$

4.3.3 Distribution scheduled hours workers

Initially, we want that the work is evenly distributed over all the workers. Hence, we minimize the difference between the maximum and minimum idle time in the available (for scheduling) time. We therefore introduce the auxiliary variables χ and υ that give the maximum difference between available and scheduled time, subtracted by the minimum difference, respectively for the judges and the legal assistants. This way, the variance between the difference in available hours and scheduled hours is minimized for both the judges and the legal assistants. These auxiliary variables are calculated in equations (4.29)-(4.34) and added to the objective function. The objective function is illustrated in Section 4.4.

$$\chi = \max_j [\sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj}] - \min_j [\sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj}]$$

$$\chi = r - s \tag{4.29}$$

$$r \geq \sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj} \quad \forall \quad j \tag{4.30}$$

$$s \leq \sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj} \quad \forall \quad j \tag{4.31}$$

$$\upsilon = \max_l [\sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al}] - \min_l [\sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al}]$$

$$\upsilon = u - w \tag{4.32}$$

$$u \geq \sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al} \quad \forall \quad l \tag{4.33}$$

$$w \leq \sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al} \quad \forall \quad l \tag{4.34}$$

4.3.4 Standards that must be met

Finally, there are a couple of standards that must be met. At first, the hearing (the appointment in the set $\mathcal{H}$) should take place within ϵ_1 days of the arrival of the case. Therefore, we would like the number of days between the last timeslot for the hearing and the timeslot of the arrival of the case to have an upper norm. We use the variable x_{ajt} to denote the timeslot of the hearing without loss of generality, since the hearing takes place in the same timeslots for the judge(s) as well as the legal assistant. We let γ_a be the first timeslot of the day after the arrival of the case. We then add the auxiliary variable ψ_a that gives the number of timeslots that the scheduled hearing in appointment a is exceeding the standard - if it is exceeding the standard - which is calculated in (4.35)[2]. In this equation we use the following notation: $[z]^+$ which means that the outcome is z if z is positive and 0 if z takes a negative value, thus $[z]^+ = \max\{0, z\}$. By using this construction, we ensure that no value is added to the objective if the appointment is scheduled within the norms, and the number of timeslots exceeding the norm is added to the penalty function if the scheduled appointment is exceeding the norm. The sum of the auxiliary variables ψ_a over all appointments ($\sum_a \psi_a$) is subsequently added to the objective, which is illustrated in Section 4.4. Then, the judgement (end of writing of the verdict) has to be finished within ϵ_2 days from the hearing. We therefore want the number of days between the biggest timeslot used for the writing of the verdict (over the judge(s) as well as the legal assistant) minus the biggest timeslot used for the hearing to have an upper norm. We ensure that the judge has a higher last timeslot for writing the verdict compared to the legal assistant in equation (4.36). In this equation we also use $[z]^+ = \max\{0, z\}$ for the same reasons as in equation (4.35). Furthermore, we add the sum of the auxiliary variables ω_a over all appointments ($\sum_a \omega_a$) that gives the number of timeslots by which this norm is exceeded for scheduled appointment a to the objective. We calculate this variable in equation (4.37) and illustrate the composition of the objective function in Section 4.4.

$$\psi_a = [\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T - \gamma_a - \epsilon_1 \cdot T]^+ \quad \forall \quad a \in \mathcal{H}, \epsilon_1 \in \mathbb{N} \tag{4.35}$$

[2]We calculate the number of timeslots in between the two moments based on the number of days in between, where we express the day as the first timeslot on that day. We use the same method of calculation in (4.37).

$$\sum_{jt} x_{ajt} \cdot \{t + M_{aj}\} \geq \sum_{lt} y_{alt} \cdot \{t + N_{al}\} \quad \forall \quad a \in \mathcal{W}, t \tag{4.36}$$

$$\begin{aligned}\omega_a = [\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} - (\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} \mod T) + T - \\ (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T) \\ -\epsilon_2 \cdot T]^+ \quad \forall \quad a \in \mathcal{H}, \epsilon_2 \in \mathbb{N}\end{aligned} \tag{4.37}$$

4.4 Objective function

4.4.1 Initial objective function

We want to minimize the number of unscheduled cases as well as the unfair distribution of the idle time of the judges and legal assistants. We introduce the binary variable n_a which is 1 if the case corresponding to appointment $a \in \mathcal{P}$ is not being scheduled and 0 otherwise. We define a case not scheduled if at least one of the appointments corresponding to that case is not scheduled, for either the judge j, the legal assistant l or both. The binary variable n_a only has a value for $a \in \mathcal{P}$ such that it is bounded for the number of cases that have to be scheduled. We thus have to define n_a as follows:

$$\sum_{jt} x_{ajt} \cdot \sum_{lt} y_{alt} = \tau_a \cdot (3 - 3 \cdot n_a) + (1 - \tau_a) \cdot (1 - n_a) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A}$$

In order to linearize the constraint, we make use of the modelling trick that linearizes the multiplication of a binary variable with a continuous variable. This trick works as follows: "The multiplication of a binary variable x_1 with a continuous variable x_2 where $0 \leq x_2 \leq u$ can be expressed by introducing a new variable y that meets the following constraints: $y \leq ux_1$, $y \leq x_2$, $y \geq x_2 - u(1 - x_1)$, $y \geq 0$" [36]. Using this method with $x_1 = \sum_{lt} y_{alt}$ and $x_2 = \sum_{jt} x_{ajt}$ we obtain the constraints in (4.38)-(4.41) where b_a is a non-negative variable. Furthermore, χ and υ are defined by equations (4.29) and (4.34). The number of timeslots a hearing is exceeding its norm is denoted by ψ_a for appointment $a \in \mathcal{H}$ and the number of timeslots the verdict was completed after its established standard is denoted by ω_a for appointment $a \in \mathcal{H}$, which are defined in equations (4.35) and (4.37). In the objective we make use of the weight factors β_1, β_2, β_3, β_4 and β_5. The objective will thus be:
min $\beta_1 \cdot \sum_a n_a + \beta_2 \cdot \chi + \beta_3 \cdot \upsilon + \beta_4 \cdot \sum_a \psi_a + \beta_5 \cdot \sum_a \omega_a$

$$b_a \leq 3 \cdot \sum_{lt} y_{alt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{4.38}$$

$$b_a \leq \sum_{jt} x_{ajt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{4.39}$$

$$b_a \geq \sum_{jt} x_{ajt} - 3 \cdot (1 - \sum_{lt} y_{alt}) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{4.40}$$

$$b_a = 3 \cdot \tau_a \cdot (1 - n_a) + (1 - \tau_a) \cdot (1 - n_a) \quad \forall \quad a \in \mathcal{P} \tag{4.41}$$

4.4.2 Second scenario

Since a lot of timeslots could remain unscheduled when aiming to distribute the idle time as fairly as possible, we also introduce an alternative objective, namely:

$$\mathbf{min}\ \beta_1 \cdot \sum_a n_a + \beta_2 \cdot \sum_j \chi_j + \beta_3 \cdot \sum_l \upsilon_l + \beta_4 \cdot \sum_a \psi_a + \beta_5 \cdot \sum_a \omega_a$$

In this objective, χ_j and υ_l are the number of timeslots judge j is available but not scheduled and legal assistant l is available but not scheduled, respectively. In the next chapter we explore the differences when implementing the different objective functions. The variables χ_j and υ_l are calculated as follows:

$$\chi_j = \sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj} \quad \forall \quad j$$

$$\upsilon_l = \sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al} \quad \forall \quad l$$

4.5 Simplifications

In order to model the problem we have made the following simplifications:

1. The first and probably most important simplification we have made is that an appointment could not be divided into multiple appointments. This makes it hard to schedule appointments for part-timers. When appointments need to be broken up, one has to use another decision variable, namely: the decision variable that determines all timeslots in which an appointment lasts, instead of the starting timeslot of the appointment. This entails the big disadvantage that the model grows enormously because of the increase of the number of decision variables. The model already has a lot of decision variables and constraints and therefore we have decided that this remains a part for further research. Besides, the model can still be used where appointments of short durations are not entered as unavailable timeslots. Since the agendas of the judges and legal assistants will still have free time (and the durations of preparation and writing of the verdict are still roughly estimated) the model remains usable.

2. We assume hearings could not last longer than a day. This means that we do not include the 'mega cases' (in Dutch: 'megazaken') into our model. These cases are very rare and the hearings last a couple of days. Implementing these cases is very complex and since they are very rare, we have decided to leave them out of the model.

3. We have decided to not implement the parameter of whether the case is media-sensitive to limit the size of the model. This is because we have implemented several specialisms and difficulties and the parameter of whether a case is media-sensitive can thus be added as an extra specialism or difficulty.

4. Some courts have special judges for summary proceedings. We have decided not to implement this explicitly, since it can also be included as an extra specialism or difficulty.

5. Some large courts (for instance the one situated in Rotterdam) work with different teams within a specialism. Since the model has to be valid for the smaller courts, we have decided to not implement this explicitly. When there are different teams (that can handle different tasks) within one specialism, those teams have to be modelled as different specialisms such that their specialties can be included in the model.

6. We have decided to not implement any limitations on the availability of court rooms. This is because the planners have indicated that the availability of the court rooms do not form a limitation in the planning. Also, most planners assign a hearing to a courtroom after it has been planned, whereby the roster of the courtrooms does not influence the planning.

7. We have decided to not model the restrictions on long vacations, defined as vacations of three weeks or longer. These restrictions are as follows: Employees should not be scheduled for a hearing a week in advance of a big vacation. The week after a so-called big vacation a one-fold hearing could be scheduled, at least three working days after return, with an upper limit of the time those involve. We have decided not to implement the restrictions on those long vacations, since they do not occur that often and implementing these constraints would be very hard since those vacations could overlay multiple planning horizons.

8. At last, we have decided to not implement the label *free* as mentioned earlier. This label indicates if someone would rather not be scheduled, but is available if there is no alternative for a hearing. In our case, one has to mark this time as available and he/she can reschedule their own time if preparation of a hearing of writing a verdict has been scheduled.

5 Results Integer Linear Program commercial law

In this chapter, we elaborate on the results of the developed model, programmed in AIMMS. A clarification of how the constraints involving a nonlinear component (such as the modulo-function) are programmed in AIMMS can be found in Appendix C. All tests have been performed on the same system: an Intel Core i5-8265U personal computer with 8 GB of RAM and we make use of the 4.71.1 64-bit Windows AIMMS-version, where CPLEX 12.10 is used as solver. We have chosen to start testing our model with on a dataset generated by ourselves, based on the available data of the cases that are handled in Almelo in 2019. The setup of this dataset can be found in Section 5.1 where we illustrate the results in Section 5.2. As last test option we wanted to use real-time data, which we discuss further in Section 5.3.

5.1 Validation data

Currently, durations of preparation and writing of the verdict are not tracked by the courts. Hence, accurate predictions of these durations could not be made in order to use as input in our model. Therefore, we have chosen to start testing our model with on a dataset generated by ourselves, based on the available data of the cases that are handled in Almelo in 2019.

The minimum of treated cases per month in Almelo in 2019 was 77 and the maximum of the treated cases per month in 2019 was 115. In order to carry out the work, the court of Almelo had access to 9.4 fte of judges and 12.7 fte of legal assistants. We work with 8 full-time judges and 12 full-time legal assistants available for hearings, since judges and legal assistants also have other tasks such as workshops and team meetings. We are aware of the fact that legal assistants also work part-time, but since we test our model using short time period and part-timers will never be able to complete the preparation of a hearing, the hearing itself and the writing of the verdict in the given time period. Therefore, we have chosen to only work with full-time legal assistants. We also make the assumption that a judge is $\frac{2}{5}$ of the time working on a case busy preparing the hearing, $\frac{1}{10}$ holding a hearing and $\frac{1}{2}$ writing the verdict.

Based on the above assumptions we have created the following case durations:

- A judge is at least $\frac{8 \cdot 36 \cdot 52/12}{115} \approx 10.85$ hours busy with a case.
- A judge is at most $\frac{8 \cdot 36 \cdot 52/12}{77} \approx 16.21$ hours busy with a case.
- A legal assistant is at least $\frac{12 \cdot 36 \cdot 52/12}{115} \approx 16.28$ hours busy with a case.
- A legal assistant is at most $\frac{12 \cdot 36 \cdot 52/12}{77} \approx 24.31$ hours busy with a case.

Based on these case durations, we have created durations of appointments. In our dataset we deal with 24 cases a week (based on the 77 to 115 cases the court of Almelo has dealt with a month in 2019) of which one is a multiple case. The durations of the appointments are determined in the following way:

- Duration preparation judge: drawing uniformly a number between $\frac{2}{5} \cdot 10.85 \approx 4.34$ and $\frac{2}{5} \cdot 16.21 = 6.48$ and then round up or down this number to the closest number without decimal places.
- Duration hearing judge: drawing uniformly a number between $\frac{1}{10} \cdot 10.85 = 1.085$ and $\frac{1}{10} \cdot 16.21 = 1.621$ and then round up or down this number to the closest number without decimal places.

- Duration writing verdict judge: drawing uniformly a number between $\frac{1}{2} \cdot 10.85 = 5.425$ and $\frac{1}{2} \cdot 16.21 = 8.105$ and then round up or down this number to the closest number without decimal places.
- Duration preparation legal assistant: drawing uniformly a number between $\frac{2}{5} \cdot 16.28 = 6.512$ and $\frac{2}{5} \cdot 24.31 = 9.724$ and then round up or down this number to the closest number without decimal places.
- Duration hearing legal assistant: This number is taken from the duration of the hearing of the judge of the same appointment.
- Duration writing verdict legal assistant: drawing uniformly a number between 16.28 and 24.31 and subtract the duration of the preparation and hearing, belonging to the same case.

Furthermore, we have used five main specialisms and four difficulties as is also dealt with in Almelo, where each judge and legal assistant is able to handle two or three specialisms and each legal assistant can handle all difficulties from one specialty and less from the other(s). The legal assistants can handle different difficulties, which is decided randomly. Furthermore, we decide randomly which judge can handle which specialism, in such a way that every specialism is covered. We use timeslots of one hour and make a schedule for a week. Furthermore, we assume that everyone is available all timeslots, since the model leaves no possibility to interrupt appointments.

5.1.1 Scenario 1

We test with different values for the weight factors. When using the above described dataset, the following ranges apply for the different parts of the objective when using the initial objective (exemplified as the first scenario): $\sum_a n_a = \{0, ..., 24\}$, $\chi = \{0, ..., 35\}$, $\upsilon = \{0, ..., 35\}$, $\sum_a \psi_a = \{0, ..., 24 \cdot 35 \cdot 8\}$ and $\sum_a \omega_a \{0, .., 24 \cdot 35 \cdot 12\}$. We test how the model behaves in the different situations. Since $\epsilon_1 = 60$ and $\epsilon_2 = 30$ in the case of commercial law, we only have timeslots $\{0, .., 35\}$ and $\gamma_a = 0 \ \forall \ a$, exceeding the norms can hardly occur. Adding smaller norms was also not an option since we only schedule for five working days and we work with the original durations of the appointments. Therefore, we have decided to test with several situations in which β_4 and β_5 are not subject to change and therefore $\beta_4 = \beta_5 = 1$ in every situation. Hence, when using this dataset, the challenge lies in scheduling as many appointments as possible and leaving as few timeslots as possible non-scheduled. In the validation set we expand the challenge by taking a larger time horizon such that the standards must be taken into account.

Since the model does not schedule appointments when giving too much weight on the equal division of the working hours, we do not consider situations where β_2 and β_3 are so high such that the best solution is not to schedule any appointment (e.g. $\beta_1 = \frac{1}{24}, \beta_2 = \beta_3 = \frac{1}{35}$). For $\beta_1 = \frac{1}{24}$ and $\beta_2 = \beta_3 = \frac{1}{100}$ the model does not schedule any appointment, but for $\beta_1 = \frac{1}{24}$ and $\beta_2 = \beta_3 = \frac{1}{125}$ we consider the case where $\beta_1 = \frac{1}{24}, \beta_2 = \beta_3 = \frac{1}{125}$. Since we want to compare different values for the weight factors, we also consider the situation where $\beta_1 = 1, \beta_2 = 0, \beta_3 = 0$), the situation where $\beta_1 = 0, \beta_2 = \frac{1}{125}, \beta_3 = \frac{1}{125}$ and finally the situation where we make β_2 and β_3 twice as small so we obtain $\beta_1 = \frac{1}{24}, \beta_2 = \frac{1}{350}, \beta_3 = \frac{1}{350}$.

5.1.2 Scenario 2

We test with different values of the weight factors. When using the above described dataset, the following ranges apply for the different parts of the objective when using the alternative objective (exemplified as the second scenario): $\sum_a n_a = \{0, ..., 24\}$, $\chi = \{0, ..., 8 \cdot 35\}$, $\upsilon = \{0, ..., 12 \cdot 35\}$, $\sum_a \psi_a = \{0, ..., 24 \cdot 35 \cdot 8\}$ and $\sum_a \omega_a \{0, .., 24 \cdot 35 \cdot 12\}$. We test how the model behaves in the different situations. When giving equal weight to the β's (in proportion to their possible values), the model does schedule the appointments, unlike with the initial objective. Hence, the situations we consider for testing the validation data in the second scenario are as follows:

1. All parts of the objective function are equally important. We determine the weight factors by dividing the importance factor by the size class and hence obtain:
   - $\beta_1 = \frac{1}{24}, \beta_2 = \frac{1}{8 \cdot 35} = \frac{1}{280}$, $\beta_3 = \frac{1}{12 \cdot 35} = \frac{1}{420}$, $\beta_4 = 1$ and $\beta_5 = 1$
2. The number of unscheduled appointments is important, the number of unscheduled (but available) timeslots is not:
   - $\beta_1 = 1$, $\beta_2 = \beta_3 = 0$, $\beta_4 = 1$ and $\beta_5 = 1$
3. The number of unscheduled appointments is not important, the number of unscheduled (but available) timeslots is important:
   - $\beta_1 = 0$, $\beta_2 = \frac{1}{280}$, $\beta_3 = \frac{1}{420}$, $\beta_4 = 1$ and $\beta_5 = 1$
4. The number of unscheduled appointments is ten times as important as the number of unscheduled (but available) timeslots:
   - $\beta_1 = \frac{10}{24}, \beta_2 = \frac{1}{280}$, $\beta_3 = \frac{1}{420}$, $\beta_4 = 1$ and $\beta_5 = 1$
5. The number of unscheduled (but available) timeslots is ten times as important as the number of unscheduled appointments:
   - $\beta_1 = \frac{1}{24}, \beta_2 = \frac{10}{280}$, $\beta_3 = \frac{10}{420}$, $\beta_4 = 1$ and $\beta_5 = 1$

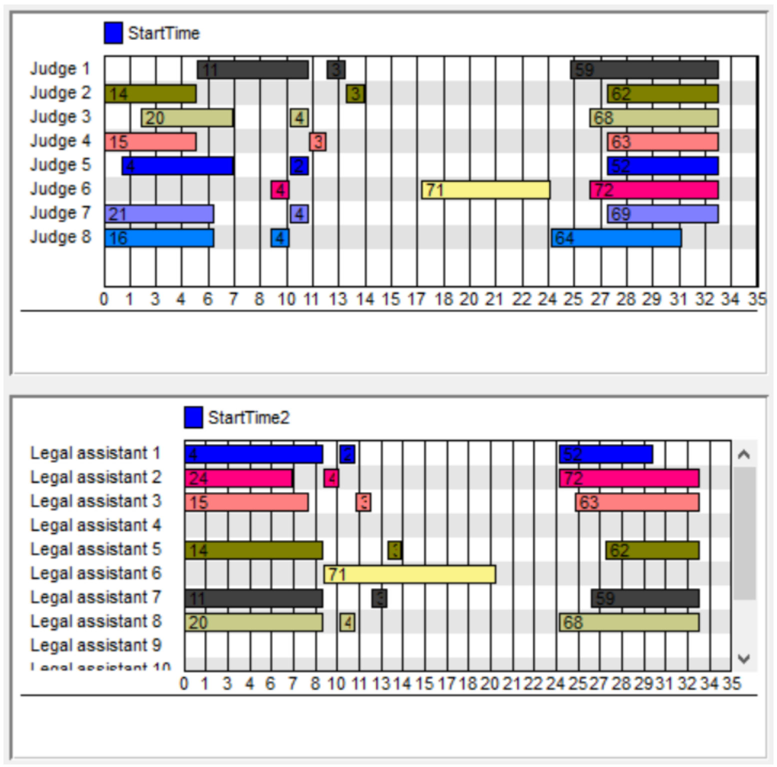


Figure 1: Screenshot result tool (scenario 1) with $\beta_1, \beta_2, \beta_3$ from situation 1

5.2 Illustration results validation data

5.2.1 Scenario 1

In this section we elaborate on some results from testing the model with the data described in Section 5.1. The results itself are given in Appendix D. The interface of the output of the model after running scenario 1 for 30 minutes (with $\beta_1 = \frac{1}{24}, \beta_2 = \frac{1}{125}, \beta_3 = \frac{1}{125}$) is shown in Figure 1. The numbers in the figure correspond to the appointment numbers and appointments of the same color belong to the same case. We can see here that for not every case, all appointments can be scheduled. For the case with eight judges, eight appointments is the maximum that can be scheduled and in the case with ten judges, twelve is the maximum (when putting no weight on the equal division of the working hours). This makes sense, since the number of workers is not enough to handle the work. We namely have eight judges and twelve legal assistants (in the initial case) with 35 timeslots, so judges are $8 \cdot 35 = 280$ timeslots available and legal assistants $12 \cdot 35 = 420$ timeslots, where there are between $24 \cdot 10.85 = 260.4$ and $24 \cdot 16.21 = 389.04$ timeslots for the judges needed to handle the appointments and between $24 \cdot 16.28 = 390.72$ and $24 \cdot 24.31 = 583.44$ timeslots for the legal assistants needed to handle the appointments. Furthermore, we see that the model is able to find a solution quickly when no weight is given on the equal distribution of timeslots, but in the other cases the optimality gap is high -around 45 percent- for every situation. This confirms that the size of the model is large and takes a long time to solve.

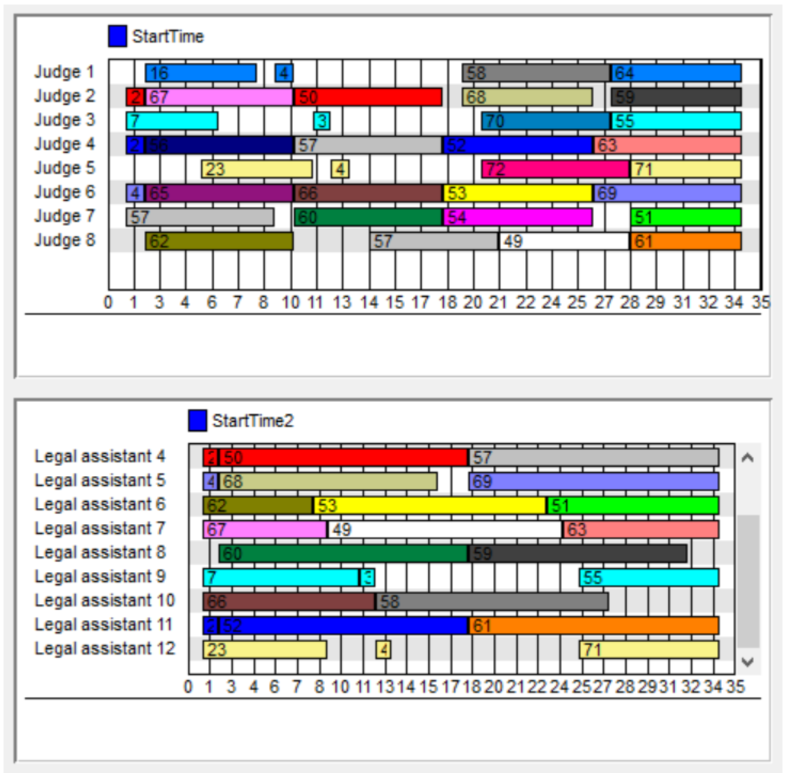


Figure 2: Screenshot result tool (scenario 2) with $\beta_1, \beta_2, \beta_3$ from situation 1

5.2.2 Scenario 2

The interface of the output of the model after running scenario 2 for 30 minutes (with $\beta_1 = \frac{1}{24}, \beta_2 = \frac{1}{280}, \beta_3 = \frac{1}{420}$) is shown in Figure 2. By trial and error we have found that the model with the second scenario is able to find reasonable solutions after ten minutes of running, instead of thirty minutes in the case of the first scenario. Also, the model is able to find a solution when aiming at minimizing the idle time (and not the number of unscheduled appointments). We see here that the optimality gap increases with the size of the model. When two judges have been added to the model, the optimality gap is in every situation more than doubled when running for the same time. Besides, we see a small increase in the optimality gap when the judges and legal assistants can handle (all difficulties of) all specialties instead of only one.

5.3 Illustration results test data

We also wanted to test the model on real data. Therefore, we have used the cases that Overijssel had to schedule during the 23rd of July. Since a lot of the judges, legal assistants, prosecutors and defendants are not available during a long time -because of the summer vacations- some of the standards could not be achieved in any way. Furthermore, we have used timeslots of a day, where preparation, hearing and writing of the verdict all cost one day and the available days are all working days from August up to and including November. Since we had to use the Outlook Calenders of the judges and legal assistants in order to extract their availability, this solution was really dependent on how they have tracked their agendas (for instance some of the workers do not keep track of every appointment in their Outlook agenda which gives us a misleading picture of their availability). The model then still gives earlier data for the hearings, since we could not take into account the preparation and elaboration time of the already scheduled cases as this is not being tracked.

As we still wanted to test the model on real data we have created another dataset, based on the hearings that have taken place, for the judges of the team situated in Almelo, in 2019. We have inserted the appointments that had to be scheduled from January (starting with an empty schedule), where they could also be scheduled in February and March. Thereafter, we wanted to use the appointments of February as appointments that had to be scheduled and we have inserted the already scheduled timeslots in February and March as unavailable. The durations of preparation, hearing and writing of the verdict have been taken from Section 5.1. The undisclosed agendas of judges the team in Almelo have been replaced by disclosed agendas of judges of the team in Zwolle, since they form one team (team Overijssel) together. Furthermore, we have used timeslots of one hour and gave the cases specialisms at random (since the specialism of a certain case is not tracked in the agendas). The specialisms of the judges was known, but those of the legal assistants have been determined at random. The appointments have to be handled within the same norms as with the validation data and the assumptions on the specialisms and difficulties also remain the same.

Unfortunately, the memory of my computer was not enough to handle all cases of January. Since we work with partly filled agendas in the sequel, the model has to go along less possibilities and could be able to solve problems over a larger time horizon. Besides, the decision variables with timeslots already filled should not be generated from the beginning, which also decreases the size of the model.

6 Stochastic Dynamic Program commercial law

In order to take the uncertainty of the durations into account as well as involving the arrival of future cases, we develop a SDP. In Section 6.1 we elaborate on the reasons to develop a SDP and what such a model entails. Subsequently, in Section 6.2 we give the formulation of the SDP and in Section 6.3 we give the solution approach. Finally, we elaborate on the -earlier mentioned-occurrence of some scheduled appointments that will not find passage in Section 6.4.

6.1 Introduction

The ILP does not cover stochastic durations and predictions for the future. The literature review has showed similarities between our problem (when including those uncertainties) and the MASPH. When future requests are taken into account, the MASPH can be formulated as a two-stage SIP [41]. In order to solve this model, it can be divided into two stages by the SAA approach and heuristic methods can then be used to solve these two stages. Also, the problem can be formulated as a MDP which can be considered as the standard method for planning under uncertainty can be solved by Approximate Dynamic Programming [46] [4]. Since there is more literature available on planning under uncertainty by using a MDP we have chosen to focus on this way of modelling.

The qualifier "Markov" (for Markov Decision Processes) can be used if the underlying stochastic processes is a stationary process that features the Markov property. The Markov property is the property that the reward functions as well as the transition probabilities only depend on the current state and the action selected by the decision maker in that state. For a more detailed described on the Markov Decision Processes and its applications, we refer to the book of Puterman [40].

To formulate a MDP, we introduce a set of points in time on which decisions are made, called *decision epochs*. At each decision epoch, the system occupies a *state* and at each state, the decision makes has to choose an *action*. As a result of choosing an action $a \in A_s$ in state s at decision epoch t [40]:

1. the decision maker receives a reward $r_t(s, a)$ and;
2. the system state at the next decision epoch is determined by the probability distribution $p_t(\cdot|s, a)$, where $\sum_{j \in S} p_t(j|s, a) = 1$.

6.2 Formulation SDP

We use the following sets:

1. The set containing the judges: $\mathcal{J} = \{1, ..., J\}$, where $j \in \mathcal{J}$;
2. The set containing the legal assistants: $\mathcal{L} = \{1, ..., L\}$, where $l \in \mathcal{L}$
3. The set containing the appointments: $\mathcal{A} = \{1, ..., A\}$, where $a \in \mathcal{A}$;
4. The set containing the type of appointments: $\mathcal{I} = \{1, ..., I\}$ where $i \in \mathcal{I}$;
5. The set containing the timeslots: $\mathcal{T} = \{1, ..., T\}$, where $t \in \mathcal{T}$.

At the end of each day or week (depending on the team), the booking agent must decide on which timeslots to book a certain case, over an infinite horizon. We thus have to deal with a rolling horizon. This rolling horizon approach is also preferred because the finite horizon approach may cause unwanted and short-term focused behavior in the last periods and it ensures that the most recent information is used [32]. To formulate the SDP we have to define the states, decisions, direct costs, transition probabilities and the optimal value function.

6.2.1 States

The state of the system, denoted $\mathbf{s} \in \mathbb{S}$ is represented by a vector $\mathbf{s} = (x_{jt}, y_{lt}, M_{aj}, N_{al}, \theta_{ads}, n_a)$ where M_{aj} and N_{al} represent the durations of appointment a for respectively judge j and legal assistant l, θ_{ads} is the binary variable that is 1 if appointment a has difficulty d and specialism s, n_a is the binary variable that is 1 if appointment a could not be scheduled previously and the definition of x_{jt} and y_{lt} is given below. The state $\mathbf{s} \in \mathbb{S}$ is thus a large vector that can be represented as follows: $\mathbf{s} = (x_{11}, .., x_{JT}, y_{11}, .., y_{LT}, M_{11}, .., M_{AJ}, N_{11}, .., N_{AL}, \theta_{111}, .., \theta_{ADS}, n_1, .., n_A)$.

$$x_{jt} = \begin{cases} 1, & \text{if judge } j \text{ is available (not yet scheduled) on timeslot } t \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{lt} = \begin{cases} 1, & \text{if judge } j \text{ is available (not yet scheduled) on timeslot } t \\ 0, & \text{otherwise.} \end{cases}$$

6.2.2 Decisions

At the end of each day or week (depending on the team), the booking agent must decide on which timeslots to book a certain case, that corresponds to three appointments: preparation, hearing and writing of the verdict. We denote an action by $\boldsymbol{\alpha} = (x_{ajt}, y_{alt})$ where x_{ajt} and y_{alt} are defined below.

$$x_{ajt} = \begin{cases} 1, & \text{if appointment } a \text{ is scheduled for judge } j \text{ to start on timeslot } t \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{alt} = \begin{cases} 1, & \text{if appointment } a \text{ is scheduled for legal assistant } l \text{ to start on timeslot } t \\ 0, & \text{otherwise.} \end{cases}$$

Of each case, three appointments have to be scheduled. The first appointment entails the preparation of the hearing and the appointments are given in the set $\mathcal{P}$, where the set $\mathcal{P}$ consists of $a = 1$ up to and including $a = |\mathcal{P}|$. The set $\mathcal{H}$ contains the to be scheduled hearings and ranges from $a = |\mathcal{P}| + 1$ up to and including $a = 2 \cdot |\mathcal{P}|$. Finally, the set $\mathcal{W}$ contains the to be scheduled appointments for writing of the verdict and ranges from $a = 2 \cdot |\mathcal{P}| + 1$ up to and including $a = 3 \cdot |\mathcal{P}|$. The set of feasible actions compatible with the state $\mathbf{s} \in \mathbb{S}$, denoted by A_s must satisfy the constraints below. Constraints (6.1) and (6.2) ascertain that a new appointment can only be scheduled on a certain timeslot if the judge or legal assistant is not yet scheduled in one of the timeslots in which the appointment takes place (where M_{aj} is the duration of appointment a for judge j and N_{al} is the duration of appointment a for legal assistant l). These durations can be drawn from the uniform distribution as is illustrated in Section 5.1. The constraints thereafter (including the variables and parameters) are reused from the Mixed Integer Linear Program (MILP) and assure that the to be scheduled appointments satisfy the necessary conditions.

$$\sum_{\tilde{a}} x_{\tilde{a}j\tilde{t}} \leq x_{j\tilde{t}} - x_{ajt} \quad \forall \quad a \neq \tilde{a}, j, t < \tilde{t} \leq t + M_{aj} - 1 \tag{6.1}$$

$$\sum_{\tilde{a}} y_{\tilde{a}l\tilde{t}} \leq y_{l\tilde{t}} - y_{alt} \quad \forall \quad a \neq \tilde{a}, l, t < \tilde{t} \leq t + N_{al} - 1 \tag{6.2}$$

$$\sum_{a} \quad x_{ajt} \leq 1, \quad \forall \quad j, t \tag{6.3}$$

$$\sum_{a} \quad y_{alt} \leq 1, \quad \forall \quad l, t \tag{6.4}$$

$$\sum_{\widetilde{l}\widetilde{t}} y_{a\widetilde{l}\widetilde{t}} \leq 2 - y_{alt} \quad \forall \quad a, l, t \tag{6.5}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a, j, t \tag{6.6}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a, j, t \tag{6.7}$$

$$(1 - \tau_a) \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq (1 - \tau_a) \cdot (2 - x_{ajt}) \quad \forall \quad a, j, t \tag{6.8}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a \in \mathcal{H}, j, t \tag{6.9}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{6.10}$$

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{P}, j, t \tag{6.11}$$

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{6.12}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{P}, l, t \tag{6.13}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{H}, l, t \tag{6.14}$$

$$\frac{x_{ajt} \cdot t - \{x_{ajt} \cdot t\} \mod T}{T} = \frac{x_{ajt} \cdot t - x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T}{T} \quad \forall \quad a \in \mathcal{H}, j, t \tag{6.15}$$

$$\sum_{\widetilde{t} < t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}\widetilde{j}\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, j, t \tag{6.16}$$

$$\sum_{\widetilde{t} < t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}\widetilde{j}\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{W}, j, t \tag{6.17}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, l, t \tag{6.18}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{W}, l, t \tag{6.19}$$

$$q_{ajlt} \leq x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{6.20}$$

$$q_{ajlt} \leq y_{alt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{6.21}$$

$$q_{ajlt} \geq x_{ajt} + y_{alt} - 1 \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{6.22}$$

$$q_{ajlt} \leq BL_{l\widetilde{t}} \cdot BJ_{j\widetilde{t}} \cdot BP_{a\widetilde{t}} \quad \forall \quad a \in \mathcal{H}, l, j, t \leq \widetilde{t} \leq t + M_{aj} - 1 \tag{6.23}$$

$$x_{ajt} \leq BJ_{j\widetilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, j, t \leq \widetilde{t} \leq t + M_{aj} - 1 \tag{6.24}$$

$$y_{alt} \leq BL_{l\widetilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, l, t \leq \widetilde{t} \leq t + N_{al} - 1 \tag{6.25}$$

$$\sum_{j} x_{ajt} = \sum_{l} y_{alt} \quad \forall \quad a \in \mathcal{H}, t \tag{6.26}$$

$$I_{js} \cdot \sum_{d} \theta_{ads} \leq 2 - x_{ajt} \quad \forall \quad a, j, t, s \tag{6.27}$$

$$J_{lsd} \cdot \theta_{ads} \leq 2 - y_{alt} \quad \forall \quad a, l, t, s, d \tag{6.28}$$

$$b_a \leq 3 \cdot \sum_{lt} y_{alt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{6.29}$$

$$b_a \leq \sum_{jt} x_{ajt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{6.30}$$

$$b_a \geq \sum_{jt} x_{ajt} - 3 \cdot (1 - \sum_{lt} y_{alt}) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{6.31}$$

$$b_a = 3 \cdot \tau_a \cdot (1 - n_a) + (1 - \tau_a) \cdot (1 - n_a) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{6.32}$$

6.2.3 Transition probabilities

Once actions are taken, the sources of uncertainty in the transition to the next state of the system are the number and the durations of the incoming appointments. As a result of choosing action $\boldsymbol{\alpha}$ in state $\mathbf{s} \in \mathbf{S}$, the state of the system the next day, denoted by $\mathbf{s}' = (x'_{11}, .., x'_{JT}, y'_{11}, .., y'_{LT})$ is defined by the following probability distribution, where p_z is the probability of a case z that brings along an appointment $a \in \mathcal{P}, a + |\mathcal{P}| \in \mathcal{H}$ and $a + 2 \cdot |\mathcal{P}| \in \mathcal{W}$:

$$p(\mathbf{s}'|\mathbf{s}, \boldsymbol{\alpha}) = \begin{cases} p_z, & \text{if } \mathbf{s}' = (x'_{jt}, y'_{lt}, M'_{aj}, N'_{al}, \theta'_{ads}, \tau'_a, n'_a) \text{ satisfies equations (6.33) and (6.34)} \\ 0, & \text{otherwise.} \end{cases}$$

$$x'_{jt} = \begin{cases} x_{j(t+T)} - \sum_{a,t<\tilde{t}\leq t+M_{aj}-1} x_{aj\tilde{t}}, & \forall \quad j, t = 0, 1, 2, .., T \cdot (q-1) - 1 \\ 1, & \forall \quad j, t = q \cdot T, .., (q-1) \cdot T - 1 \end{cases} \tag{6.33}$$

$$y'_{lt} = \begin{cases} y_{l(t+T)} - \sum_{a,t<\tilde{t}\leq t+N_{al}-1} y_{al\tilde{t}}, & \forall \quad l, t = 0, 1, 2, .., T \cdot (q-1) - 1 \\ 1, & \forall \quad l, t = q \cdot T, .., (q-1) \cdot T - 1 \end{cases} \tag{6.34}$$

$$n'_a = n_a - \sum_{a \in \mathcal{P}} b_a \tag{6.35}$$

$$p_z = \prod_{a=1}^{A} P(M'_{aj} = m_k) \cdot P(N'_{al} = n_k) \cdot P(\theta'_{ads} = \Theta_{ads}) \cdot P(\tau_a = 1) \tag{6.36}$$

$$p_s(s_k) = P(S = s_k) = \frac{1}{N}, k = \{1, .., N\} \tag{6.37}$$

$$p_s(s; \lambda) = \frac{e^{-\lambda}\lambda^s}{s!} \text{ for } s = 0, 1, 2, .. \tag{6.38}$$

In equations (6.33) and (6.34) the new parameter q is introduced which denotes the number of days of the rolling horizon, where $q = 5$ in the case of a rolling horizon of one week and T is the number of timeslots in a day. Furthermore, each case must be scheduled for both the judge and the legal assistant and hence the case z can be defined as a six-tuple where $z = (a_j, a_{j+|\mathcal{P}|}, a_{j+2\cdot|\mathcal{P}|}, a_l, a_{l+|\mathcal{P}|}, a_{l+2\cdot|\mathcal{P}|})$. Then, a_j is the duration of the preparation belonging to case z that has to be scheduled for the judge, $a_{j+|\mathcal{P}|}$ is the duration of the hearing of the appointment belonging to case z that has to be scheduled for the judge and $a_{j+2\cdot|\mathcal{P}|}$ is the duration of the writing of the verdict of the appointment belonging to case z that has to be scheduled for the judge. The fourth up to and including sixth element of the tuple z are dealing with the same appointments, but contain possibly different durations, namely the durations that have to be used when scheduling the appointments for the legal assistants. The probability p_z is the product of the probabilities of having a duration for a certain appointment, as shown in equation (6.36), where the distribution of M_{aj} is equal for each j and the distribution of N_{al} is equal for each l and $(m_1, .., m_N)$ and $(n_1, .., n_N)$ contain the values that M_{aj} and N_{al} could take, respectively. Since the durations are drawn from uniform distributions, these probabilities can be calculated using equation (6.37), which shows the probability function of S that has N possible values $(S_1, .., S_N)$, where the possible values are uniformly distributed. Equations (6.33) and (6.34) define the new number of appointment slots that are booked on timeslot t as a function of appointment slots previously booked on timeslot $t + 1$ plus all the new bookings that affected that timeslot, for the judges and legal assistants, respectively. We assume that the arriving cases follow a Poisson process. Hence, the formula of the probability mass function for the Poisson process -given in (6.38)- can be used to determine the number of expected case-arrivals. Finally, we use a uniform distribution for τ_a: whether the case is one-fold or a multiple case. We make the assumption that 10% of the cases arriving is a multiple case [3].

6.2.4 Direct costs

The direct costs associated with choosing action $\boldsymbol{\alpha}$ in state $\mathbf{s}$ arises from two sources: the costs of scheduling an appointment after the norm that is settled for the appointment and the costs of not equally dividing the work over the judges and legal assistants. With the first source, we refer to the hearings that must take place a determined number of weeks (twelve weeks in case of commercial law) after the arrival of that certain case, of which the penalty is modelled in equation (6.40). The verdict must be written a determined number of weeks (six weeks in case of commercial law) after the hearing has taken place, of which the penalty is modelled in equation (6.41). Beneath equation (6.41) the penalty for not equally dividing the work over the judges is calculated. Here, the minimum difference between the available time and the scheduled time for a judge is subtracted from the maximum difference in order to make the difference as small as possible for all judges, of which the formula is given in (6.42). The same way of modelling has been used to model the penalty for not equally dividing the work over the legal assistants, of which the formula is given in (6.43). The final direct costs are then defined by taking the weighted sum over the above mentioned components, where $\beta_1, \beta_2, \beta_3$, β_4 and β_5 are used as weight factors, as is shown in (6.39).

$$c(\mathbf{s}, \boldsymbol{\alpha}) = \beta_1 \cdot c_1(\mathbf{s}, \boldsymbol{\alpha}) + \beta_2 \cdot c_2(\mathbf{s}, \boldsymbol{\alpha}) + \beta_3 \cdot c_3(\mathbf{s}, \boldsymbol{\alpha}) + \beta_4 \cdot c_4(\mathbf{s}, \boldsymbol{\alpha}) + \beta_5 \cdot c_5(\mathbf{s}, \boldsymbol{\alpha}) \tag{6.39}$$

$$c_1(\mathbf{s}, \boldsymbol{\alpha}) = \sum_{a \in \mathcal{H}} [\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T - \gamma_a - \epsilon_1 \cdot T]^+ \tag{6.40}$$

$$\begin{aligned} c_2(\mathbf{s}, \boldsymbol{\alpha}) = \sum_{a \in \mathcal{H}} [\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} - (\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} \mod T) + T \\ -(\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T) - \epsilon_2 \cdot T]^+ \end{aligned} \tag{6.41}$$

$$c_3(\mathbf{s}, \boldsymbol{\alpha}) = \max_j [\sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj}] - \min_j [\sum_t BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj}] \tag{6.42}$$

$$c_4(\mathbf{s}, \boldsymbol{\alpha}) = \max_l [\sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al}] - \min_l [\sum_t BL_{lt} - \sum_{at} y_{alt} \cdot N_{al}] \tag{6.43}$$

$$c_5(\mathbf{s}, \boldsymbol{\alpha}) = \sum_{a=1}^{A} n_a \tag{6.44}$$

6.2.5 Optimal value function

The objective function minimizes the direct costs and expected future costs for this discounted rolling horizon MDP. Since we are interested in determining the optimal stationary policy, we write the objective function in the form of Bellman's optimality equations as follows of the ν-discounted costs associated with state $\mathbf{s} \in \mathbf{S}$:

$$v(\mathbf{s}) = \min_{a \in A} \{c(\mathbf{s}, \boldsymbol{\alpha}) + \nu \sum_{\mathbf{s}' \in S} v(\mathbf{s}')\} \quad \forall \quad \mathbf{s} \in \mathbf{S}$$

6.3 Solution approach

6.3.1 Methodology

Solving the problem is a complex task since the size of the state space grows exponentially with the number of state variables. A number of sophisticated methods for solving these problems have been developed, called ADP [8]. Linear programming algorithms is a method that can be used for discounted infinite-horizon MDPs and can be summarized in the following steps [8]:

1. Write the optimality equations in their linear programming form;
2. Write the value function as a linear combination of basis functions;
3. Formulate the approximate equivalent linear program;
4. Solve the approximate equivalent linear program via constraint sampling or column generation;
5. Compute approximate optimal actions.

The strategy illustrated above is useful if the state space is relatively small, but if the state space becomes larger it becomes impossible to solve the linear program and compute the optimal actions determined with the Bellman equations due to the curse of dimensionality. Therefore, we consider a method to reduce the problem to be able to use ADP.

The large outcome space can be handled through the post-decision state which is defined in Definition 1. The optimality equations can then be rewritten using this post-decision state. A complete sketch of the ADP algorithm using the post-decision state variable for finite horizons can be found in Algorithm 1 [39]. Following standard mathematical notation, each sample path is indexed by the Greek letter ω, where we can call $p_t(\omega)$ a sample realization such that the sequence $p_1(\omega), p_2(\omega), p_3(\omega), ... = \omega^n$ can be referred to as the sample path. By choosing ω at random, randomness is created. Furthermore, we define Ω as a set of all possible sample realizations, where $\omega \in \Omega$. By defining $\hat{\Omega}$ as the set of discrete observations of $\omega \in \Omega$ we can talk about $p(\omega)$ being the probability that we sample ω from within the set $\hat{\Omega}$. W_{t+1} is then defined as the exogenous information becoming available during time interval t [39].

Definition 1. *The post-decision variable is the state of the system after we have made a decision, but before any new information arrives. We denote this by the state S_t^a or the state-action pair (S_t, a_t) [39].*

```
Algorithm 1 Forward dynamic programming using the post-decision state variable [39]
Result: Value functions $(\bar{V}_t^N)_{t=1}^T$
Step 0: Initialization
    (a) Initialize $\bar{V}_t^0, t \in \mathcal{T}$.
    (b) Set $n = 1$.
    (c) Initialize $S_0^1$.
Step 1: Choose a sample path $\omega^n$.
Step 2: Do for $t = 0, 1, 2, .., T$:
    (a) Solve:
            $\bar{v}_t^n = \max_{a_t \in \mathcal{A}_t^n} (C_t(S_t^n, a_t) + \bar{V}_t^{n-1}(S^{M,a}(S_t^n, a_t));$   (6.45)
        and:
            $a_t^n = \operatorname{argmax}_{a_t \in \mathcal{A}_t^n}(C_t(S_t^n, a_t) + \bar{V}_t^{n-1}(S^{M,a}(S_t^n, a_t)).$   (6.46)
    (b) If $t > 0$, update $\bar{V}_{t-1}^{n-1}$ using
            $\bar{V}_{t-1}^n(S_{t-1}^{a,n}) = (1 - \alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{a,n}) + \alpha_{n-1}\hat{v}_t^n.$   (6.47)
    (c) Find the post-decision state
            $S_t^{a,n} = S^{M,a}(S_t^n, a_t^n);$   (6.48)
        and the next pre-decision state
            $S_{t+1}^n = S^M(S_t^n, a_t^n, W_{t+1}(\omega^n)).$   (6.49)
Step 3: Increment $n$. If $n \leq N$, go to step 1.
```

Furthermore, we define $V_t(S_t)$ as the value of being in state S_t just *before* the decision a has been made and $V_t^a(S_t^a)$ is the value of being in state $V_t^a(S_t^a)$ immediately *after* decision a. The function $S^{M,a}(S_t, a_t)$ then takes us from a decision node (pre-decision state) to an outcome node (post-decision state). The use of forward dynamic programming in combination with the post-decision state variable avoids the need to approximate the expectation explicitly within the optimization problem. Since the decision function “sees” the approximation of the value function $V_t(S_t^a)$ directly -instead of indirectly through the approximation of the expectation- we are able to control the structure of the value function [39]. This is especially useful when the problem is an integer problem that requires special structure, as is the case in our problem.

Algorithm 2 ADP with post-decision state applied to our problem

```
Result: Approximation of the value functions $(\bar{V}_d^N)_{d=0}^D$
Initialize $\bar{V}^0(S_d)$ for all states $S_d$;
 Set n = 1;
 Choose initial state $S_0^1$
 for $n = 1$ to $N$ do
     Choose sample path $\hat{\omega}^n$ consisting of a number of appointments (drawn from the Poisson
       distribution) with a duration, specialism and difficulty where these are drawn from a uni-
       form random distribution;
     for $d = 0, 1, ..., D$ do
         Solve the following under the given constraints (as an ILP):
             $\hat{v}_d^n = \max_{a_t \in \mathcal{A}_d^n} -(C(S_d^n, a_d) + \gamma \bar{V}_d^{n-1}(S^{M,a}(S_d^n, a_d)));$   (6.50)
         and
             $a_d^n = \operatorname{argmax}_{a_d \in \mathcal{A}_d^n} -(C(S_d^n, a_d) + \gamma \bar{V}_d^{n-1}(S^{M,a}(S_d^n, a_d)));$   (6.51)
         if $d > 0$ then
             Update $\bar{V}_{d-1}^n(S_{d-1}^{a,n})$ using
                 $\bar{V}_{d-1}^n(S_{d-1}^{a,n}) = (1 - \alpha_{n-1})\bar{V}_{d-1}^{n-1}(S_{d-1}^{a,n}) + \alpha_{n-1}\hat{v}_d^n;$   (6.52)
         end
         Find the post-decision state
             $S_d^{a,n} = S^{M,a}(S_d^n, a_d^n);$   (6.53)
         and the next pre-decision state
             $S_{d+1}^n = S^M(S_d^n, a_d^n, W_{d+1}(\omega^n));$   (6.54)
     end
end
```

We have rewritten the algorithm from Powell [39] in order to fit to our problem (by for instance transferring the maximization problem into a minimization problem by maximizing the negative objective function) and obtained Algorithm 2. Here, $C(s_d^n, a_d)$ are the direct costs (that do not depend on d in our case) and $\bar{V}_d^{n-1}(S^{M,a}(S_d^n, a_t))$ is the approximation of the value function that can be found iteratively and γ represents the discounting factor. At $d = 0$ we look at a horizon of four months. Since we aim on planning every week -and we make use of the rolling horizon- at $d = 1$ we look again at four months, where the first week of the last time period is cut off and one new week is added. These four months have emerged from the norm composed by the "Programma Tijdige Rechtspraak" for a case within commercial law to go from arrival of the case to the completion of the verdict. *Please note:* we have used the d in this algorithm to refer to the length of the horizon, since the t in our state as well as the action refers to a timeslot (of one hour, in our case). In (6.52) α_{n-1} is known as a "stepsize" and generally takes values between 0 and 1. This "stepsize" aims at using observations of noisy data ($\hat{v}^n$) to approximate the mean of the distribution from which the observations are being drawn. This is needed because of the randomness in $\hat{v}^n$ arising from the way we approximated the expectation [39]. We have chosen to define α_n as the deterministic formula $\frac{10}{9+n}$.

6.3.2 Value function approximation

Despite the fact that we have already found a matching algorithm for the solution method, there is still quite a challenge ahead. We still have to design an approximation for the *cost-to-go function* ($\bar{V}_t^n(S_t^{a,n})$ in Algorithm 2) that is computationally tractable and provides a good approximation of the future cost, as a function of the current state [32] [51]. This can be done in several ways [39]:

1. Using lookup tables (with aggregation);
2. Using parametric models;
3. Using nonparametric models.

In order to use lookup tables (with aggregation) the family of aggregation functions $G^g : \mathcal{S} \rightarrow \mathcal{S}^{(g)}$ has to be defined. Here, $\mathcal{S}^{(g)}$ represents the gth level of aggregation of the state space $\mathcal{S}$ and the single aggregation function G maps the disaggregate state $s \in \mathcal{S} = \mathcal{S}^{(0)}$ into an aggregated space $\mathcal{S}^{(g)}$. For further study of the method of aggregation, we refer to Powell [39].

There are many applications where aggregation is naturally hierarchical and each higher level can be represented as an aggregation of the previous level. However, in most cases there is no reason to assume the structure is hierarchical. Also, when we use value function approximation, the issue arises that two different states may have the same behavior -due to the aggregation- despite the fact that the states are different and should perhaps exhibit different behaviors [39]. Besides, the number of possible feature vector increases exponentially with the number of features, which entails that look-up tables are only practical when there are very few features [51].

With parametric models we refer to regression methods such as linear regression, shrinkage methods and support vector regression [39]. In ADP, the independent variables x_i are created using *basis functions* that reduce potentially large state variables into a small number of *features*. Hence, instead of an independent variable x_i we would have a basis function $\phi_f(S)$, where $f \in \mathcal{F}$ is a *feature* [39]. This method has received a tremendous amount of attention in literature and we go further into these parametric models below. However, the power of parametric models is matched by their fundamental weakness: they are only effective if one can design an effective parametric model which could be a frustrating art [39]. Besides, the quality of approximations is determined by the quality of the chosen features [51]. Therefore, nonparametric models have come into play. With nonparametric models we refer to methods that build local approximations to functions using observations rather than depending on functional approximations such as k-nearest neighbor and kernel regression. It is an active area of research that offers tremendous potential, but significant hurdles remain before this approach can be widely adopted [39].

We have chosen to work with the *basis functions* approach, since this approximation strategy works well when the state space and outcome space are large and they are relatively easy to use [32]. Also, we are not dealing with a hierarchical structure which is why we will not deal with lookup tables (using aggregation). We explain the strategy of using basis functions in detail, after which we go further into the basis function used in the game “Tetris” and the basis functions that can be applied to our problem.

Basis functions can be used if particular features of a state vector -that have significant impact on the value function- can be identified. The basis functions are then created for each individual feature that reflects the impact of the feature of the value function. Hence, the chosen features have to be independently separable. We thus define:

$$\mathcal{F} = \text{set of features}$$

$$\phi_f(S_t) = \text{basis function for the feature } f \in \mathcal{F} \text{ for the state } S_t$$

Thereafter, the value function approximation can be defined as in (6.55) where θ_f^n is a weight for each feature $f \in \mathcal{F}$ and $\phi_f(S_t^x)$ is the value of the particular feature $f \in \mathcal{F}$ given the post-decision state S_t^x [32]. Such a feature vector is meant to represent the most salient properties of a given state [51]. The weight ϕ_f^n is updated recursively and the iteration counter is indicated with n [32].

$$\bar{V}_t^n(S_t^x) = \sum_{f \in \mathcal{F}} \theta_f^n \phi_f(S_t^x) \quad \forall \quad t \in \mathcal{T} \tag{6.55}$$

In order to update the estimate of the value of being in a state, we make use of linear models because of their simplicity. For updating the value function approximations, the recursive least squares method is an effective approach [32]. This approximate value iteration algorithm for linear models is captured in Algorithm 3, where the derivation of the formulas (6.59)-(6.63) can be found in Powell [39]. In this algorithm, β is the discount factor, for which we take 0.95. As before, we need to adapt the algorithm a little as we are dealing with a minimization problem, whereas Powell [39] is dealing with a maximization problem; we hence rewrite our minimization problem into a maximization problem by maximizing the objective function multiplied by -1. Furthermore, we initialize G^n by using the zero matrix for G^0 and we initialize the weights θ^n in the value function approximation as $\theta^0 = 1$ for all time periods. Furthermore, since we use a stepsize rule for α_n (namely $\alpha_n = \frac{10}{9+n}$) and λ_n at iteration n should be calculated as stated in (6.58).

In the initial algorithm of Powell, he has used the matrix B which he thereafter replaces by the matrix $G^n = (B^n)^{-1}$ as the discount produced by λ_n was related to the choice of the stepsize rule. We have also chosen a stepsize rule in order to calculate λ_n (namely $\alpha_n = \frac{10}{9+n}$) to reflect the nonstationarity of the observations and therefore we also make use of the matrix $G^n = (B^n)^{-1}$ instead of the matrix B.

Algorithm 3 Approximate value iteration using a linear model [39]

```
Result: Approximation of the value functions $(\bar{V})^N$
Initialize $\bar{V}^0, G^0, \theta^0$ and $S^1$;
for $n$ = 1 to $N$ do
    Solve:
        $\hat{v}^n = \max_{a \in \mathcal{A}} (C(S^n, a) + \beta \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a)))$    (6.56)
    and:
        $a^n = \operatorname{argmax}_{a \in \mathcal{A}} (C(S^n, a) + \beta \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a)));$    (6.57)
    Update the value function recursively using equations (6.59) - (6.63) to obtain $\theta^n$.
        $\lambda_n = \alpha_{n-1} \cdot \frac{1 - \alpha_n}{\alpha_n}$    (6.58)
        $\gamma^n = \lambda_n + (\phi^n)^T (G^{n-1})^{-1} \phi^n$    (6.59)
        $G^n = \lambda_n G^{n-1} + \phi^n (\phi^n)^T$    (6.60)
        $\bar{V}_s(\theta^{n-1}) = \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a^n))$    (6.61)
        $\hat{\epsilon}^n = \bar{V}_s(\theta^{n-1}) - \hat{v}^n$    (6.62)
        $\theta^n = \theta^{n-1} - \frac{1}{\gamma^n} (G^n)^{-1} \phi^n \hat{\epsilon}^n$    (6.63)
    Choose a sample $W^{n+1} = W(\omega^{n+1})$ and determine the next stage using some policy such as:
        $S^n = S^M(S^{n+1}, a^n, W^{n+1})$
end
```

We have encountered a few flaws in this algorithm and have therefore adjusted it by a few points. At first, we have left out the "$+(\phi^n)^T(G^{n-1})^{-1}\phi^n$" for $n = 1$ in (6.59) since one needs the inverse of G^{n-1} and one simply cannot take the inverse of G^0 as G^0 is defined as the zero matrix by Powell [39]. For the same reason, we have left out the "$-\frac{1}{\gamma^n}(G^{n-1})^{-1}\phi^n\hat{\epsilon}^n$" in (6.63). Also, we have defined ϕ^n as a row vector instead of a column vector (as it is defined in [39]) since the matrix dimensions would otherwise not agree, for instance in (6.60) where $\phi^n(\phi^n)^T$ should give a n by n matrix in order to obtain a correct formula. Likewise, because of these matrix dimensions, we have taken the transpose of $(G^n)^{-1}\phi^n$ in (6.63). Powell [39] defines the matrix G as $(B^n)^{-1}$ and when implementing this definition in the original expression for θ^n (including matrix B) we obtain the following formula for θ^n:

$$\theta^n = \theta^{n-1} - \frac{1}{\gamma^n}(G^{n-1})^{-1}\phi^n\hat{\epsilon}^n$$

Hence, we implement this formula in our approximate value iteration instead of the one he has provided in the algorithm. After having implemented the algorithm this way, we obtained a singular matrix G^n for $n \geq 1$ and we have found that there are a few assumptions that should hold in order for this algorithm to work successfully. These can be found in Assumption 1 and 2, where we denote the states by $i = 1, .., n$, the transition probabilities by $p_{ij}, i, j = 1, .., n$, i_k

the state at time k and ϕ is the $n \times s$ matrix that has as rows the feature vectors $\phi(i), i = 1, .., n$ [5]. We elaborate on these assumptions below.

Assumption 1 (Steady-state probabilities)**.** *The Markov chain has steady-state probabilities $\xi_1, ..., \xi_n$, which are positive, i.e. for all $i = 1, .., n$:*

$$\lim_{N \to \infty} \frac{1}{N} \sum_{k=1}^{N} P(i_k = j | i_0 = i) = \xi_j > 0, \quad j = 1, .., n$$

Assumption 2 (Rank ϕ)**.** *The matrix ϕ has rank s.*

The system has a unique solution under conditions that can be somewhat restrictive, for instance the assumption that the Markov chain corresponding to the optimal policy has a unique steady-state distribution with positive components, that the projection norm involves this distribution and that ϕ has linearly independent columns [6]. Because of this rank assumption on ϕ, the inverse of G^n exists for sufficiently large n. As a practical matter, it is common to add a small positive multiple of the identity matrix to G^n to ensure its matrix exists [5]. We have therefore implemented this method in order to obtain an invertible matrix G^n for $n \geq 1$, where we choose 10^{-6} as small positive number by which we multiply the identity matrix. We give the full algorithm, including these updates, in Algorithm 4.

Algorithm 4 Approximate value iteration using a linear model – revised

```
Result: Approximation of the value functions $(\bar{V})^N$
Initialize $\bar{V}^0, G^0, \theta^0$ and $S^1$;
for $n = 1$ to $N$ do
    Solve:
        $\hat{v}^n = \max_{a \in \mathcal{A}} -(C(S^n, a) + \beta \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a)))$    (6.64)
    and:
        $a^n = \operatorname{argmax}_{a \in \mathcal{A}} -(C(S^n, a) + \beta \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a)));$    (6.65)
    Update the value function recursively using equations (6.66) - (6.73) to obtain $\theta^n$.
        $\lambda_n = \alpha_{n-1} \cdot \frac{1 - \alpha_n}{\alpha_n}$    (6.66)
        $\gamma^1 = \lambda_1$    (6.67)
        $\gamma^n = \lambda_n + (\phi^n)^T (G^{n-1})^{-1} \phi^n \quad \forall \quad n > 1$    (6.68)
        $G^n = \lambda_n G^{n-1} + \phi^n (\phi^n)^T + 10^{-6} \mathbb{1}$    (6.69)
        $\bar{V}_s(\theta^{n-1}) = \sum_f \theta_f^{n-1} \phi_f(S^{M,a}(S^n, a^n))$    (6.70)
        $\hat{\epsilon}^n = \bar{V}_s(\theta^{n-1}) - \hat{v}^n$    (6.71)
        $\theta^1 = \theta^0$    (6.72)
        $\theta^n = \theta^{n-1} - \frac{1}{\gamma^n}((G^{n-1})^{-1} \phi^n)^T \hat{\epsilon}^n \quad \forall \quad n > 1$    (6.73)
    Choose a sample $W^{n+1} = W(\omega^{n+1})$ and determine the next stage using some policy such as:
        $S^{n+1} = S^M(S^n, a^n, W^{n+1})$
end
```

We will now gain more insight into basis functions by considering the features used in solving the well-known game Tetris by ADP. Solving Tetris by ADP has been well studied in literature and shows similarities with our problem; in both situations, an object (falling brick or appointment) needs to be placed in a party filled board (already fallen bricks or already scheduled appointments). However, Tetris is a worst-case scenario for the evaluation of automatic control systems in some sense, since humans excel at Tetris [51]. We briefly describe the problem; we refer to [23] for a more detailed description on the game.

A state s in Tetris consists of two components: the description of the board b and the type of the falling piece p. All controllers rely on an evaluation function that gives value to each possible action at a given state. Then, the controller chooses the action with the highest value that encodes the rotation and translation applied to the falling piece [10]. Since the total number of states is large in Tetris (about $m2^{hw}$ where m is the number of different shapes of falling objects, and h and w are the height and width of the grid, respectively [7]) the evaluation function f is usually defined as a linear combination of a set of features $\phi_1, .., \phi_K$. The evaluation function $f_k(\cdot) = \phi_k(\cdot)\theta_k$ specifies the performance by the parameter vector θ. The features used for a state-action pair (s, a) may depend on the description of the board b' resulted from taking action a in state s [23].

In 1996, Bertsekas and Tsisiklis have introduced 22 features that are often used in solving Tetris by ADP with success [47] [10] [33] [22]. For example, Bertsekas and Ioffe [7] have used these features to generate policies that averaged 3183 points a game, which is comparable to an expert human player. The 22 basis functions are [10]:

- Ten basis functions $\phi_0, .., \phi_9$ mapping the state to the height h_k of each of the team columns;
- Nine basis functions $\phi_{10}, .., \phi_{18}$, each mapping the state to the absolute difference between the heights of successive columns: $|h_{k+1} - h_k|, k = \{1, .., 9\}$;
- One basis function ϕ_{19} that maps state to the maximum column height: $\max_k h_k$;
- One basis function, ϕ_{20} that maps state to the number of 'holes' in the wall;
- One basis function, ϕ_{21} that is equal to one at every state.

Scherrer et al. [47] also give the Dellacherie-Thiery (D-T) features which consist of the six features of Dellacherie plus three additional features proposed by Thiery and Scherrer and a constant offset feature. The best policies reported in literature have been learned using the D-T features, i.e. [47]:

- The landing height of the falling piece;
- The row transitions;
- The number of board wells;
- The hole depth;
- The number of rows with holes;
- The pattern diversity feature.

We have gained more insight into possible basis functions, but the assumption that we have a good set of basis functions remains critical, and hard to verify [39]. Hence, also when one has a good understanding of the problem, he or she can only hope to do well by choosing the basis functions carefully so that the linear model has a chance of representing the true value function correctly. By using the above analysis and available experience and intuition about the underlying MDP, we have come up with the following basis functions:

1. The amount of free time periods consisting of one loose block: one single free timeslot. Since we have assumed that the minimum time needed for preparation is four timeslots, empty consecutive timeslots of one, two or three hours can only be filled with hearings and therefore we want to minimize the number of those loose blocks consisting of one, two or three timeslots (by introducing the first three features). The first basis function is calculated per worker (judge and legal assistant) since we are dealing with a relatively small amount of workers compared to the number of specialisms. When there are more workers, it makes sense to calculate these basis functions per specialism and difficulty. We would then get $S + S \cdot D$ outcomes for this basis function, where S is the amount of specialisms and D is the amount of difficulties. The pseudocode to calculate the values of the first basis function is given in Algorithm 5 where $k = 1$; where we use datasets of

judges one up to and including J in order to get $\phi_{1,1}$ up to and including $\phi_{1,J}$. In order to obtain $\phi_{1,J+1},..,\phi_{1,J+L}$ the datasets of legal assistants one up to and including L are inserted in Algorithm 6.

Algorithm 5 Basis function $\phi_k : \phi_{k,1},..,\phi_{k,J}$ for $k = \{1, 2, 3\}$

```
Result: $\phi_k$;
runlength = []
counter = 0
Import vector $x_{jt}$
for $t \in \mathcal{T}$ do
    if $x_{jt} == 0$ then
        counter = counter + 1
    end
    if $x_{jt} == 1$ then
        runlength.append(counter)
        counter = 0
    end
end
$\phi_k$ = runlength.count($k$)
```

Algorithm 6 Basis function $\phi_k : \phi_{k,J+1},..,\phi_{J+L}$ for $k = \{1, 2, 3\}$

```
Result: $\phi_k$ for $k = \{1, 2, 3\}$;
runlength = []
counter = 0
Import vector $y_{lt}$
for $t \in \mathcal{T}$ do
    if $y_{lt}$ == 0 then
        counter = counter + 1
    end
    if $y_{lt}$ == 1 then
        runlength.append(counter)
        counter = 0
    end
end
$\phi_k$ = runlength.count($k$)
```

2. The amount of two consecutive free timeslots in the schedule. The pseudocode to calculate the values of this basis function is given in Algorithm 5 and 6 where $k = 2$; where we again use the dataset of judges one up to and including J in order to get $\phi_{2,1}$ up to and including $\phi_{2,J}$ and the dataset of legal assistant one up to and including legal assistant L in order to get $\phi_{2,J+1}$ up to and including $\phi_{2,J+L}$.

3. The amount of three consecutive free timeslots in the schedule. The pseudocode to calculate the value of this basis function is given in Algorithm 5 and 6 where $k = 3$; where we again use the dataset of judges one up to and including J in order to get $\phi_{3,1}$ up to and including $\phi_{3,J}$ and the dataset of legal assistant one up to and including legal assistant L in order to get $\phi_{3,J+1}$ up to and including $\phi_{3,J+L}$.

4. The number of consecutive nonscheduled time periods. Since more appointments can be scheduled in the lengthier consecutive free time periods, we aim at minimizing the amount

of those consecutive free time periods (since more of those time periods implies that the separate time periods are smaller). The pseudocode to calculate the value of this basis function for the judges one up to and including J (the values $\phi_{4,1},..,\phi_{4,J}$) is given in Algorithm 7. In order to calculate $\phi_{4,J+1},..\phi_{4,J+L}$ the data of the legal assistant one up to and including legal assistant L is inserted in an algorithm similar to Algorithm 7, but then applied for the legal assistants instead of the judges. Hence, it is adjusted in a similar manner as Algorithm 5 is adjusted to Algorithm 6.

Algorithm 7 Basis function $\phi_4 : \phi_{4,1},..,\phi_{4,J}$

```
Result: $\phi_4$;
runlength = []
counter = 0
Import vector $x_{jt}$
for $t \in \mathcal{T}$ do
    if $x_{jt}$ == 0 then
        counter = counter + 1
    end
    if $x_{jt}$ == 1 then
        runlength.append(counter)
        counter = 0
    end
end
$\phi_4 = \sum_{k=1}^{|\mathcal{T}|} runlength.count(k)$
```

5. The first moment when six consecutive timeslots are available. Since we need maximal six timeslots for scheduling a preparation, every case needs a preparation and since we want to schedule cases as early as possible, we aim at having the first moment with six consecutive timeslots as nearby as possible. The pseudocode to calculate the values of this basis function for $\phi_{5,1},.,\phi_{5,J}$ is given in Algorithm 8. Again, an algorithm similar to Algorithm 8 but applied to the data of legal assistants is used in order to obtain $\phi_{5,J+1},..,\phi_{5,J+L}$.

Algorithm 8 Basis function ϕ_5: $\phi_{5,1},..,\phi_{5,J}$

```
Result: $\phi_5$;
counter = 0
Import vector $x_{jt}$
for $t \in \mathcal{T}$ do
    if $x_{jt}$ == 0 then
        counter = counter + 1
        if counter == 6 then
            $\phi_5$ = t - 5
            break
        end
    end
    if $x_{jt}$ == 1 then
        counter = 0
    end
end
```

6. All free timeslots that fall before free timeslots of length greater or equal to four timeslots.

Since a preparation has to take place before the hearing, consecutive free time periods of one two or three timeslots cannot be used if there is no place to schedule the preparation beforehand. The pseudocode to calculate the value of this basis function is given in Algorithm 9. Again, an algorithm similar to Algorithm 9 but applied to the data of legal assistants is used in order to obtain $\phi_{6,J+1}, .., \phi_{6,J+L}$.

Algorithm 9 Basis function $\phi_6 : \phi_{6,1}, .., \phi_{6,J}$

```
Result: $\phi_6$;
runlength = []
counter = 0
Import vector $x_{jt}$
for $t \in \mathcal{T}$ do
    if $x_{jt}$ == 0 then
        counter = counter + 1
        if counter == 4 then
            break
        end
    end
    if $x_{jt}$ == 1 then
        runlength.append(counter)
        counter = 0
    end
end
$\phi_6 = \sum_{t=1}^{|\mathcal{T}|}$runlength.count(t)
```

7. The number of timeslots in which a judge and legal assistant of the same specialism are available. Since we need both a judge and a legal assistant to be available at the same time for a one-fold hearing, we aim at having as much as possible moments on which both are available. In order to calculate the values of this feature, the dataset has to be split up per specialism and Algorithm 10 could then be executed for each of the specialisms.

Algorithm 10 Basis function $\phi_7 : \phi_{7,1}, ..\phi_{7,S}$

```
Result: $\phi_7$;
runlength = []
counter = 0
Import vector $x_{jt}$ and $y_{lt}$
for $t \in \mathcal{T}$ do
    if $x_{jt} = 0$ or $y_{lt} = 0$ then
        counter = counter + 1
    end
    else
        runlength.append(counter)
        counter = 0
    end
end
$\phi_7 = \sum_{k=1}^{|\mathcal{T}|} runlength.count(k)$
```

8. The first timeslot on which a judge and legal assistant of the same specialism are available for two consecutive timeslots. Since we want to schedule hearings as early as possible and we need a judge and legal assistant for a (one-fold) hearing that lasts maximal two

timeslots, we aim at having the first two timeslots on which both a judge and a legal assistant of the same specialism are available as nearby as possible. In order to calculate the values of this feature, the dataset has to be split up per specialism and Algorithm 11 could then then executed for each of the specialisms.

Algorithm 11 Basis function $\phi_8 : \phi_{8,1}, .., \phi_{8,S}$

```
Result: $\phi_8$;
counter = 0
Import vector $x_{jt}$ and $y_{lt}$
for $t \in \mathcal{T}$ do
    if $x_{jt} = 0$ or $y_{lt} = 0$ then
        $counter = counter + 1$
        if $counter == 2$ then
            $\phi_8 = t$
            break
        end
    end
    else
        $counter = 0$
    end
end
```

9. Apart from the above sets of features, we also introduce a constant offset feature. Scherrer et al. [47] found that some features plus the constant offset achieved the best performance in Tetris. Also, Hulshof et al. [32] claim that in case there is no independent constant in the set of predictors $\mathcal{F}$ in a linear regression model, the model is forced to go through the origin and in order to prevent bias resulting therefrom, a constant term should be added as one of the elements in $\mathcal{F}$. Hence, we introduce $\phi_9 = 1$ for every state.

Basis functions themselves do not have to be linear [8], but since these basis functions have to be implemented in the objective function of the ILP in the Python MIP-package, they have to be rewritten into linear constraints. We show how we have addressed this in Appendix F.

Since the basis functions had to be rewritten into linear constraints, we could not implement all the basis functions we originally had in mind. A basis function that could be a good addition to the model is the number of consecutive free timeslots of length ≥ 5. Since these time periods can be used to scheduled a preparation, hearing and a verdict, we aim at maximizing the number of those free time periods. Hence, we want to minimize the negative amount of consecutive free timeslots of length ≥ 5. The pseudocode to calculate the value of this feature is given in Algorithm 12. Again, an algorithm similar to Algorithm 12 but applied to the data of legal assistants is used in order to obtain $\phi_{7,J+1}, .., \phi_{7,J+L}$. Since one needs a lot of constraints in order to implement this basis function (which we show in Appendix F in the ILP -which could slow down the model enormously- we have decided to let the implementation of this basis function be a topic for further research.

Algorithm 12 Basis function $\phi_{10} : \phi_{10,1}, .., \phi_{10,L}$

```
Result: $\phi_{10}$;
runlength = []
counter = 0
Import vector $x_{jt}$
for $t \in \mathcal{T}$ do
    if $x_{jt} == 0$ then
        $counter = counter + 1$
    end
    if $x_{jt} == 1$ then
        $runlength.append(counter)$
        $counter = 0$
    end
end
$\phi_{10} = -\sum_{t=5}^{|\mathcal{T}|} runlength.count(t)$
```

6.4 Uncertainty about the continuation of appointments

As earlier mentioned, some scheduled appointments will not take place. By this we refer to, for example, hearings (and the corresponding appointment for writing of the verdict) that are canceled because the involved parties have reached a solution together before that time. Also, parties can agree on a settlement during the hearing (in Dutch: "schikking"), causing cancellation of the corresponding appointment for the writing of the verdict. In our model, we did not include this uncertainty. One could include this uncertainty by freeing up some timeslots in which preparation or writing of the verdict is scheduled. For this, the information is needed which appointments are scheduled when for which judge and legal assistant instead of only that there is an appointment scheduled for a judge and legal assistant. This is needed because otherwise timeslots in which hearings are scheduled, can be released, what is not aimed for. Then, x_{ajt} and y_{alt} should be included in the state instead of x_{jt} and y_{lt} that ensure an even more enormous state space. Therefore, we have decided to not include this uncertainty in the model itself. This remains a topic for further research. Though, we did think of a way of including this uncertainty when using the model. One could for instance insert more available timeslots for a judge or a legal assistant than that he/she is actually available, since they will get rid of some scheduled timeslots eventually. The newly added timeslots should involve moments on which hearings could never take place, such as evenings, causing that hearings will never be scheduled in these new timeslots.

7 Results Stochastic Dynamic Program commercial law

This chapter elaborates on the results of the developed model, programmed in Python. All tests have been performed on the same system: an Intel Core i5-8265U personal computer with 8 GB of RAM and we have used Python 3 in the Google Colab environment. Section 7.1 discusses the (convergence) performance of the ADP algorithm using a toy-sized problem, after which we compare the performance of our model with the performance using a myopic policy in Section 7.2. We end with testing the model using larger instances in Section 7.3.

7.1 Validating ADP algorithm

Several strategies exist for deciding if the ADP algorithm functions appropriate. These strategies consist of the following [39]:

1. Plotting the objective function over the iterations;
2. Evaluating one or more performance statistics over the iterations;
3. Subjectively evaluating the behavior of the system after the algorithm has completed.

The problem has a large state and action space. Because of this, we use a toy-sized problem to find out whether the ADP algorithm functions appropriately using the strategies mentioned above. We use the following input in order to create the toy-sized problem:

- On average, one case arrives (following the Poisson process);
- We use an optimality gap of 0.05 and a maximum run time of the ILP of three minutes. If the ILP is not able to find a feasible solution in this time, it searches for a new sample path and run the algorithm again. Here, we call a solution feasible if it satisfies the constraints. In addition, its objective value must be within the boundaries of the optimality gap, which we have initialized at five percent;
- We use two judges and two legal assistants;
- We work with one specialism and one difficulty. Naturally, both judges and legal assistants are able to handle this specialism and difficulty.
- We start with a horizon of one week, which can move up two days. This makes the total amount of timeslots 49. Of these 49 timeslots, nine or ten timeslots were assigned unavailable. This was divided in a random manner. Furthermore, we make the assumption that the other parties (prosecutor and defendant) are always available. Since we use this small time horizon, the time one appointment exceeding the norm is left out of the objective function. In the objective function, c_3 and c_4 are multiplied by $\frac{1}{\#timeslots}$ since it has a larger size class than the number of not-scheduled appointments, which we multiply by one. The exact availability that was used is as follows, where 1 defines a timeslot in which a judge j or legal assistant l is available and 0 unavailable, and the availability for the different judges and legal assistants is separated by the brackets:

  ```
  Availability_judges =
      [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
      1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
      [1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
      1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
  ```

```
Availability_legalassistants =
    [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

We started with all basis functions and a small amount of iterations ($n = 50$). We have observed that the algorithm cannot find a solution when the non-available timeslots are more scattered. This is probably caused by the small horizon in which the appointments could be scheduled. Moreover, using this algorithm, the incoming appointments of a certain iteration should be scheduled in the same time horizon. This results in a greater difficulty during scheduling, if the non-available timeslots are more scattered. Appendix G shows the output of the first ten iterations of the algorithm when using the input given above. We observe that the algorithm is able to give a feasible solution most of the time. In the first iteration, no appointments arrive, whereas they could have when cases arrive with an average of one. In the second iteration, one case arrives for which it is optimal to not schedule this in the first horizon ($t = 1$). We see that the algorithm then continues to $t = 2$ and $t = 3$ where it schedules the appointments in the correct order and the hearing on the same time for the judge and the legal assistant.

We then tested a couple of instances, described in Table 2. Since this study only includes one specialism, basis function 7 and 8 will never be used and we research the differences when ex- and including the constant factor (instance 1 versus instance 2). Furthermore, we have made use of a maximum time of finding a solution for the ILP of 180 seconds and a gap of 0.05. We have drawn up some more test instances (instance 3 up to and including instance 7) in order to test the influence of these chosen parameters.

Table 2: Test instances toy-sized problem

Instance	**Used basis functions**	**Optimality gap**	**Maximum duration 1 ILP**
1	1 up to and including 6	0.05	180
2	All (1 up to and including 6 + 9)	0.05	180
3	All (1 up to and including 6 + 9)	0	180
4	All (1 up to and including 6 + 9)	0.1	180
5	All (1 up to and including 6 + 9)	0.5	180
6	All (1 up to and including 6 + 9)	0.05	60
7	All (1 up to and including 6 + 9)	0.05	300

We have set up a few experiments for the first two instances, varying the number of iterations. These experiments, including their run times, are stated in Table 3 (for instance 1) and Table 4 (for instance 2). These run times give an indication, but can slightly differ when using the same input, since the algorithm makes use of Poisson arrivals with an average of one. Hence, there are iterations where more than one case arrives, due to the randomness of the Poisson distribution.

Table 3: Experiments toy-sized problem (instance 1)

Experiment	**Number of iterations**	**Run time**
1.1	50	562 seconds ($\approx$ 9 minutes)
1.2	100	$1,190$ seconds ($\approx$ 20 minutes)
1.3	250	$2,637$ seconds ($\approx$ 44 minutes)
1.4	500	$5,852$ seconds ($\approx$ 98 minutes)

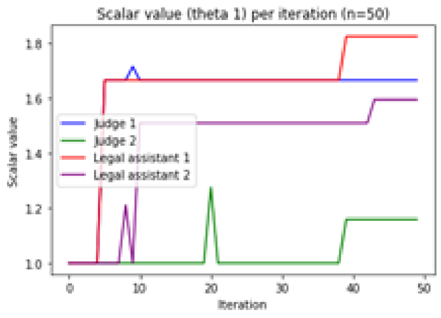
Table 4: Experiments toy-sized problem (instance 2)

Experiment	**Number of iterations**	**Run time**
2.1	50	510 seconds ($\approx$ 9 minutes)
2.2	100	$1,022$ seconds ($\approx$ 17 minutes)
2.3	250	$2,881$ seconds ($\approx$ 48 minutes)
2.4	500	$5,868$ seconds ($\approx$ 98 minutes)

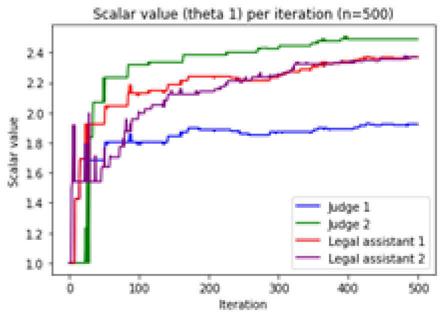
We can thus conclude that the algorithm is able to find solutions within an acceptable time; it takes around 1.5 hours to execute 500 iterations. From the results we conclude that 50 iterations is not enough for the algorithm to converge. For example, Figure 3 shows the progress of the value for θ_1 for the different workers when using all basis functions, for 50 as well as 500 iterations. The algorithm has not converged for $n = 50$, but for 250 iterations, the values seem to become more stable. The values still progress afterwards, but since one needs to make a tradeoff between the run time and the convergence we state that the algorithm is enough stabilised after 250 iterations. Besides, Figure 7b shows the (relative) stabilization of the objective function at 250 iterations.

Figure 7b shows that the algorithm still oscillates when using up to 500 iterations. This is caused by the fact that in every iteration, a new sample path is generated using Poisson arrivals with an average of one. This means that two or more cases can arrive in one iteration (which we also see in Appendix G) that cannot all be scheduled in this short time horizon. Having some non-scheduled appointments subsequently contributes to the (variance of the) objective function.

From Figures 3-6 we conclude that the values for θ_1 up to and including θ_4 continue to develop during the execution of the algorithm, where most of the alterations happen in the beginning (up to 100 iterations). The values for the scalars θ_5 and θ_6 stayed 1 during all iterations. The fifth basis function plots the first moment when six consecutive timeslots are available, which does not change using this toy-sized problem instance with a small time horizon and a small amount of judges or legal assistants. The same applies to the sixth basis function that represents the number of free timeslots that fall before free timeslots of length ≥ 4. Furthermore, the algorithm gives the fourth basis function -that represents the number of consecutive nonscheduled time periods- the highest weight.

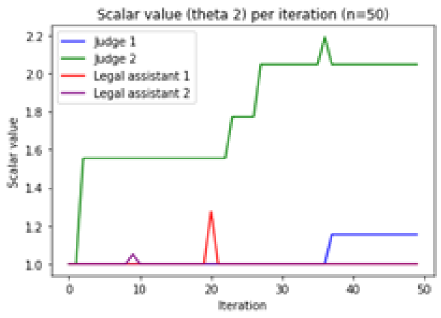


(a) n = 50 (experiment 2.1)

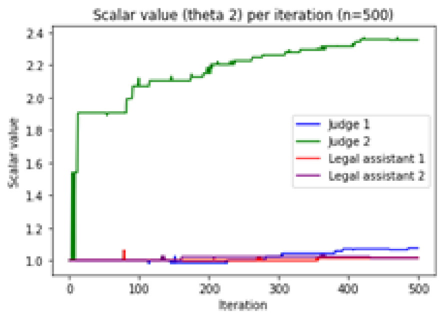


(b) n = 500 (experiment 2.4)

Figure 3: Progress of the values for θ_1 (when using all basis functions)

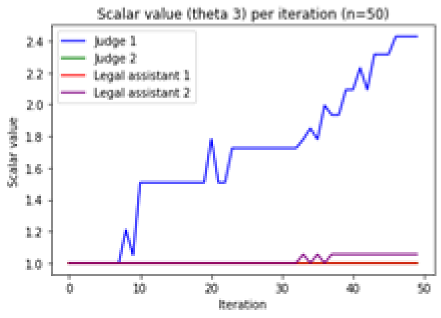


(a) n = 50 (experiment 2.1)

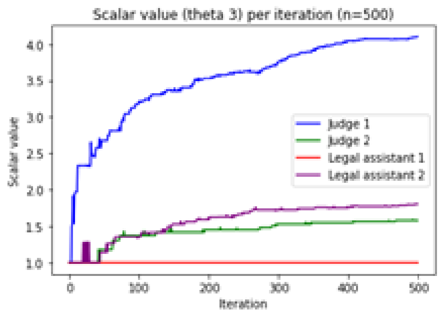


(b) n = 500 (experiment 2.4)

Figure 4: Progress of the values for θ_2 (when using all basis functions)

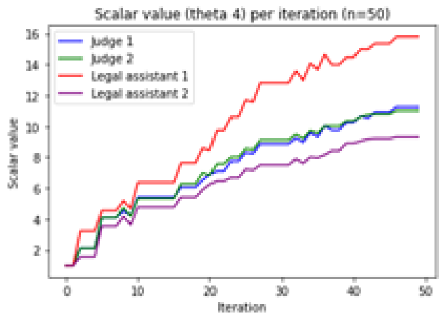


(a) n = 50 (experiment 2.1)

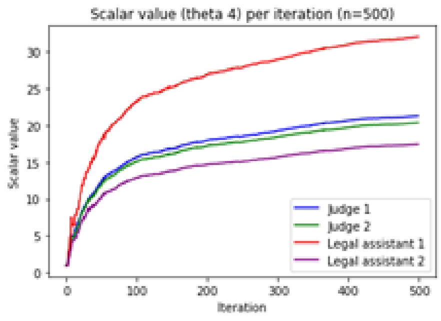


(b) n = 500 (experiment 2.4)

Figure 5: Progress of the values for θ_3 (when using all basis functions)

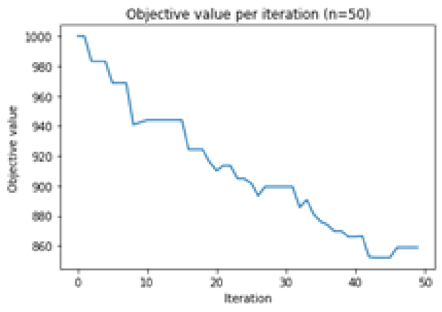


(a) n = 50 (experiment 2.1)

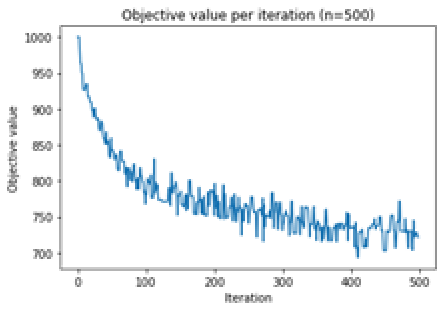


(b) n = 500 (experiment 2.4)

Figure 6: Progress of the values for θ_4 (when using all basis functions)



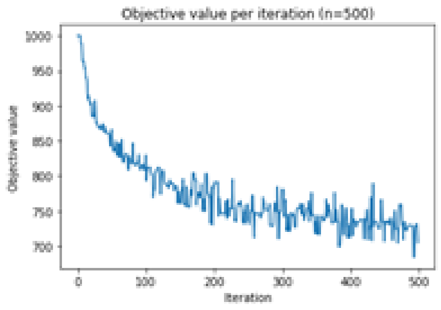
(a) n = 50 (experiment 2.1)



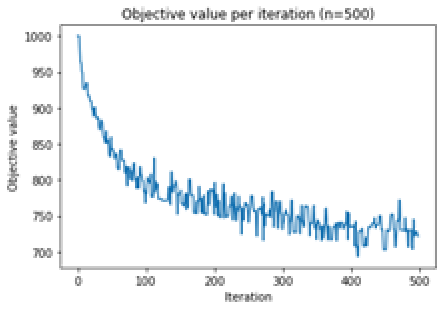
(b) n = 500 (experiment 2.4)

Figure 7: Progress of objective value (when using all basis functions)

Moreover, we can conclude that adding the constant term does not always lead to a better objective. Figure 8 shows that the model including the constant term starts stabilizing sooner than the model without the constant term, but ultimately the objective does not differ significantly. The small difference in the objective value cannot be explained by a small scalar value for this constant basis function. On the contrary, Figure 9 shows that the scalar of the constant basis function was relatively high compared to the other scalars. The small differences in the objective function can be explained by the other scalars: the scalars of the other basis functions were higher for the instance in which the constant basis function was not used, compared to the case where the constant basis function was used. The model including the constant term takes longer to run, which can be seen when comparing Table 3 with Table 4. Because of the mathematical correctness -since the model without constant term is forced to go through the origin- we do not leave this constant term out of the model despite its possible disabilities.



(a) Excluding constant term (experiment 1.4)



(b) Including constant term (experiment 2.4)

Figure 8: Progress of the objective value ($n = 500$)

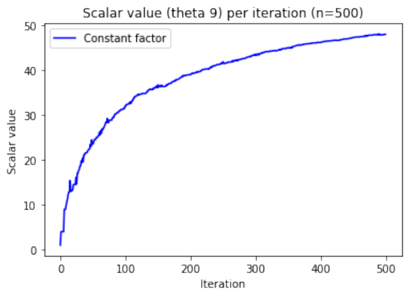
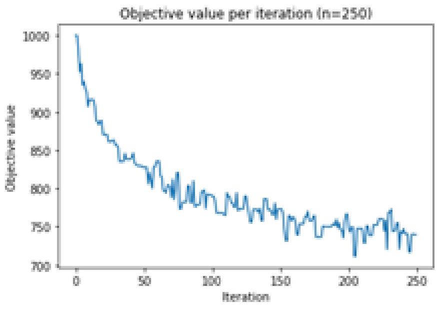


Figure 9: Progress of the constant factor (experiment 2.4)

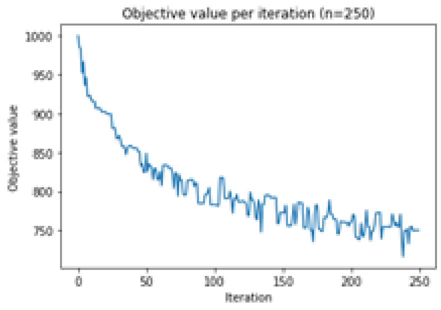
In this study we have also experimented with instances 3 up to and including 7, using 250 iterations. From Table 5 we can conclude that changing the optimality gap does not significantly change the run time (when comparing instances 3, 4 and 5). Figure 10 shows the progress of the objective value for instance 2, 3 and 5, which does not show many differences. From this, we can conclude that the algorithm is able to find the optimal solution within the three minutes we gave the algorithm as maximum duration to execute one ILP. Furthermore, changing the maximum duration to one or five minutes (having an optimality gap of 0.05) does not lead to a change in the run time. Figure 11 shows the progress of the objective for three situations: a maximum duration of 60 seconds per ILP, a maximum duration of 180 seconds per ILP and a maximum duration of 300 seconds per ILP. This confirms that the maximum duration of 180 seconds used initially -for solving one ILP- is enough.

Table 5: Run times toy-sized problem (instance 3 up to and including 7, $n = 250$)

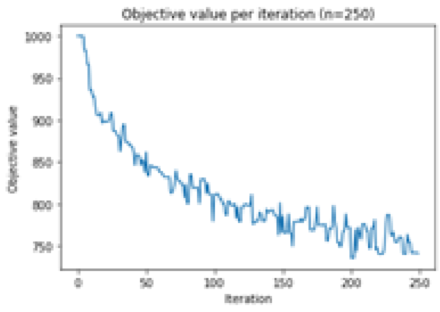
Instance	**Run time**
3	$2,801$ seconds (≈ 48 minutes)
4	$2,677$ seconds (≈ 45 minutes)
5	$2,733$ seconds (≈ 46 minutes)
6	$2,451$ seconds (≈ 41 minutes)
7	$2,468$ seconds (≈ 41 minutes)



(a) Optimality gap = 0.05 (instance 2)

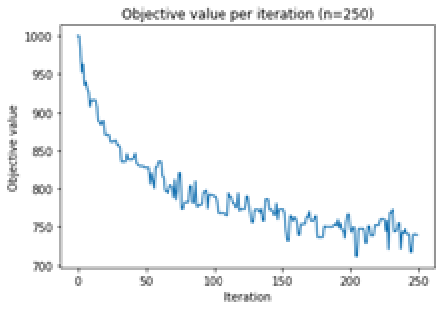


(b) Optimality gap = 0 (instance 3)

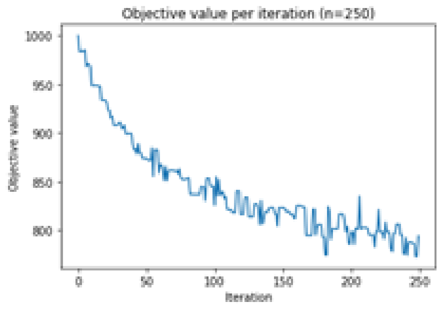


(c) Optimality gap = 0.5 (instance 5)

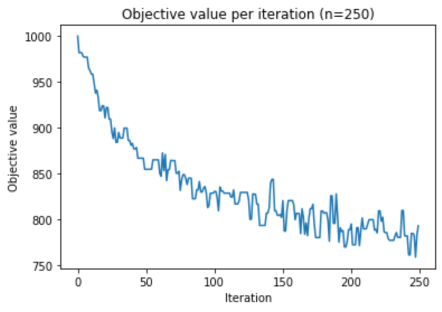
Figure 10: Progress of the objective value (with different optimality gaps, $n = 250$)



(a) Max. duration = 180 seconds (instance 2)



(b) Max. duration = 60 seconds (instance 6)



(c) Max. duration = 300 seconds (instance 7)

Figure 11: Progress of the objective value (with different max. durations for solving one ILP, $n = 250$)

7.2 Comparison myopic policy

Now we have concluded that the ADP-algorithm works properly, we still have to compare the outcomes with the planning strategy used nowadays. The planning strategy used nowadays can be described as a *myopic* policy: a policy that minimizes the appointment starting times. In our model however, we prefer all appointments to be scheduled. Hence, we have implemented our SDP-model, only changing the objective into the following:

$$c(\mathbf{s}, \alpha) = 1000 - (36 \cdot \sum_{a} c_a + \sum_{a,t,j} x_{ajt} \cdot t + \sum_{a,l,t} y_{alt} \cdot t)$$

We have chosen to multiply the variable that represents the number of not-scheduled appointments with 36 as our initial time horizon includes 35 timeslots (causing that $\sum_{a,j,t} x_{ajt} \cdot t$ and $\sum_{a,l,t} y_{alt} \cdot t$ cannot be greater than 35). Furthermore, we want that (slightly) more weight is put on not scheduling appointments compared to the appointment time. We use the parameters of instance 1 and 2 from Section 7.1, with $n = 500$. Since we make use of a self-learning algorithm, we create two situations: one using the initial scalars (all ones) and one using the final scalars (determined by the algorithm). Hence, we test with the two instances, described in Table 6.

Table 6: Test instances toy-sized problem – myopic policy

Instance	**Used data**	**Used scalars**
8	From instance 2 Section 7.1	All ones
9	From instance 2 Section 7.1	By the algorithm determined weights

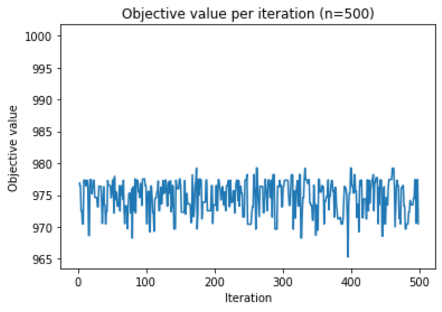
Table 7 shows the run times of these new instances. From this, we conclude that the myopic policy requires less running time; the model only needs around two thirds of the running time compared to the model using the original objective function. This can be explained by the fact that the used scalars are determined beforehand and hence need not be updated every iteration.

Figure 12a and 12b show the initial objective values -using the objective value function of our SDP- when using the myopic policy for instance 8. Figure 12c shows the initial objective values when using a myopic policy for instance 9. Using all ones as scalars naturally gives higher objective values, since the basis functions multiplied with their scalars (in total multiplied with -1) is part of our objective function. We also see that the objective oscillates around the same level for all iterations, which is the case as we are not using a self-learning algorithm anymore. This shows that it is not required to execute more iterations when the weights have already been identified (only in order to adequately compare the run times).

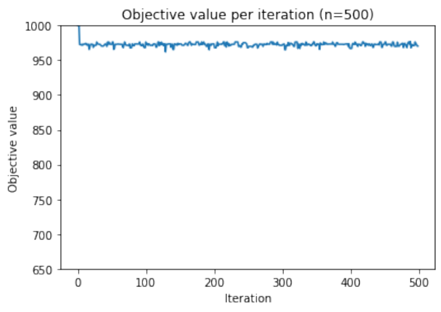
When using the scalars as determined by the ADP-algorithm (instance 9), after 250 iterations, the objective function oscillates between 700 and 800 using our planning strategy and between 650 and 750 using the myopic strategy. These differences are clearly visible in Figure 13, where the objective using our planning strategy is plotted in Figure 13a and using the myopic planning strategy in Figure 13b. Since we are dealing with a maximization problem, we conclude that using our planning strategy gives better results than using a myopic planning strategy.

Table 7: Run times myopic policy (instance 8 and 9, $n = 500$)

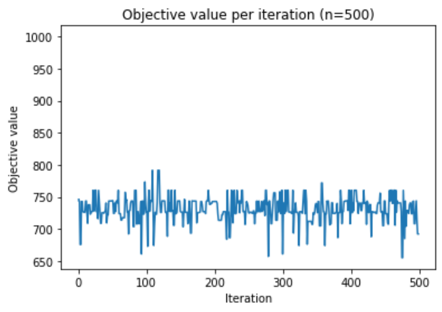
Instance	**Run time**
8	$3,619$ seconds (≈ 60 minutes)
9	$4,313$ seconds (≈ 72 minutes)



(a) Using all ones for the scalars (y-axis ranging from 965 to 1000)

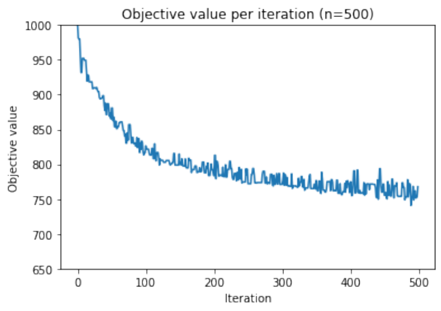


(b) Using all ones for the scalars (y-axis ranging from 650 to 1000, similar to Figure 12c)

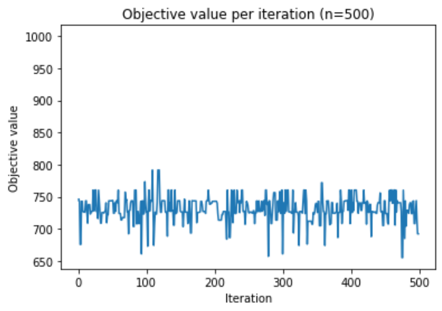


(c) Using the scalars determined by the ADP-algorithm

Figure 12: Progress of the objective value using a myopic strategy (instance 8 and 9, $n = 500$)



(a) Using our planning strategy



(b) Using the myopic policy (with scalars determined by the ADP algorithm)

Figure 13: Progress of the objective value our strategy vs. myopic ($n = 500$)

7.3 Enlarging the state space

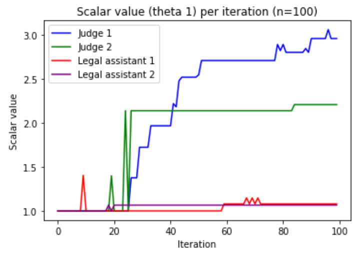
The toy-sized problem we have tested with is smaller than the real-life sized problem. The real-life test data we wanted to use for this case is based on the data from 2019 by the team "commercial law" in Almelo and includes the following:

- A larger time horizon is used, in which multiple cases could be scheduled. The horizon should include at least 20 weeks. This is the horizon one case could occupy, looking at the established norms. The judges and legal assistants are less available in the beginning of the horizon and are becoming increasingly available towards the end of the horizon.

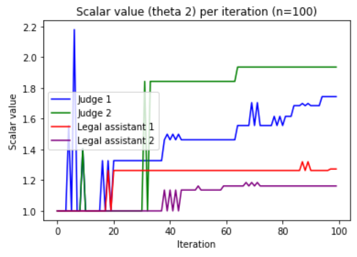
- This test instance includes eight judges and ten legal assistants.
- We would like to use of 27 specialisms and four difficulties, as coincides with the size of clustering in Overijssel. On average, the judges can handle seven specialisms and legal assistants can handle six specialisms. These are both divided randomly, in a way in which every specialism is covered.
  Note: When testing the ILP we have used five specialisms. It has become clear, however, that these specialisms are further divided into 27 so-called "knowledge areas" we wanted to take along these 27 knowledge areas. Therefore, these 27 knowledge areas are also included to simulate a real-life situation as closely as possible.
- On average, sixteen cases (a week) arrive of which ten percent consists of a multiple case.

As we are dealing with an ILP, the run time grows exponentially with the size of the state space. Therefore, we have run the algorithm with a horizon of five weeks instead of the 20 that one case would entail. Of these five weeks, the first week is part of the initial time horizon and the other timeslots can be filled using the rolling horizon. The exact input for the availability can be found in Appendix H. The algorithm is in this case not able to run for 200 or 250 iterations at once, because that would use more RAM than available. Therefore, the algorithm ran for 100 iterations -this took the algorithm $14,913$ seconds: around the four hours- and after that, this was repeated using the values that were just found for the weights as initial weight factor (what took the algorithm $10,718$ seconds: around the three hours). Figure 14 shows the progress of $\theta_1, \theta_2, \theta_3$ and θ_4 for the first 100 iterations. Figure 15 shows the progress of $\theta_1, \theta_2, \theta_3$ and θ_4 when executing the second 100 iterations, using the scalars found in the first 100 iterations (at the end of Figure 14) as initial scalars. Figure 15 shows that the scalars are (almost) stabilised at the end and hence, the algorithm also performs well for a larger time horizon. We also see that the weights are higher and more spread out over the different employees compared to the smaller time horizon. This higher weight is caused by an increase of the values for the fourth, fifth and sixth feature, due to more available timeslots of four or longer after each other, later on in the additional time horizon. Since the values of the first three basis functions stay almost the same, the other functions get relatively more influence. This gets compensated by an increase in the weights of the first three basis functions. Moreover, Figure 17 confirms that one needs to run more than the 100 iterations for the algorithm to be stabilised as the objective value is not stabilised after the 100 iterations. The method of running the algorithm multiple times to determine the weights can be used in real-life in order to determine the weights for each team, as the planning must be made per team and the teams differ in size and formation. Thereafter, the algorithm only needs to be executed for a few iterations in order to determine the planning strategy. Finally, Figure 17 shows that our planning strategy gives better results (as we are dealing with a maximization problem) than the myopic policy.

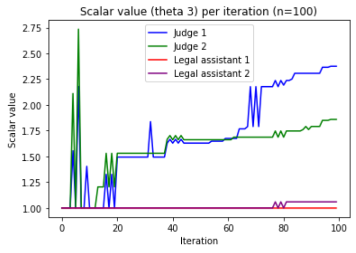
We have also tried to run the algorithm for the full 20 weeks with the found scalars. Because of the fact that the scalars then do not need to be updated anymore, one could reason that running the ADP-algorithm would take up less memory. On the other side, when the state space will be increased, executing one ILP -for which we use the Python MIP-package- will use more memory as the values for ϕ need to be determined over all possible timeslots. After experimenting with different time horizons and leaving the rest of the input the same as for the toy-sized problem, we can conclude that the model is able to solve instances with a time horizon up to 80 days (i.e. 560 timeslots or around the eleven weeks) of which one week (i.e. 35 timeslots) is part of the initial time horizon. When expanding this horizon with one week, we encountered the memory issues. This problem could be solved by using a device with more RAM or examining ways in order to shrink the state space. Furthermore, as we make use of the MIP-package, other implementations can be examined that possibly need less memory to solve one ILP.



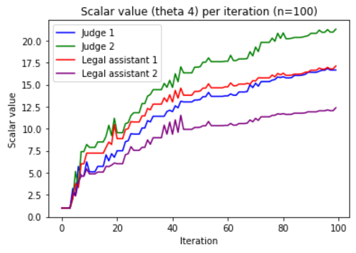
(a) θ_1



(b) θ_2

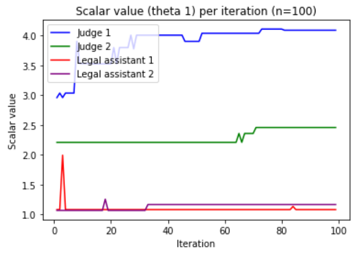


(c) θ_3

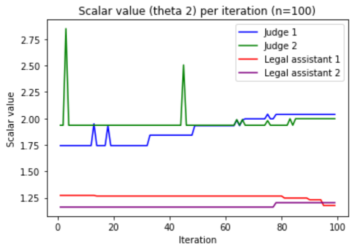


(d) θ_4

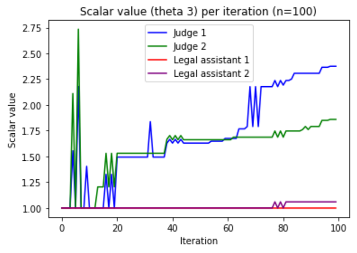
Figure 14: Progress of the values for $\theta_1, \theta_2, \theta_3$ and θ_4 when using a larger time horizon ($n = 100$)



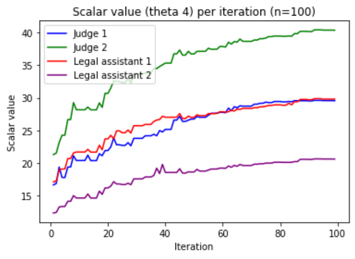
(a) θ_1



(b) θ_2

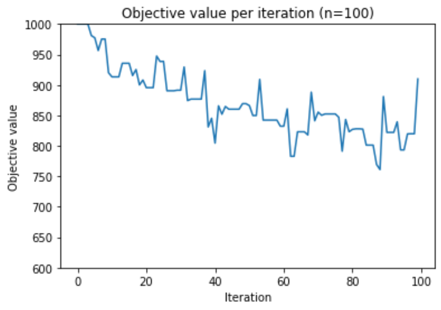


(c) θ_3

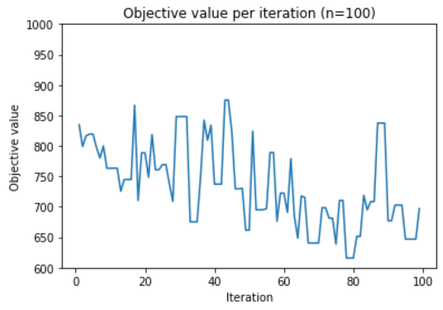


(d) θ_4

Figure 15: Progress of the values for $\theta_1, \theta_2, \theta_3$ and θ_4 when using a larger time horizon with updated initial scalars ($n = 100$)

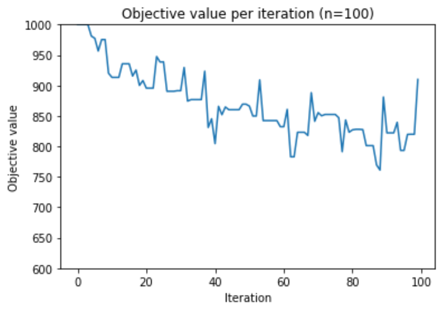


(a) Using our planning strategy

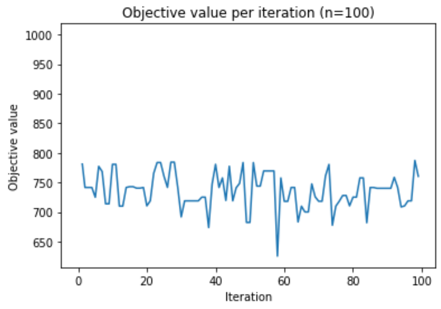


(b) Using our planning strategy (with updated initial scalars)

Figure 16: Progress of the objective value (executing the algorithm two times for $n = 100$)



(a) Using our planning strategy

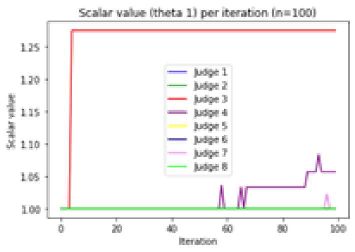


(b) Using the myopic policy (with scalars determined by the ADP algorithm)

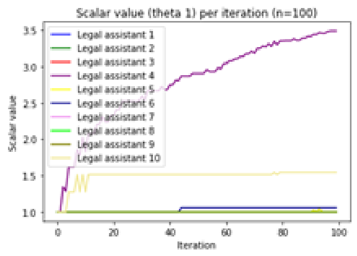
Figure 17: Progress of the objective value ($n = 100$)

When having eight judges and ten legal assistants (the used availability is given in Appendix H where the rest of the used input is the same as for the toy-sized problem), the model could not run for 200 or 250 iterations at once, because of the already mentioned memory issues. In order to solve this problem, we have executed the algorithm twice for 100 iterations in order to obtain stabilised values for the scalars. It took the solver $17,213$ seconds (approximately five hours) to run the first 100 iterations and $20,734$ seconds (approximately six hours) to run the second 100 iterations. One would thus need around eleven hours in order to obtain the values for θ, which is more then eleven times as long as one needs for the toy-sized problem (with two judges and two legal assistants). Part of this increased run time can be explained by the fact that the algorithm has to update a lot more values for θ. For example, Figure 18 shows the progress of the values for θ_1, θ_2 and θ_4 for both the judges as well as the legal assistants (where $n = 100$). For the fourth basis function, every θ is updated for every iteration. Not all values for θ_1 and θ_2 were updated every iteration, but the values of θ_3, θ_4 and θ_5 were (often) not updated at all. Moreover, we are now working with a larger state and action space.

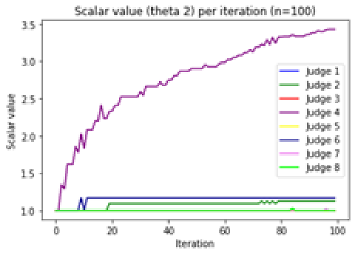
Because of the exponentially increasing running time when increasing the state space, we have tested with six specialisms (instead of the 27 knowledge areas used in Overijssel) and four difficulties as this coincides with the size of the clustering in Rotterdam. Here, the judges and legal assistants could handle four out of six specialisms and the legal assistants could handle three out of four difficulties. When using the toy-sized problem with the addition of these specialisms and difficulties, it took the model $3,641$ seconds (≈ 61 minutes) to run 250 instances, which took the algorithm without specialisms approximately 48 minutes. When enlarging the time horizon as well as adding the specialisms compared to the toy-sized problem, it took the model eight hours in order to execute 50 iterations. Since the problem is too big to run 100 or more iterations at once, one would need to execute the algorithm four times for 50 iterations in order to initialize the weights for this larger instance. Hence, in total one needs 32 hours in order to initialize the weights for such a large instance, which is why we have not experimented with problems that big.



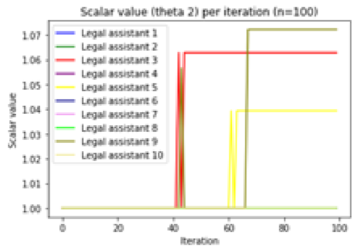
(a) θ_1 judges



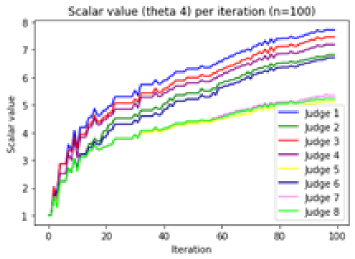
(b) θ_1 legal assistants



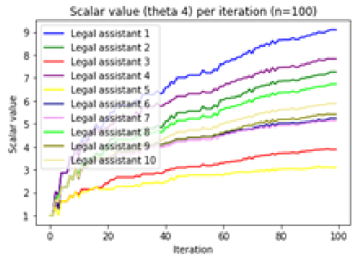
(c) θ_2 judges



(d) θ_2 legal assistants



(e) θ_4 judges



(f) θ_4 legal assistants

Figure 18: Progress of the values for θ_1, θ_2 and θ_4 for the judges and the legal assistants ($n = 100$)

We have also researched the influence of having two cases arrive instead of one as well as having ten percent of the cases arriving a multiple case, separately. The remainder of the input used is similar to the input for our toy-sized problem. The addition of multiple cases did -in contrast- not lead to larger running instances nor memory problems. At last, we have run the instance where on average two cases arrived instead of one. It took our model $41,439$ seconds (approximately 12.5 hours) to solve this problem, using 50 iterations. Comparing this to the run time of the problem with one case arriving for 50 iterations (which took approximately 2 hours) we conclude that the run time grows enormously when more appointments arrive. Since more appointments arrive within the same time horizon, the algorithm was not able to find a solution in more iterations, which causes the execution of some iterations multiple times, ensuring this long running time.

8 Limitations and further research

Our problem statement and model have certain limitations that provide directions for future research. Section 8.1 discusses these limitations. Subsequently, we give the recommendations for further research in Section 8.2.

8.1 Limitations

To begin with, we have looked at the planning of subpoenas (in Dutch: 'dagvaardigingen') within the courts, where summary proceedings (in Dutch: 'kort gedingen') have to be scheduled within the same team. Subpoenas concern formal written orders issued by a court that requires a person to appear in court and testify, or produce documents. Summary proceedings concern such procedures without the formalities for the speedy disposition of a case. For the summary proceedings, other norms apply which we did not take into account. Furthermore, we have made some simplifications when building the ILP, which we discussed in Section 4.5 and also apply to our MDP.

Secondly, we did not take into account the uncertainties that could occur after the planning has been created. These uncertainties refer to the fact that cases may be cancelled any time during the process. For example, before the hearing, parties may already have reached a solution together, causing the hearing to be cancelled. In this case, a verdict does not need to be written. Another reason for cancelling is that parties could agree on a settlement during the hearing. Again, no verdict is written here. The models do not include these uncertainties; we have further elaborated on this limitation in Section 6.4.

Thirdly, some decisions that were made in the modeling phase, were made without knowing its direct consequences. For instance, we had to choose what the decision variable would be: the starting time of an appointment or the time one is working on the appointment. We have chosen the decision variable that decides on the starting time. This decision was made because the problem at hand is relatively big and otherwise, the decision space would grow enormously. Choosing this decision variable does limit the model, since an appointment can then only take place in sequence and the duration cannot be cut up. Using the same reasoning, we have decided not to implement the appointment type in the state of the MDP. At last, we could not take into account the penalty for scheduling after the established norms, since we could not solve problems of such a large time horizon.

8.2 Further research

When continuing this research, data has to be obtained such that the durations of the appointments (to be scheduled) are based on real-life data of appointment durations. Therefore, this remains a topic of further research. This includes data regarding durations of preparing a hearing and writing the verdict. Moreover, data has to be obtained relating to the arrivals of cases: historical data on when cases of certain specialisms and difficulties have arrived. Only when this data is available, one can test whether such a planning method (as proposed in this research) yields benefits in practice. Below we present a more detailed list with the required data:

- The duration of the preparation of a case (in minutes or hours) per specialism, difficulty and judge or legal assistant. This way, one can compose a distribution for each judge and legal assistant. This can then be used in order to schedule the preparation of a new incoming appointment for the corresponding judge or legal assistant;

- The duration of the writing of the verdict of a case. For the writing of the verdict, a distribution will have to be drawn up, using the same parameters as for the preparation of the case;
- The distribution of the duration of the preparation of a hearing and the writing of the verdict for the different judges dealing with a multiple case. When dealing with a multiple case, one judge is the main person responsible and has to spend more time preparing the case and writing the verdict. Data has to be collected in order to determine this division. Since this data was not available during our research, we made the assumption that the distribution of the duration of multiple cases is for each judge similar to the distribution of the duration for one-fold cases;
- The exact date and time of incoming appointments, kept up per specialism and difficulty. This way, a distribution can be drawn for the arrival of cases;
- The amount (and type) of cases that did not yield a hearing or a writing of the verdict over a horizon of a couple of months. Nowadays, this is not tracked precisely. Some of the cases that did not yield a hearing or writing of the verdict could be tracked, but the specialism and difficulty of these cases is not tracked anywhere. When one keeps track of this, for instance for a few months, one can forecast the uncertainty of the hearing and writing of the verdict and this uncertainty could then be integrated in the model.

Furthermore, the basis functions are constructed based on the assumptions regarding the durations of appointments and must therefore be reconsidered when these durations are specified by real-life data. This also creates a ground for further research. For instance, when the preparation of a hearing appears to last at least three hours (instead of the four hours we used) the third basis function does not add value anymore.

Another recommendation is to investigate the input and output used for the planning, for the different jurisdictions as well as the different teams (situated on a certain location). We have identified a lot of differences as well as similarities between the different jurisdictions and different courts of which the courts were not aware of. For example, the team situated in Rotterdam makes use of six specialisations, whereas Overijssel makes use of 27 so-called *knowledge areas*. Furthermore, some members of the team of administrative law were unaware of the fact that the teams of commercial law do not use an initial schedule. These differences and similarities have not been mapped. Considering the fact that almost everyone stays within their own team at the Rechtspraak, we recommend to map these differences and similarities. When these differences and similarities have been mapped, one can specify the modifications that have to be made to the basic model in order for the model to be valid for other teams and/or jurisdictions. Furthermore, this can also work the other way around: teams can adapt their planning strategy in order to use the model.

We have also concluded that the real-life sized problems contain a large state and action space and require a lot of memory to solve. Therefore we recommend to research ways in order to shrink the size of the state and/or action space, for instance by cutting the problem into smaller problems, such as one for each specialism, and solving them separately. We have seen that the solver consumes notably more memory for solving a problem including different specialisms compared to solving the same problem without these specialisms. Hence, cutting the problem into smaller problems based on specialisms and solving them separately could be a valid (first) step in order to solve larger problem instances.

Another topic for further research concerns the current set-up of the ADP algorithm. This set-up could be improved. First of all, the current set-up of the ADP algorithm is single pass, which means that at each step forward in time in the algorithm, the value function approximations are updated [32]. The ADP can also be used with a double pass approach, in which the algorithm first simulates observations and computes decisions for *all* time periods in one iteration, before updating the value function approximations. This double pass approach may lead to a faster convergence of the ADP algorithm and therefore remains a topic for further research. At last, when using this algorithm, all appointments arriving in a certain iteration have to be scheduled in the same horizon. When continuing research, one could investigate whether it is possible to adapt the algorithm in order to be able to schedule different appointments belonging to the same case in different horizons (so for example the preparation in $t = 1$ and the hearing in $t = 2$).

Apart from the possible improvements of our MDP one could also analyze the other solution methods, such as transforming the stochastic program into a two-stage problem which can then be solved using the SAA approach, as proposed in literature and discussed in Section 2. Likewise, one could research whether methods other than ADP for solving the MDP could be appropriate for this problem instance. One could then think of heuristic methods or using ALP, which are other possible methods proposed in literature in order to solve MDPs.

9 Conclusions and recommendations

The aim of this research was to compose an efficient planning and scheduling strategy for the teams commercial and administrative law. Commercial law uses a different strategy than administrative law: in administrative law, the schedulers make an initial schedule in which hearings should take place, whereas commercial law starts planning without initial schedule. This difference arises from the fact that hearings within commercial law take around 1.5 hours and could therefore be planned per case, whereas hearings within administrative law last a shorter time (for instance, the treatment of one case could take only ten minutes) and have to be bundled in one hearing. This research focuses on commercial law and the created model needs to be adjusted or extended in order to be valid for administrative law; we give a first draft of the extensions that are (at least) needed for administrative law in Appendix E. Since we have focused on commercial law, we have focused on optimizing the planning strategy, not dealing with the composition of an initial schedule. To do this, we started with an ILP in order to properly select the variables and constraints that are needed in order to solve this problem. After programming this ILP in AIMMS, there is still an optimality gap of around 45 percent after running the algorithm for half an hour, with a horizon of only two weeks. This differs from real life, where one is dealing with a timespan of around 20 weeks to cover one case from arrival of the case to completion of the verdict. Since this ILP does not deal with appointments that arrive in the future, we constructed an MDP, making use of the already drafted constraints of the ILP.

Our MDP incorporates uncertainty in two key processes, namely the arrival of the cases and the duration of the appointments. We present an ADP approach using post-decision states and a rolling horizon. This ADP approach can be used to solve a toy-sized problem instance in a reasonable time. The results of using our planning strategy compared to the myopic policy indicate that the ADP performs better than the myopic policy for the toy-sized problem as well as when enlarging the time horizon.

During the research some memory problems arose when inserting large problem instances. In this thesis we have presented different ways in order to deal with these memory issues, such as executing the algorithm twice. Here, the scalars found in the first execution are used as initial scalars when running the algorithm a second time. We have shown that using this method has led to stabilized values of θ as well as better results, compared to when using a myopic policy. Besides, when the final scalars have been determined, the algorithm only needs a few iterations in order to solve the problem, as the algorithm is not self-learning anymore.

The real-life sized instance presented in this thesis is developed based on data of the teams situated in Rotterdam and Overijssel. As a result, the instances used are of comparable network structure and size of other courts in the Netherlands. Our study incorporates two courts, but only small adjustments are needed for the model to be workable in commercial law teams of other courts. Moreover, the developed model is generic as the objective function can be adopted when different courts would use different norms, or when they have another aim when planning the different cases. For the different jurisdictions, constraints might have to be removed or added, but the principle of the algorithm, scheduling three appointments per case -taking into account the uncertainty in arrival of cases and duration of appointments- can be extended to the other jurisdictions.

Section 8 has stated some limitations of the model and includes possibilities for further research. Further research mainly involves acquiring data: data on the durations of different appointments as well as the arrivals of cases and the uncertainty when cases are scheduled. Reducing the state space is also an important topic for further research. When this data is available and the state space has been reduced, one can revise the algorithm based on the data and test the algorithm for real-life problem instances. Another possibility includes expanding the algorithm such that appointments arriving in the same iteration could be scheduled spread out over different time horizons.

We end with a few recommendations regarding the organisation of such a project within the Rechtspraak. We think that the project team needs to make clear what they would like to achieve with such a project. We have looked for *similar* problem instances in healthcare, covering offline, online as well as stochastic online solvers. When continuing research with a clearer view of how the Rechtspraak is willing to use such a solver, one can search for more specified literature. Furthermore, it would really help if there is one person who knows what data is available and where this is available. When the Rechtspraak obtains an overview of the available data and releases the indicated required data, one only has to take a few steps in order to test this model on real-life data. As the required steps are described in this thesis, one solely has to follow these steps in order to research the *real-life profit* such a model can entail. All in all, we have laid a good foundation for a working model, offering promising perspectives for continuation and implementation of this research.

References

[1] Albareda-Sambola, M., van der Vlerk, M. and Fernández, E. Exact solutions to a class of stochastic generalized assignment problems. *European Journal of Operational Research*, 173(2):465–487, 2006.

[2] Artigues, C., Leus, R. and Nobibon, F. Robust optimization for resource-constrained project scheduling with uncertain activity durations. *Flexible Services and Manufacturing Journal*, 25(1-2):175–205, 2013.

[3] Baas, R., de Groot-van Leeuwen, L. and Laemers, M. *Rechtspreken: samen of alleen: Over meervoudige en enkelvoudige rechtspraak*. Sdu Uitgevers, 2010.

[4] Barz, C. and Rajaram, K. Elective patient admission and scheduling under multiple resource constraints. *Production and Operations Management*, 24(12):1907–1930, 2015.

[5] Bertsekas, D. *Dynamic Programming and Optimal Control: Volume II, Third Edition*. Athena Scientific, Nashua (NH), United States of America, 2007.

[6] Bertsekas, D. Feature-based aggregation and deep reinforcement learning: A survey and some new implementations. *IEEE/CAA Journal of Automatica Sinica*, 6(1):1–31, 2019.

[7] Bertsekas, D. and Ioffe, S. Temporal differences-based policy iteration and applications in neuro-dynamic programming. *Report LIDS-P-2349*, pages 1–19, 1996 (Revised 1997).

[8] Boucherie, R. and Dijk, van, N. *Markov Decision Processes in Practice*. Springer International Publishing AG 2017, Cham, Switzerland, 2017.

[9] Braaksma, A., Kortbeek. N., Post, G. and Nollet, F. Integral multidisciplinary rehabilitation treatment planning. *Operations Research for Health Care*, 3(3):145–159, 2014.

[10] Calafiore, G. and Dabbene, F. *Probabilistic and Randomized Methods for Design under Uncertainty*. Springer-Verlag, London, United Kingdom, 2006.

[11] Cardoen,B., Demeulemeester, E. and Beliën, J. Operating room planning and scheduling: A literature review. *European Journal of Operations Research*, 201(3):921–932, 2009.

[12] Condotta, A. and Shakhlevich, N. Scheduling patient appointments via multilevel template: A case study in chemotherapy. *Operations Research for Health Care*, 3(3):129–144, 2014.

[13] Conforti, D., Guerriero, F. and Guido, R. Optimization models for radiotherapy patient scheduling. *4OR: A Quarterly Journal of Operations Research*, 6(1):263–278, 2008.

[14] Creemers, S. The Resource-constrained Project Scheduling Problem with Stochastic Activity Durations. *IEEE International Conference on Industrial Engineering and Engineering Management*, pages 453–457, 2014.

[15] Creemers, S. Minimizing the expected makespan of a project with stochastic activity durations under resource constraints. *Journal of Scheduling*, 18(3):263–273, 2015.

[16] Creemers, S. The preemptive stochastic resource-constrained project scheduling problem. *European Journal of Operational Research*, 277(1):238–247, 2019.

[17] Creemers, S., Leus, R. and Lambrecht, M. Scheduling Markovian PERT networks to maximize the net present value. *Operations Research Letters*, 38(1):51–56, 2010.

[18] Deblaere, F., Demeulemeester, E. and Herroelen, W. Proactive policies for the stochastic resource-constrained project scheduling problem. *European Journal of Operational Research*, 214(2):308–316, 2011.

[19] Escudero, L., Garín, A., Monge, J. and Unzueta, A. Some matheuristic algorithms for multistage stochastic optimization models with endogenous uncertainty and risk management. *European Journal of Operational Research*, 285(3):988–1001, 2020.

[20] Fan, X., Tang, J. and Yan, C. Appointment scheduling optimization with two stages diagnosis for clinic outpatient. *Computational Statistics*, 35(1):469–490, 2020.

[21] Franz, L. and Miller, J. Scheduling Medical Residents to Rotations: Solving the Large-Scale Multiperiod Staff Assignment Problem. *Operations Research*, 41(2):269–279, 1993.

[22] Furmston, T. and Barber, B. A unifying perspective of parametric policy search methods for markov decision processes. *Proceedings of the Advances in Neural Information Processing Systems*, 4(1):2726–2734, 2012.

[23] Gabillon, V., Ghavamzadeh, M. and Scherrer, B. Approximate dynamic programming finally performs well in the game of tetris. *Advances in Neural Information Processing Systems*, Conference paper (27th Annual Conference on Neural Information Processing Systems, NIPS 2013; Lake Tahoe, NV; United States):1–9, 2013.

[24] Gutjahr, W. Bi-Objective Multi-Mode Project Scheduling Under Risk Aversion. *European Journal of Operational Research*, 246(2):421–434, 2015.

[25] Güler, M. A hierarchical goal programming model for scheduling the outpatient clinics. *Expert Systems With Applications*, 40(12):4906–4914, 2013.

[26] Gür, S. and Tamer, E. Application of Operational Research Techniques in Operating Room Scheduling Problems: Literature Overview. *Journal of Healthcare Engineering*, 2018(4):1–15, 2018.

[27] Habibi, F., Barzinpour, F. and Sadjadi, S. Resource-constrained project scheduling problem: review of past and recent developments. *Journal of Project Management*, 3(2):55–88, 2018.

[28] Halman, N. and Nannicini, G. Toward breaking the curse of dimensionality: an FPTAS for Stochastic Dynamic Programs with multidimensional actions and scalar states. *SIAM Journal on Optimization*, 29(2):1131–1163, 2018.

[29] Hartmann, S. *Project scheduling under limited resources: Models, methods, and applications.* Springer: Number 478 in Lecture Notes in Economics and Mathematical Systems, Berlin, Germany, 1999.

[30] Hartmann, S. and Briskorn, D. A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operations Research*, 207(1):1–14, 2010.

[31] Homen-deMello, T., Matos, V. and Finardi, E. Sampling strategies and stopping criteria for stochastic dual dynamic programming: a case study in long-term hydrothermal scheduling. *Energy Syst*, 2(1):1–37, 2011.

[32] Hulshof, P., Mes, M., Boucherie, R. and Hans, E. Patient admission planning using Approximate Dynamic Programming. *Flexible Services and Manufacturing Journal*, 28(1):30–61, 2016.

[33] Kakade, S. A natural policy gradient. *Conference paper: Neural information processing systems foundation*, 2002.

[34] Ke, H. and Liu, B. Project scheduling problem with stochastic activity duration times. *Applied Mathematics and Computations*, 168(1):342–353, 2005.

[35] Kopinga, W. and Andringa, R. Duizenden strafzaken vertraagd door coronacrisis. 'Dit kan zo niet langer'. `https://nos.nl/artikel/2329219-duizenden-strafzaken-vertraagd-door-coronacrisis-dit-kan-zo-niet-langer.html`, 2020. Last visited on 03-04-2020.

[36] Manthey, B. *Optimization Modeling.* University of Twente, Enschede, 2018.

[37] Marynissen, J. and Demeulemeester, E. Literature review on multi-appointment scheduling problem in hospitals. *European Journal of Operational Research*, 272(2):407–419, 2019.

[38] Pentico, D. Assignment problems: A golden anniversary study. *European Journal of Operational Research*, 176(2):774–793, 2007.

[39] Powell, W. *Approximate Dynamic Programming: Solving the Curses of Dimensionality (Second Edition).* Wiley, United States of America, 2011.

[40] Puterman, M. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, United States of America, 1994.

[41] Pérez, E., Ntaimo, L, Malavé, C., Bailey, C. and McCormack, P. Stochastic online appointment scheduling of multi-step sequential procedures in nuclear medicine. *Health Care Management Science*, 16(4):281–299, 2013.

[42] Rafiee, M., Kianfar, F. and Farhadkhani, M. A multistage stochastic programming approach in project selection and scheduling. *The International Journal of Advanced Manufacturing Technology*, 70(1):2125–2137, 2014.

[43] Reyes, A., Sucar, E., Ibargüengoytia, P. and Morales, E. Planning under uncertainty applications in power plants using factored Markov Decision Processes. *Energies*, 13(9):1–17, 2020.

[44] Roland, B., Di Martinelly, C. and Riane, F. Operating theatre optimization: A resource-constrained based solving approach. *IEEE Computer Society. Conference Paper: International Conference on Service Systems and Service Management*, 1(n/a):443–448, 2006.

[45] Rostami, S., Creemers, S. and Leus, R. New strategies for stochastic resource-constrained project scheduling. *Journal of Scheduling*, 21(1):349–365, 2018.

[46] Sauré, A., Begen, M. and Patrick, J. Dynamic multi-priority, multi-class patient scheduling with stochastic service times. *European Journal of Operational Research*, 280(1):254–265, 2020.

[47] Scherrer, B., Ghavamzadeh, M., Gabillon, V., Lesner, B. and Geist, M. Approximate modified policy iteration and its application to the game of tetris. *Journal of Machine Learning Research*, 16(1):1629–1676, 2015.

[48] Sevinc, S., Sanli, U. and Goker, E. Algorithms for scheduling of chemotherapy plans. *Computers in Biology and Medicine*, 43(12):2103–2109, 2013.

[49] Singh, K., Philpott, A. and Wood, K. Dantzig-Wolfe decomposition for solving multistage stochastic capacity-planning problems. *Operations Research*, 57(5):1271–1286, 2009.

[50] Song, H. and Huang, H. A successive convex approximation method for multistage workforce capacity planning problem with turnover. *European Journal of Operational Research*, 188(1):29–48, 2008.

[51] Tsitsiklis, J. and Van Roy, B. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1-3):59–94, 1996.

[52] Van De Vonder, S., Demeulemeester, E. and Herroelen, W. A classification of predictive-reactive project scheduling procedures. *Journal of Scheduling*, 10(3):195–207, 2007.

[53] Vanhoucke, M. Scheduling an R&D Project with Quality-Dependent Time Slots. *International Conference on Computational Science and Its Applications: Part of the Lecture Notes in Computer Science*, 3982(1):621–630, 2006.

[54] Vogl, P., Braune, R. and Doerner, K. Scheduling recurring radiotherapy appointments in an ion beam facility. *Journal of Scheduling*, 22(1):137–154, 2019.

[55] Yang, H. and Cheng, Y. Finding the critical path in an activity network with time-switch constraints. *European Journal of Operational Research*, 120(3):603–613, 2000.

[56] Zhou, S. and Yue, Q. Appointment scheduling for multi-stage sequential service systems with stochastic service durations. *Computers and Operations Research*, 112(1):1–13, 2019.

[57] Öncan, T. A Survey of the Generalized Assignment Problem and Its Applications. *INFOR: Information Systems and Operational Research*, 45(3):123–141, 2007.

Appendices

A Glossary Dutch terms

Since most of the law-related terms in this thesis have been translated from Dutch, this appendix gives an overview of these terms as well as the original Dutch terms.

English term (translated)	**Dutch term** (original)
Appealing	Beroepschrift indienen
Administrative law	Bestuursrecht
Commercial law	Handelsrecht
Court	Zaak
Courtroom	Zittingszaal
Courts of justice	Gerechtshoven
Defendant	Gedaagde
Hearing	Zitting
Judge	Rechter
Judgement	Uitspraak
Jurisdiction	Rechtsgebied
Lawyer	Advocaat
Prosecutor	Eiser
Settlement	Schikking
Subpoena	Dagvaardiging
Summary proceedings	Kort geding (handelsrecht)/ voorlopige voorzieningen (bestuursrecht)
Team chairman	Teamvoorzitter
Training matters	Opleidingszaken
Verdict	Vonnis

B Overview Integer Linear Program commercial law

This Appendix covers the ILP-model made for the commercial law planning problem. In this appendix we start with an overview of the binary variables in Table 8 and the integer variables in Table 9. We give an overview of the binary parameters in Table 10 and the integer parameters in Table 11. Thereafter, we give the whole ILP: the objective and its constraints.

Table 8: Binary variables ILP

x_{ajt}	1 if appointment a is assigned to judge(s) j and starts at (the beginning of) timeslot t
y_{alt}	1 if appointment a is assigned to legal assistant l and starts at (the beginning of) timeslot t
q_{ajlt}	1 if appointment a is assigned to judge(s) j, legal assistant l and starts at (the beginning of) timeslot t
n_a	1 if appointment a is not scheduled
b_a	1 if one of the appointments $a \in \mathcal{P}, a + \lvert\mathcal{P}\rvert \in \mathcal{H}$ or $a + 2 \cdot \lvert\mathcal{P}\rvert$ cannot be scheduled for either a legal assistant and one judge (for a one-fold case) or three judges (for a multiple case)

Table 9: Integer variables ILP

χ	The maximum difference in available and scheduled hours for the judges minus the minimum difference in available and scheduled hours for the judges
υ	The maximum difference in available and scheduled hours for the legal assistants minus the minimum difference in available and scheduled hours for the legal assistants

Table 10: Binary parameters ILP

τ_a	1 if appointment a is a multiple case
BJ_{jt}	1 if judge j is available in timeslot t
BL_{lt}	1 if legal assistant l is available in timeslot t
BP_t	1 if the prosecutor is available in timeslot t
$C_{a\widetilde{a}}$	1 if appointment a should take place before $\widetilde{a}$
I_{js}	1 if judge j can handle cases of specialism s
J_{lsd}	1 if legal assistant l can handle cases of specialism s and difficulty d
θ_{ads}	1 if appointment a has difficulty d and belongs to specialism s

Table 11: Integer parameters ILP

M_{aj}	The duration of appointment a (in number of timeslots) for judge j
N_{al}	The duration of appointment a (in number of timeslots) for legal assistant l
T	The number of timeslots in each day
γ_a	First timeslot of the day after arrival of the case belonging to appointment a
ϵ_1	Upper limit (norm in days) of having a hearing, after announcement of the case
ϵ_2	Upper limit (norm in days) of judgement, after the hearing

min $\beta_1 \cdot \sum_a n_a + \beta_2 \cdot \chi + \beta_3 \cdot \upsilon + \beta_4 \cdot \sum_a \psi_a + \beta_5 \cdot \sum_a \omega_a$
s.t.

$$\sum_a x_{ajt} \leq 1, \quad \forall \quad j,t \tag{B.1}$$

$$\sum_a y_{alt} \leq 1, \quad \forall \quad l,t \tag{B.2}$$

$$\sum_{\widetilde{a},\widetilde{t}|t<\widetilde{t}\leq t+M_{aj}-1} x_{\widetilde{a}j\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a,j,t \tag{B.3}$$

$$\sum_{\widetilde{a},\widetilde{t}|t<\widetilde{t}\leq t+N_{al}-1} y_{\widetilde{a}l\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a,l,t \tag{B.4}$$

$$\sum_{\widetilde{l}\widetilde{t}} y_{a\widetilde{l}\widetilde{t}} \leq 2 - y_{alt} \quad \forall \quad a,l,t \tag{B.5}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a,j,t \tag{B.6}$$

$$\tau_a \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a,j,t \tag{B.7}$$

$$(1-\tau_a) \cdot \sum_{\widetilde{j}\widetilde{t}} x_{a\widetilde{j}\widetilde{t}} \leq (1-\tau_a) \cdot (2 - x_{ajt}) \quad \forall \quad a,j,t \tag{B.8}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \leq \tau_a \cdot (4 - x_{ajt}) \quad \forall \quad a \in \mathcal{H},j,t \tag{B.9}$$

$$\tau_a \cdot \sum_{\widetilde{j}} x_{a\widetilde{j}t} \geq 3 \cdot \tau_a \cdot x_{ajt} \quad \forall \quad a \in \mathcal{H},j,t \tag{B.10}$$

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{P},j,t \tag{B.11}$$

$$\sum_{\widetilde{t}} x_{(a+|\mathcal{P}|)j\widetilde{t}} \geq x_{ajt} \quad \forall \quad a \in \mathcal{H},j,t \tag{B.12}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{P}, l, t \tag{B.13}$$

$$\sum_{\widetilde{t}} y_{(a+|\mathcal{P}|)l\widetilde{t}} \geq y_{alt} \quad \forall \quad a \in \mathcal{H}, l, t \tag{B.14}$$

$$\frac{x_{ajt} \cdot t - \{x_{ajt} \cdot t\} \mod T}{T} = \frac{x_{ajt} \cdot t - x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T}{T} \quad \forall \quad a \in \mathcal{H}, j, t \tag{B.15}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}\widetilde{j}\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, \widetilde{a}, j, t \tag{B.16}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot x_{\widetilde{a}\widetilde{j}\widetilde{t}} \leq 1 - x_{ajt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{W}, \widetilde{a}, j, t \tag{B.17}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}\widetilde{l}\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{P}, \widetilde{a} \in \mathcal{H}, \widetilde{a}, l, t \tag{B.18}$$

$$\sum_{\widetilde{t}<t} C_{a\widetilde{a}} \cdot y_{\widetilde{a}\widetilde{l}\widetilde{t}} \leq 1 - y_{alt} \quad \forall \quad a \in \mathcal{H}, \widetilde{a} \in \mathcal{W}, \widetilde{a}, l, t \tag{B.19}$$

$$q_{ajlt} \leq x_{ajt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{B.20}$$

$$q_{ajlt} \leq y_{alt} \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{B.21}$$

$$q_{ajlt} \geq x_{ajt} + y_{alt} - 1 \quad \forall \quad a \in \mathcal{H}, j, l, t \tag{B.22}$$

$$q_{ajlt} \leq BL_{l\widetilde{t}} \cdot BJ_{j\widetilde{t}} \cdot BP_{a\widetilde{t}} \quad \forall \quad a \in \mathcal{H}, l, j, t \leq \widetilde{t} \leq t + M_{aj} - 1 \tag{B.23}$$

$$x_{ajt} \leq BJ_{j\widetilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, j, t \leq \widetilde{t} \leq t + M_{aj} - 1 \tag{B.24}$$

$$y_{alt} \leq BL_{l\widetilde{t}} \quad \forall \quad a \in \mathcal{P}, a \in \mathcal{W}, l, t \leq \widetilde{t} \leq t + N_{al} - 1 \tag{B.25}$$

$$\sum_{j} x_{ajt} = \sum_{l} y_{alt} \quad \forall \quad a \in \mathcal{H}, t \tag{B.26}$$

$$I_{js} \cdot \sum_{d} \theta_{ads} \leq 2 - x_{ajt} \quad \forall \quad a, j, t, s \tag{B.27}$$

$$J_{lsd} \cdot \theta_{ads} \leq 2 - y_{alt} \quad \forall \quad a, l, t, s, d \tag{B.28}$$

$$\chi = r - s \tag{B.29}$$

$$r \geq \sum_{t} BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj} \quad \forall \quad j \tag{B.30}$$

$$s \leq \sum_{t} BJ_{jt} - \sum_{at} x_{ajt} \cdot M_{aj} \quad \forall \quad j \tag{B.31}$$

$$v = u - w \tag{B.32}$$

$$u \geq \sum_{t} BL_{lt} - \sum_{at} y_{alt} \cdot N_{al} \quad \forall \quad l \tag{B.33}$$

$$w \leq \sum_{t} BL_{lt} - \sum_{at} y_{alt} \cdot N_{al} \quad \forall \quad l \tag{B.34}$$

$$\psi_a = [\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T - \gamma_a - \epsilon_1 \cdot T]^+ \quad \forall \quad a \in \mathcal{H}, \epsilon_1 \in \mathbb{N} \tag{B.35}$$

$$\sum_{jt} x_{ajt} \cdot \{t + M_{aj}\} \geq \sum_{lt} y_{alt} \cdot \{t + N_{al}\} \quad \forall \quad a \in \mathcal{W}, t \tag{B.36}$$

$$\begin{aligned} \omega_a = [\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} - (\sum_{j\tilde{t}} x_{(a+|\mathcal{P}|)j\tilde{t}} \cdot \{\tilde{t} + M_{(a+|\mathcal{P}|)j} - 1\} \mod T) + T - \\ (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - (\sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} \mod T) + T) \\ -\epsilon_2 \cdot T]^+ \quad \forall \quad a \in \mathcal{H}, \epsilon_2 \in \mathbb{N} \end{aligned} \tag{B.37}$$

$$b_a \leq 3 \cdot \sum_{lt} y_{alt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{B.38}$$

$$b_a \leq \sum_{jt} x_{ajt} \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{B.39}$$

$$b_a \geq \sum_{jt} x_{ajt} - 3 \cdot (1 - \sum_{lt} y_{alt}) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{B.40}$$

$$b_a = 3 \cdot \tau_a \cdot (1 - n_a) + (1 - \tau_a) \cdot (1 - n_a) \quad \forall \quad a \in \mathcal{A}, a + |\mathcal{P}| \in \mathcal{A}, a + 2 \cdot |\mathcal{P}| \in \mathcal{A} \tag{B.41}$$

C AIMMS-model commercial law

In this appendix we elaborate on how we have implemented the constraints in AIMMS as described in Section 4.1 containing a modulo or maximization function. These constraints have been adjusted in order to keep the model linear. This is about the constraints in equation (4.15), (4.35) and (4.37) from Section 4.1.

We have reformulated the modulo function as follows: Suppose we have $\mod(x, y)$ with y integer. Then, $\mod(x, y)$ can be linearized by adding the variable $r \geq 0$ (where $r = \mod(x, y)$), z being a new integer variable and introducing the following constraints:

$$x = z \cdot y + r$$

$$r \leq y - 1$$

We have reformulated the maximization function as follows: Suppose we have $z = [h]^+$ and one wants to add z to the minimization objective, then make the free variable y, where y is defined similar to h, and the nonnegative variable q. Then, q has to be added to the objective and the following constraint has to be added to the model: $q \geq y$.

In order to implement equation (4.15) linearly, we have introduced the integer variables $HM1_{ajt}$ and $HM2_{ajt}$ and the nonnegative variables r_{ajt} and $s1_{ajt}$. Furthermore, we have introduced the constraints (C.1) until (C.5) in order to model the equation (4.15) in AIMMS.

$$x_{ajt} \cdot t = HM1_{ajt} \cdot T + r_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{C.1}$$

$$r_{ajt} \leq T - 1 \quad \forall \quad a \in \mathcal{H}, j, t \tag{C.2}$$

$$\frac{x_{ajt} \cdot t - r_{ajt}}{T} = \frac{x_{ajt} \cdot t - s1_{ajt}}{T} \quad \forall \quad a \in \mathcal{H}, j, t \tag{C.3}$$

$$x_{ajt} \cdot (t + M_{aj} - 1) = HM2_{ajt} \cdot T + s1_{ajt} \quad \forall \quad a \in \mathcal{H}, j, t \tag{C.4}$$

$$s1_{ajt} \leq T - 1 \quad \forall \quad a \in \mathcal{H}, j, t \tag{C.5}$$

In order to implement equation (4.35) linearly -so without the maximization and the modulo function- we have introduced the free variable $Standards1_a$, the integer variable $HMS1_a$ and the nonnegative variables $w1_a$ and $vnew_a$. Furthermore, we have introduced constraints (C.6), (C.7), (C.7), (C.9) and have added $\sum_a vnew_a$ in the objective function in AIMMS in order to add $\sum_a \psi_a$ as ψ_a is defined in equation (4.35).

$$Standards1_a = \sum_{jt} x_{ajt} \cdot \{t + M_{aj} - 1\} - w1_a + T - \gamma_a - \epsilon_1 \cdot T \quad \forall \quad a \in \mathcal{H}, \epsilon_1 \in \mathbb{N} \tag{C.6}$$

$$vnew_a \geq Standards1_a \quad \forall \quad a \in \mathcal{H} \tag{C.7}$$

$$\sum_{jt} x_{ajt} \cdot (t + M_{aj} - 1) = HMS1_a \cdot T + w1_a \quad \forall \quad a \in \mathcal{H} \tag{C.8}$$

$$w1_a \leq T-1 \quad \forall \quad a \in \mathcal{H} \tag{C.9}$$

In order to implement equation (4.37) linearly -so without the maximization and the modulo function- we have introduced the free variable $Standards3_a$, the integer variable $HMS3Deel1_{a,\tilde{t}}$ and $HMS3Deel2_{at}$ and the nonnegative variables $vnew2_a$, $w3Deel1_{a,\tilde{t}}$ and $w3Deel2_{at}$. Furthermore, we have introduced the constraints (C.10), (C.11), (C.12), (C.13), (C.14) and (C.15) and have added $\sum_a vnew2_a$ in the objective function in AIMMS in order to add $\sum_a \omega_a$ as ω_a is defined in equation (4.37).

$$\begin{aligned} Standards3_a = & \sum_{j\tilde{t}} [x_{(a+|\mathcal{H}|)j\tilde{t}} \cdot (\tilde{t} + M_{(a+|\mathcal{H}|)j} - 1)] - \sum_{\tilde{t}} [w3Deel1_{a,\tilde{t}}] + T \\ & - \sum_{jt} [x_{ajt} \cdot (t + M_{aj} - 1)] - \sum_{t} [w3Deel2_{at}] + T - \epsilon_2 \cdot T \quad \forall \quad a \in \mathcal{H} \end{aligned} \tag{C.10}$$

$$vnew2_a \geq Standards3_a \quad \forall \quad a \in \mathcal{H} \tag{C.11}$$

$$\sum_j x_{(a+|\mathcal{H}|)j\tilde{t}} \cdot (\tilde{t} + M_{(a+|\mathcal{H}|)j} - 1) = HMS3Deel1_{a\tilde{t}} \cdot T + w3Deel1_{a\tilde{t}} \quad \forall \quad a \in \mathcal{H}, \tilde{t} \tag{C.12}$$

$$w3Deel1_t \leq T-1 \quad \forall \quad a, t \tag{C.13}$$

$$\sum_j x_{ajt} \cdot (t + M_{aj} - 1) = HMS3Deel2_{at} \cdot T + w3Deel2_{at} \quad \forall \quad a \in \mathcal{H}, t \tag{C.14}$$

$$w3Deel2_{at} \leq T-1 \quad \forall \quad a, t \tag{C.15}$$

D Results Integer Linear Program commercial law

In this appendix we give the results of the different tests, where we give the results of the tests on the validation data. In the different tests we have not changed the values of β_4 and β_5: $\beta_4 = \beta_5 = 1$ in every situation.

D.1 Scenario 1

Table 12: Results after 30 minutes of running (with 8 judges and 12 legal assistants)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	χ	v
$\frac{1}{24}$	$\frac{1}{125}$	$\frac{1}{125}$	45.83 %	21	4	2
1	0	0	0.00 % (within 317 sec)	16	22	35
0	$\frac{1}{125}$	$\frac{1}{125}$	n/a	24	0	0
$\frac{1}{24}$	$\frac{1}{350}$	$\frac{1}{350}$	49.25 %	22	3	21

Table 13: Results after 30 minutes of running (with 10 judges and 12 legal assistants)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	χ	v
$\frac{1}{24}$	$\frac{1}{125}$	$\frac{1}{125}$	47.20 %	21	4	5
1	0	0	14.29 %	14	21	9
0	$\frac{1}{125}$	$\frac{1}{125}$	n/a	24	0	0
$\frac{1}{24}$	$\frac{1}{350}$	$\frac{1}{350}$	41.60 %	20	3	5

Table 14: Results after 30 minutes of running (with 8 judges and 12 legal assistants where the judges and legal assistant can handle (all difficulties of) all specialties)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	χ	v
$\frac{1}{24}$	$\frac{1}{125}$	$\frac{1}{125}$	50.00 %	24	0	0
1	0	0	0.00 % (within 145.75 sec.)	16	1	18
0	$\frac{1}{125}$	$\frac{1}{125}$	n/a	24	0	0
$\frac{1}{24}$	$\frac{1}{350}$	$\frac{1}{350}$	49.25 %	22	3	21

D.2 Scenario 2

Note: In the first situation in Table 15, the model is still in the presolving phase after ten minutes. Therefore, the model was not able to provide a solution after ten minutes of running.

Table 15: Results after 10 minutes of running (with 8 judges and 12 legal assistants)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	$\sum_j \chi_j$	$\sum_l v_l$
$\frac{1}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	10.67 %	20	64	111
1	0	0	6.25 %	16	159	282
0	$\frac{1}{280}$	$\frac{1}{420}$	31.78 %	24	71	74
$\frac{10}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	9.08 %	17	88	159
$\frac{1}{24}$	$\frac{10}{280}$	$\frac{10}{420}$	24.67 %	23	70	94

Table 16: Results after 10 minutes of running (with 10 judges and 12 legal assistants)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	$\sum_j \chi_j$	$\sum_l v_l$
$\frac{1}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	n/a	n/a	n/a	n/a
1	0	0	25.00 %	16	243	270
0	$\frac{1}{280}$	$\frac{1}{420}$	83.99 %	24	169	259
$\frac{10}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	79.52 %	24	196	259
$\frac{1}{24}$	$\frac{10}{280}$	$\frac{10}{420}$	80.38 %	24	196	259

Table 17: Results after 10 minutes of running (with 8 judges and 12 legal assistants where the judges and legal assistants can handle (all difficulties of) all specialties)

β_1	β_2	β_3	**Optimality gap**	$\sum_a n_a$	$\sum_j \chi_j$	$\sum_l v_l$
$\frac{1}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	10.13 %	20	62	109
1	0	0	27.55 %	19	212	331
0	$\frac{1}{280}$	$\frac{1}{420}$	39.20 %	23	69	99
$\frac{10}{24}$	$\frac{1}{280}$	$\frac{1}{420}$	1.28 %	16	79	158
$\frac{1}{24}$	$\frac{10}{280}$	$\frac{10}{420}$	30.54 %	23	68	94

E Additions for Integer Linear Program administrative law

In this appendix we go further into the adjustments and additions that have to be made to the ILP for commercial law in order to be valid for the planning problem of administrative law. We explain the parameters and constraints that have to be added in Section E.1. An overview of the additions to model for commercial law, including the new parameter, variables and constraints is given in Section E.2. Unfortunately, due to time restrictions, these additions have not been implemented and tested.

E.1 Explanation additions ILP administrative law

For administrative law, one hearing contains different cases of the same specialism, with the same judge(s) and legal assistant. Hence, we must adjust the model of commercial law in order to fit to the procedure of administrative law. In the model for administrative law, we introduce the new index h for hearings and the new decision variable z_{aht}, which we define as follows:

$$z_{aht} = \begin{cases} 1, & \text{if appointment } a \in \mathcal{H} \text{ is assigned to hearing } h \text{ starting at timeslot } t \\ 0, & \text{otherwise} \end{cases}$$

Each appointment that entails a hearing has to be scheduled within a hearing, which we model in equation (E.1). Furthermore, z_{aht} should start at the same timeslot already defined in x_{alt} for $a \in \mathcal{H}$, where we use x_{alt} without loss of generality. We model this in equation (E.2).

$$\sum_{ht} z_{aht} = 1 \quad \forall \quad a \in \mathcal{H} \tag{E.1}$$

$$z_{aht} \leq x_{alt} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.2}$$

Furthermore, each hearing must be handled by the same judge(s) and legal assistant and can only contain cases of the same specialism. To model the constraints that each hearing must contain cases of the same specialism, we introduce the binary auxiliary variable $\hat{z}_{hs}$ that is 1 if hearing h contains cases of specialism s. The constraints are modelled in equations (E.3) and (E.4). The constraints that each hearing should be handled by the same judge(s) and legal assistant are then as follows:

$$\sum_{jt} x_{ajt} \cdot z_{aht} \leq 3 \cdot \tau_a + (1 - \tau_a) \quad \forall \quad a \in \mathcal{H}, h$$

$$\sum_{lt} y_{alt} \cdot z_{aht} \leq 1 \quad \forall \quad a \in \mathcal{H}, h$$

Since these constraints contain multiplications of two binary variables, they are not linear. We employ the earlier used modeling trick to make the multiplication of two binary variable linear by introducing the binary auxiliary variables r_{ahjt} and s_{ahlt}. The obtained linear constraints are given in equation (E.5) up to and including (E.12).

$$\hat{z}_{hs} \leq \sum_{d} \theta_{ads} \cdot \sum_{t} z_{aht} \quad \forall \quad a \in \mathcal{H}, h, s \tag{E.3}$$

$$\sum_{s} \hat{z}_{hs} = 1 \quad \forall \quad h \tag{E.4}$$

$$r_{ahjt} \leq x_{ajt} \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.5}$$

$$r_{ahjt} \leq z_{aht} \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.6}$$

$$r_{ahjt} \geq x_{ajt} + z_{aht} - 1 \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.7}$$

$$\sum_{jt} r_{ahjt} \leq 3 \cdot \tau_a + (1 - \tau_a) \quad \forall \quad a \in \mathcal{H}, h \tag{E.8}$$

$$s_{ahlt} \leq y_{alt} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.9}$$

$$s_{ahlt} \leq z_{aht} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.10}$$

$$s_{ahlt} \geq y_{alt} + z_{aht} - 1 \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.11}$$

$$\sum_{lt} s_{ahlt} \leq 1 \quad \forall \quad a \in \mathcal{H}, h \tag{E.12}$$

A maximum of four hearings can be scheduled a day per pair of judge(s) and legal assistant. In order to model this constraint, we let $\mathcal{T}_\mu$ be the set of timeslots t within a day μ. We thus want the following:

$$\sum_{h} x_{ajt} \cdot z_{aht} \leq 4 \quad \forall \quad a \in \mathcal{H}, j, t \in \mathcal{T}_\mu$$

$$\sum_{h} y_{alt} \cdot z_{aht} \leq 4 \quad \forall \quad a \in \mathcal{H}, l, t \in \mathcal{T}_\mu$$

In order to make these constraints linear, we make use of the earlier introduced auxiliary binary variables r_{ahjt} and s_{ahlt} and model the constraints in equations (E.13) and (E.14).

$$\sum_{h} r_{ahjt} \leq 4 \quad \forall \quad a \in \mathcal{H}, j, t \in \mathcal{T}_\mu \tag{E.13}$$

$$\sum_{h} s_{ahlt} \leq 4 \quad \forall \quad a \in \mathcal{H}, l, t \in \mathcal{T}_\mu \tag{E.14}$$

E.2 Overview additions ILP administrative law

We give an overview of the added variables in Table 18 and the added parameter in Table 19. Furthermore, we have added the set $\mathcal{T}_\mu$ that gives all timeslots t within day μ. Afterwards, we give the added constraints in order to obtain the model for administrative law. The objective has not been changed.

Table 18: Added binary variables for ILP commercial law

z_{aht}	1 if appointment $a \in \mathcal{H}$ is assigned to hearing h and starts at (the beginning of) timeslot t
r_{ahjt}	1 if appointment a is assigned to hearing h, judge j and starts at (the beginning of) timeslot t
s_{ahlt}	1 if appointment a is assigned to hearing h, legal assistant l and starts at (the beginning of) timeslot t

Table 19: Added binary parameter for ILP administrative law

$\hat{z}_{hs}$	1 if hearing h contains cases of specialism s

The constraints that have to be added to the model for commercial law, in order to obtain the model for administrative law are stated below.

$$\sum_{ht} z_{aht} = 1 \quad \forall \quad a \in \mathcal{H} \tag{E.15}$$

$$z_{aht} \leq x_{alt} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.16}$$

$$\hat{z}_{hs} \leq \sum_{d} \theta_{ads} \cdot \sum_{t} z_{aht} \quad \forall \quad a \in \mathcal{H}, h, s \tag{E.17}$$

$$\sum_{s} \hat{z}_{hs} = 1 \quad \forall \quad h \tag{E.18}$$

$$r_{ahjt} \leq x_{ajt} \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.19}$$

$$r_{ahjt} \leq z_{aht} \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.20}$$

$$r_{ahjt} \geq x_{ajt} + z_{aht} - 1 \quad \forall \quad a \in \mathcal{H}, h, j, t \tag{E.21}$$

$$\sum_{jt} r_{ahjt} \leq 3 \cdot \tau_a + (1 - \tau_a) \quad \forall \quad a \in \mathcal{H}, h \tag{E.22}$$

$$s_{ahlt} \leq y_{alt} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.23}$$

$$s_{ahlt} \leq z_{aht} \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.24}$$

$$s_{ahlt} \geq y_{alt} + z_{aht} - 1 \quad \forall \quad a \in \mathcal{H}, h, l, t \tag{E.25}$$

$$\sum_{lt} s_{ahlt} \leq 1 \quad \forall \quad a \in \mathcal{H}, h \tag{E.26}$$

$$\sum_{h} r_{ahjt} \leq 4 \quad \forall \quad a \in \mathcal{H}, j, t \in \mathcal{T}_{\mu} \tag{E.27}$$

$$\sum_{h} s_{ahlt} \leq 4 \quad \forall \quad a \in \mathcal{H}, l, t \in \mathcal{T}_{\mu} \tag{E.28}$$

F Calculation basis functions in ILP-form

In this appendix we elaborate on how we have implemented the basis functions in Python, making use of the MIP-package. We make use of these two modelling tricks:

1. You can model $(y = x_1)$ OR $(y = x_2)$ OR .. OR $(y = x_n)$ linearly by using the following constraints:
$$y \geq x_i \quad \forall \quad i = \{1, .., n\}$$
$$y \leq \sum_i x_i$$

2. You can model $(y = x_1)$ AND $(y = x_2)$ AND .. AND $(y = x_n)$ linearly by using the following constraints:
$$y \leq x_i \quad \forall \quad i = \{1, .., n\}$$
$$y \geq [\sum_i x_i] - n + 1$$

For basis function 1 up to and including 8 we show how we can calculate the value of the basis function per judge, where the value of these basis functions per legal assistants can be calculated similarly: replacing only x_{jt} by y_{lt} and ϕ_{kj} by ϕ_{kl} for $k = 1, .., 8$. We show the added variables and constraints per basis function:

1. In the first basis function, we calculate the amount of free time periods consisting of one loose block (e.g. 1 single free timeslot). We introduce the variable a_{jt} such that $\phi_{1j} = \sum_t a_{jt}$ where:
$$a_{jt} = \begin{cases} 1, & \text{if } x_{j(t-1)} = 0 \text{ and } x_{jt} = 1 \text{ and } x_{j(t+1)} = 0 \\ 0, & \text{otherwise.} \end{cases}$$
We thus have the following constraint:
"IF $(x_{jt} = 1)$ AND $(x_{j(t-1)} = 0)$ AND $(x_{j(t+1)} = 0)$ THEN $a_{jt} = 1$". We model this linearly by introducing constraints (F.1)-(F.5) and add $\sum_j \phi_{1j}$ to the objective (in our ILP in Python).
$$a_{jt} \leq x_{jt} \quad \forall \quad j, t \tag{F.1}$$
$$a_{jt} \leq 1 - x_{j(t-1)} \quad \forall \quad j, t \tag{F.2}$$
$$a_{jt} \leq 1 - x_{j(t+1)} \quad \forall \quad j, t \tag{F.3}$$
$$a_{jt} \geq x_{jt} - x_{j(t-1)} - x_{j(t+1)} \quad \forall \quad j, t \tag{F.4}$$
$$\phi_{1j} = \sum_t a_{jt} \quad \forall \quad j \tag{F.5}$$

2. In the second basis function, we calculate the amount of 2 consecutive free timeslots in the schedule. We therefore introduce the variable b_{jt} such that $\phi_{2j} = \sum_t b_{jt}$ where:
$$b_{jt} = \begin{cases} 1, & \text{if } x_{j(t-1)} = 0 \text{ and } x_{jt} = 1 \text{ and } x_{j(t+1)} = 1 \text{ and } x_{j(t+2)} = 0 \\ 0, & \text{otherwise.} \end{cases}$$
We thus have the following constraint:
"IF $(x_{jt} = 1)$ AND $(x_{j(t+1)} = 1)$ AND $(x_{j(t-1)} = 0)$ AND $(x_{j(t+2)} = 0)$ THEN $b_{jt} = 1$".

We model this linearly by introducing constraints (F.6)-(F.11) and add $\sum_j \phi_{2j}$ to the objective (in our ILP in Python).

$$b_{jt} \leq x_{jt} \quad \forall \quad j, t \tag{F.6}$$

$$b_{jt} \leq x_{j(t+1)} \quad \forall \quad j, t \tag{F.7}$$

$$b_{jt} \leq 1 - x_{j(t-1)} \quad \forall \quad j, t \tag{F.8}$$

$$b_{jt} \leq 1 - x_{j(t+2)} \quad \forall \quad j, t \tag{F.9}$$

$$b_{jt} \geq x_{jt} + x_{j(t+1)} - x_{j(t-1)} - x_{j(t+2)} - 1 \quad \forall \quad j, t \tag{F.10}$$

$$\phi_{2j} = \sum_t b_{jt} \quad \forall \quad j, t \tag{F.11}$$

3. In the third basis function, we calculate the amount of 3 consecutive free timeslots in the schedule. We introduce the variable c_{jt} such that $\phi_{3j} = \sum_t c_{jt}$ where:

$$c_{jt} = \begin{cases} 1, & \text{if } x_{j(t-1)} = 0 \text{ and } x_{jt} = 1 \text{ and } x_{j(t+1)} = 1 \text{ and } x_{j(t+2)} = 1 \text{ and } x_{j(t+3)} = 0 \\ 0, & \text{otherwise.} \end{cases}$$

We thus get the following constraint:
"IF ($x_{jt} = 1$) AND ($x_{j(t+1)} = 1$) AND ($x_{j(t+2)} = 1$) AND ($x_{j(t-1)} = 0$) AND ($x_{j(t+3)} = 0$) THEN $c_{jt} = 1$". We model this linearly by introducing constraints (F.12)-(F.18) and add $\sum_j \phi_{3j}$ to the objective (in our ILP in Python).

$$c_{jt} \leq x_{jt} \quad \forall \quad j, t \tag{F.12}$$

$$c_{jt} \leq x_{j(t+1)} \quad \forall \quad j, t \tag{F.13}$$

$$c_{jt} \leq x_{j(t+2)} \quad \forall \quad j, t \tag{F.14}$$

$$c_{jt} \leq 1 - x_{j(t-1)} \quad \forall \quad j, t \tag{F.15}$$

$$c_{jt} \leq 1 - x_{j(t+3)} \quad \forall \quad j, t \tag{F.16}$$

$$c_{jt} \geq x_{jt} + x_{j(t+1)} + x_{j(t+2)} - x_{j(t-1)} - x_{j(t-3)} - 2 \quad \forall \quad j, t \tag{F.17}$$

$$\phi_{3j} = \sum_t c_{jt} \quad \forall \quad j \tag{F.18}$$

4. In the fourth basis function, we calculate the number of consecutive nonscheduled time periods. This can also be seen as the number of times a free timeslot follows a scheduled timeslot, per judge (or legal assistant). We thus introduce the variable d_{jt} where such that $\phi_{4j} = \sum_t d_{jt}$ where:

$$d_{jt} = \begin{cases} 1, & \text{if } x_{jt} = 0 \text{ and } x_{j(t+1)} = 1 \\ 0, & \text{otherwise.} \end{cases}$$

We thus have the following constraint:
"IF ($x_{jt} = 0$) AND ($x_{j(t+1)} = 1$) THEN $d_{jt} = 1$." We model this linearly by introducing constraints (F.19)-(F.22) and add $\sum_j \phi_{4j}$ to the objective (in our ILP in Python).

$$d_{jt} \leq x_{j(t+1)} \quad \forall \quad j, t \tag{F.19}$$

$$d_{jt} \leq 1 - x_{jt} \quad \forall \quad j, t \tag{F.20}$$

$$d_{jt} \geq x_{j(t+1)} - x_{jt} \quad \forall \quad j, t \tag{F.21}$$

$$\phi_{4j} = \sum_t d_{jt} \quad \forall \quad j \tag{F.22}$$

5. In the fifth basis function, we calculate the first moment when 6 consecutive timeslots are available. In order to do this, we first introduce the binary variable f_{jt} that is 1 if there have been 6 consecutive free timeslots on timeslot t. In order to do this, we use the same manner as for basis function 1, 2 and 3 and obtain constraints (F.23)-(F.30).

$$f_{jt} \leq x_{jt} \quad \forall \quad j, t \tag{F.23}$$

$$f_{jt} \leq x_{j(t+1)} \quad \forall \quad j, t \tag{F.24}$$

$$f_{jt} \leq x_{j(t+2)} \quad \forall \quad j, t \tag{F.25}$$

$$f_{jt} \leq x_{j(t+3)} \quad \forall \quad j, t \tag{F.26}$$

$$f_{jt} \leq x_{j(t+4)} \quad \forall \quad j, t \tag{F.27}$$

$$f_{jt} \leq x_{j(t+5)} \quad \forall \quad j, t \tag{F.28}$$

$$f_{jt} \leq 1 - x_{j(t-1)} \quad \forall \quad j, t \tag{F.29}$$

$$f_{jt} \geq x_{jt} + x_{j(t+1)} + x_{j(t+2)} + x_{j(t+3)} + x_{j(t+4)} + x_{j(t+5)} - x_{j(t-1)} - 5 \quad \forall \quad j, t \tag{F.30}$$

We then introduce a new binary variable g_{jt} that is 1 if timeslot t is the first timeslot in which judge j has six consecutive available timeslots. We let g_{jt} be 1 if $f_{jt} = 1$ and $f_{jp} = 0$ where $p < t$. We then use the following constraint in order to calculate g_{jt}:

$$g_{jt} \geq f_{jt} \cdot (1 - f_{jp}) \quad \forall \quad p < t, j$$

Since this constraint contains a multiplication of two binary variables, we introduce the new binary variable q_{jt} in order to model $f_{jt} \cdot (1 - f_{jp})$ and implement (F.31)-(F.36) in our model in Python and we add ϕ_{5j} to the objective (of our ILP in Python).

$$q_{jt} \leq f_{jt} \quad \forall \quad j, t \tag{F.31}$$

$$q_{jt} \leq (1 - f_{jp}) \quad \forall \quad j, p < t \tag{F.32}$$

$$q_{jt} \geq f_{jt} + (1 - f_{jp}) - 1 \quad \forall \quad j, p < t \tag{F.33}$$

$$g_{jt} \geq q_{jt} \quad \forall \quad j, t \tag{F.34}$$

$$g_{jt} \leq 1 - f_{jp} \quad \forall \quad j, p < t \tag{F.35}$$

$$\phi_{5j} = \sum_{t} g_{jt} \cdot t \quad \forall \quad j \tag{F.36}$$

6. In the sixth basis function, we calculate the number of free timeslots that fall before free timeslots of length ≥ 4. In the same manner as for ϕ_5 we calculate the first timeslot in which four consecutive free timeslots take place, by introducing the binary variable h_{jt} that is 1 if t is a timeslot on which judge j has 4 consecutive free timeslots, k_{jt} is an auxiliary variable in order to linearize the multiplication of binary variables h_{jt} and $(1 - h_{jp})$ where $p < t$ and the binary variable m_{jt} that is 1 if t is the first timeslot on which judge j has 4 consecutive free timeslots. In order to model these variables, we introduce equations (F.37)- (F.46). We calculate the first timeslot on which judge j has 4 consecutive free timeslots by $\sum_t m_{jt} \cdot t$ and introduce equation (F.48) in order to calculate ϕ_{6j}.

$$h_{jt} \leq x_{jt} \quad \forall \quad j, t \tag{F.37}$$

$$h_{jt} \leq x_{j(t+1)} \quad \forall \quad j, t \tag{F.38}$$

$$h_{jt} \leq x_{j(t+2)} \quad \forall \quad j, t \tag{F.39}$$

$$h_{jt} \leq x_{j(t+3)} \quad \forall \quad j, t \tag{F.40}$$

$$h_{jt} \leq 1 - x_{j(t-1)} \quad \forall \quad j, t \tag{F.41}$$

$$h_{jt} \geq x_{jt} + x_{j(t+1)} + x_{j(t+2)} + x_{j(t+3)} - x_{j(t-1)} - 3 \quad \forall \quad j, t \tag{F.42}$$

$$k_{jt} \leq h_{jt} \quad \forall \quad j, t \tag{F.43}$$

$$k_{jt} \leq (1 - h_{jp}) \quad \forall \quad j, p < t \tag{F.44}$$

$$k_{jt} \geq h_{jt} + (1 - h_{jp}) - 1 \quad \forall \quad j, p < t \tag{F.45}$$

$$m_{jt} \geq k_{jt} \quad \forall \quad j, t \tag{F.46}$$

$$m_{jt} \leq 1 - h_{jp} \quad \forall \quad j, p < t \tag{F.47}$$

$$\phi_{6j} = \sum_{t=0}^{\sum_t m_{jt} \cdot t} a_{jt} + b_{jt} + c_{jt} \quad \forall \quad j \tag{F.48}$$

7. In the seventh basis function, we calculate the number of timeslots in which a judge and legal assistant of the same specialism are available. We introduce the variable n_{st} such that $\phi_{7s} = \sum_t n_{st}$ where:

$$n_{st} = \begin{cases} 1, & \text{if } x_{jt} = 1 \text{ for at least one } j \text{ of specialism } s \text{ and } y_{lt} = 1 \text{ for at least one } l \text{ of specialism } s \\ 0, & \text{otherwise.} \end{cases}$$

We thus want to have the following: for certain s and t: IF $(\sum_j x_{jt} \cdot I_{js} \geq 1)$ AND $(\sum_l y_{lt} \cdot \sum_d J_{lsd} \geq 1)$ for certain s and t THEN $n_{st} = 1$. This can be rewritten in a set of linear constraints using all the constraints below, where we introduce the binary variables $\hat{n}_{st}$ that is 1 if $\sum_j x_{jt} \cdot I_{js} \geq 1$ and $\tilde{n}_{st}$ that is 1 if $\sum_l y_{lt} \cdot \sum_d J_{lsd} \geq 1$.
In order to calculate $\hat{n}_{st}$ we rewrite the IF-THEN constraint into the following OR-constraint: $\sum_j x_{jt} \cdot I_{js} \geq 1$ OR $\hat{n}_{st} = 1$. This OR-constraint can then be rewritten into (F.49) and (F.50) where y is a binary variable and M_1 is sufficiently large.

$$\sum_j x_{jt} \cdot I_{js} \geq 1 - M_1 \cdot y \quad \forall \quad s, t \tag{F.49}$$

$$\hat{n}_{st} = 1 - y \quad \forall \quad s, t \tag{F.50}$$

In order to calculate $\tilde{n}_{st}$ we rewrite the IF-THEN constraint into the following OR-constraint: $\sum_l y_{lt} \cdot \sum_d J_{lsd} \geq 1$ OR $\tilde{n}_{st} = 1$. This OR-constraint can then be rewritten into (F.51) and (F.52) where 1 is a binary variable and M_12 is sufficiently large.

$$\sum_l y_{lt} \cdot \sum_d J_{lsd} \geq 1 - M_2 \cdot q \quad \forall \quad s, t \tag{F.51}$$

$$\tilde{n}_{st} = 1 - q \quad \forall \quad s, t \tag{F.52}$$

Then, in order to calculate n_{st} we end by adding (F.53) - (F.53) to our linear model.

$$n_{st} \leq \tilde{n}_{st} \quad \forall \quad s, t \tag{F.53}$$

$$n_{st} \leq \hat{n}_{st} \quad \forall \quad s, t \tag{F.54}$$

$$n_{st} \geq \hat{n}_{st} + \tilde{n}_{st} - 1 \quad \forall \quad s, t \tag{F.55}$$

8. In the eighth basis function, we calculate the first timeslot on which a judge and legal assistant of the same specialism are available for two consecutive timeslots. We should thus find the first timeslot t (per specialism s) for which $n_{st} = 1$ for two consecutive timeslots. In order to do this, we first introduce the binary variable z_{st} that is 1 if there have been 2 consecutive timeslots where $n_{st} = 1$.

$$z_{st} \leq n_{st} \quad \forall \quad s, t \tag{F.56}$$

$$z_{st} \leq n_{s(t+1)} \quad \forall \quad s, t \tag{F.57}$$

$$z_{st} \leq 1 - n_{s(t-1)} \quad \forall \quad s, t \tag{F.58}$$

$$z_{st} \geq n_{st} + n_{s(t+1)} - n_{s(t-1)} - 1 \quad \forall \quad s, t \tag{F.59}$$

We introduce a new binary variable w_{st} that is 1 if timeslot t is the first timeslot in which $z_{st} = 1$. We thus get:
IF ($z_{st} = 1$) AND ($z_{sp} = 0$) for $p < t$, THEN $w_{st} = 1$. This can be modeled using the following constraint:

$$w_{st} \geq z_{st} \cdot (1 - z_{sp}) \quad \forall \quad s, p < t$$

Since this constraint contains a multiplication of two binary variables, we introduce the new binary variable u_{st} in order to model $z_{st} \cdot (1 - z_{sp})$ and implement (F.60)-(F.65) in our model in Python and we add ϕ_{9s} to the objective (of our ILP in Python).

$$u_{st} \leq z_{st} \quad \forall \quad s, t \tag{F.60}$$

$$u_{st} \leq (1 - z_{sp}) \quad \forall \quad s, p < t \tag{F.61}$$

$$u_{st} \geq z_{st} + (1 - z_{sp}) - 1 \quad \forall \quad s, p < t \tag{F.62}$$

$$w_{st} \geq u_{st} \quad \forall \quad s, t \tag{F.63}$$

$$u_{st} \leq 1 - z_{sp} \quad \forall \quad s, p < t \tag{F.64}$$

$$\phi_{8s} = \sum_{t} w_{st} \cdot t \quad \forall \quad s \tag{F.65}$$

9. The ninth basis function is the constant offset feature. Hence, we have $\phi_9 = 1$ for every state.

In the tenth basis function, we calculate the number of consecutive free timeslots of length ≥ 5. We therefore introduce a function w_{kj} that calculates the amount of free time periods consisting of k blocks, similar to as what is done in order to calculate ϕ_{1j}, ϕ_{2j} and ϕ_{3j}. We then calculate ϕ_{10j} by using (F.66) where M is a large number. Since the number of constraints increases enormously by increasing M, we have chosen not to implement this basis function yet. If time allows, we can experiment with different values for M.

$$\phi_{10j} = \sum_{k=5}^{M} v_{kj} \quad \forall \quad j \tag{F.66}$$

G Output MDP toy-sized problem (n=50)

In this appendix we give the output of the first 10 iterations of the MDP using the following input:

- On average, one case arrives (following the Poisson process).
- We make use of an optimality gap of 0.05 and a maximum run time of the ILP of three minutes.
- We make use of 2 judges and 2 legal assistants.
- We make use of 1 specialism and 1 difficulty. Obviously, both judges and legal assistants can handle the specialism and difficulty.
- We start with a horizon of one week, that can move up two days (which makes the total amount of timeslots 49). The used availability is as follows:

```
Availability_judges =
    [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
Availability_legalassistants =
    [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

- We make use of 50 iterations.

The output is then as follows:
Note: The appointment numbers corresponding to one cases are here modeled as consecutive numbers (so e.g. appointment numbers $0, 1$ and 2 belong to the first arriving case, whereas in our model the appointment numbers $0, |\mathcal{P}|, 2 \cdot |\mathcal{P}|$ belong to the first arriving case).

```
Iteration n: 1
Iteration n: 2
Iteration t: 1
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 8]
Appointment durations for the legal assistants: [7, 2, 14]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 8]
Appointment durations for the legal assistants: [7, 2, 14]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 8]
Appointment durations for the legal assistants: [7, 2, 14]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 7
Appointment 1 judge 1 timeslot 16
Appointment 2 judge 1 timeslot 18
Appointment 0 legal assistant 1 timeslot 6
Appointment 1 legal assistant 1 timeslot 16
Appointment 2 legal assistant 1 timeslot 18
Objective value = 989.6142857142858
Iteration n: 3
Iteration t: 1
Appointment numbers: {0, 1, 2, 3, 4, 5}
Appointment durations for the judges: [6, 2, 8, 6, 2, 7]
Appointment durations for the legal assistants: [10, 2, 13, 7, 2, 11]
OptimizationStatus.OPTIMAL
Appointment 3 judge 0 timeslot 2
Appointment 4 judge 0 timeslot 12
Appointment 5 judge 0 timeslot 21
Appointment 3 legal assistant 0 timeslot 1
Appointment 4 legal assistant 0 timeslot 12
Appointment 5 legal assistant 0 timeslot 17
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [7, 2, 11]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 0
Appointment 1 judge 1 timeslot 18
Appointment 2 judge 1 timeslot 23
Appointment 0 legal assistant 1 timeslot 11
Appointment 1 legal assistant 1 timeslot 18
Appointment 2 legal assistant 1 timeslot 23
Objective value = 961.5295455486684
Iteration n: 4
Iteration t: 1
Appointment numbers: {0, 1, 2, 3, 4, 5}
Appointment durations for the judges: [7, 2, 6, 7, 2, 8]
Appointment durations for the legal assistants: [9, 2, 9, 8, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Appointment 3 is not scheduled
Appointment 4 is not scheduled
Appointment 5 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2, 3, 4, 5}
Appointment durations for the judges: [7, 2, 6, 7, 2, 8]
Appointment durations for the legal assistants: [9, 2, 9, 8, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 11
Appointment 1 judge 1 timeslot 18
Appointment 2 judge 1 timeslot 23
Appointment 0 legal assistant 1 timeslot 9
Appointment 1 legal assistant 1 timeslot 18
Appointment 2 legal assistant 1 timeslot 23
Appointment 3 is not scheduled
Appointment 4 is not scheduled
Appointment 5 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [7, 2, 8]
Appointment durations for the legal assistants: [8, 2, 10]
OptimizationStatus.INFEASIBLE
NO FEASIBLE SOLUTION IN ITERATION 4!!!
Iteration n: 4
Iteration t: 1
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [9, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [9, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 0
Appointment 1 judge 1 timeslot 18
Appointment 2 judge 1 timeslot 23
Appointment 0 legal assistant 1 timeslot 9
Appointment 1 legal assistant 1 timeslot 18
Appointment 2 legal assistant 1 timeslot 23
Objective value = 987.3204671436191
Iteration n: 5
Iteration n: 6
Iteration t: 1
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [7, 2, 8]
Appointment durations for the legal assistants: [7, 2, 15]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [7, 2, 8]
Appointment durations for the legal assistants: [7, 2, 15]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [7, 2, 8]
Appointment durations for the legal assistants: [7, 2, 15]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 6
Appointment 1 judge 1 timeslot 16
Appointment 2 judge 1 timeslot 18
Appointment 0 legal assistant 1 timeslot 6
Appointment 1 legal assistant 1 timeslot 16
Appointment 2 legal assistant 1 timeslot 18
Objective value = 942.7612030874521
Iteration n: 7
Iteration t: 1
Appointment numbers: {0, 1, 2, 3, 4, 5}
Appointment durations for the judges: [6, 2, 7, 7, 2, 7]
Appointment durations for the legal assistants: [7, 2, 9, 9, 2, 11]
OptimizationStatus.OPTIMAL
Appointment 0 judge 0 timeslot 2
Appointment 1 judge 0 timeslot 17
Appointment 2 judge 0 timeslot 19
Appointment 0 legal assistant 0 timeslot 1
Appointment 1 legal assistant 0 timeslot 17
Appointment 2 legal assistant 0 timeslot 19
Appointment 3 is not scheduled
Appointment 4 is not scheduled
Appointment 5 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [7, 2, 7]
Appointment durations for the legal assistants: [9, 2, 11]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 11
Appointment 1 judge 1 timeslot 18
Appointment 2 judge 1 timeslot 23
Appointment 0 legal assistant 1 timeslot 9
Appointment 1 legal assistant 1 timeslot 18
Appointment 2 legal assistant 1 timeslot 23
Objective value = 928.8911046424823
Iteration n: 8
Iteration t: 1
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 6]
Appointment durations for the legal assistants: [8, 2, 13]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 6]
Appointment durations for the legal assistants: [8, 2, 13]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 6]
Appointment durations for the legal assistants: [8, 2, 13]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 7
Appointment 1 judge 1 timeslot 16
Appointment 2 judge 1 timeslot 18
Appointment 0 legal assistant 1 timeslot 5
Appointment 1 legal assistant 1 timeslot 16
Appointment 2 legal assistant 1 timeslot 18
Objective value = 939.0382454401264
Iteration n: 9
Iteration t: 1
Appointment numbers: {0, 1, 2, 3, 4, 5}
Appointment durations for the judges: [6, 2, 8, 5, 2, 6]
Appointment durations for the legal assistants: [7, 2, 9, 9, 2, 8]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 7
Appointment 1 judge 1 timeslot 17
Appointment 2 judge 1 timeslot 19
Appointment 0 legal assistant 0 timeslot 1
Appointment 1 legal assistant 0 timeslot 17
Appointment 2 legal assistant 0 timeslot 19
Appointment 3 is not scheduled
Appointment 4 is not scheduled
Appointment 5 is not scheduled
Iteration t: 2
```

```
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [5, 2, 6]
Appointment durations for the legal assistants: [9, 2, 8]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [5, 2, 6]
Appointment durations for the legal assistants: [9, 2, 8]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Objective value = 916.3531957670531
Iteration n: 10
Iteration t: 1
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [10, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 2
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [10, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 is not scheduled
Appointment 1 is not scheduled
Appointment 2 is not scheduled
Iteration t: 3
Appointment numbers: {0, 1, 2}
Appointment durations for the judges: [6, 2, 7]
Appointment durations for the legal assistants: [10, 2, 10]
OptimizationStatus.OPTIMAL
Appointment 0 judge 1 timeslot 7
Appointment 1 judge 1 timeslot 16
Appointment 2 judge 1 timeslot 18
Appointment 0 legal assistant 1 timeslot 3
Appointment 1 legal assistant 1 timeslot 16
Appointment 2 legal assistant 1 timeslot 18
Objective value = 923.03446927044
```

H Input MDP test data (enlarging state space)

In this appendix we show the availability used when testing with larger state spaces. When enlarging the horizon, but still using two judges and two legal assistants, we used the following availability:

```
Availability_judges =
    [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
    0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
    1, 1, 1, 1, 1, 1, 1, 0,0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1,
    1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0,
    0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]

Availability_legalassistants =
    [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
    1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1,1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
    0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1]]
```

When using the same horizon as for the toy-sized problem, but using eight judges and ten legal assistants, we used the following availability:

```
Availability_judges =
    [[1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
    0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 0, 0, 1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0,
    1, 1, 1, 1, 1, 1, 1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]

Availability_legalassistants =
    [[1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0,
    0, 1, 1, 1, 1, 1, 1, 1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1,
    1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 0, 0, 0, 0,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1] ,
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```