Approximate dynamic programming for weakly coupled Markov decision processes with perfect and imperfect information

Mahshid Salemi Parizi

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2018

Reading Committee:

Archis Ghate, Chair

Shan Liu

Christina Mastrangelo

Program Authorized to Offer Degree:
Industrial & Systems Engineering

University of Washington

Abstract

Approximate dynamic programming for weakly coupled
Markov decision processes with perfect and
imperfect information

Mahshid Salemi Parizi

Chair of the Supervisory Committee:
Professor Archis Ghate
Industrial & Systems Engineering

A broad range of optimization problems in applications such as healthcare operations, revenue management, telecommunications, high-performance computing, logistics and transportation, business analytics, and defense, have the following form. Heterogeneous service requests arrive dynamically and stochastically over slotted time. A request may require multiple resources to complete. The decision-maker may collect a reward on successfully completing a service request, and may also incur costs for rejecting requests or for delaying service. The decision-maker's goal is to choose how to dynamically allocate limited resources to various service requests so as to optimize a certain performance-metric.

Despite the prevalence of these problems, a majority of existing research focuses only on their stylized models. While such stylized models are often insightful, several experts have commented in recent literature reviews that their applicability is limited in practice. On the other hand, more realistic models of such problems are computationally difficult to solve owing to the curse of dimensionality.

The research objective of this dissertation is to build Markov decision process (MDP) models of four classes of dynamic resource allocation problems under uncertainty, and then to develop algorithms for their approximate solution. Specifically, most MDP models in this dissertation will possess the so-called weakly coupled structure. That is, the MDP is com-

posed of several sub-MDPs; the reward is additively separable and the transition probabilities are multiplicatively separable over these sub-MDPs; and the sub-MDPs are joined only via linking constraints on the actions they choose. The dissertation proposes mathematical programming-based and simulation-based approximate dynamic programming methods for their solution. Performance of these methods is compared against one-another and against heuristic resource allocation policies. An outline of this dissertation is described below.

Chapter 1 investigates a class of scheduling problems where dynamically and stochastically arriving appointment requests are either rejected or booked for future slots. A customer may cancel an appointment. A customer who does not cancel may fail to show up. The planner may overbook appointments to mitigate the detrimental effects of cancellations and no-shows. A customer needs multiple renewable resources. The system receives a reward for providing service; and incurs costs for rejecting requests, appointment delays, and overtime. Customers are heterogeneous in all problem parameters. The chapter provides a weakly coupled MDP formulation of these problems. Exact solution of this MDP is intractable. An approximate dynamic programming method rooted in Lagrangian relaxation, affine value function approximation, and constraint generation is applied to this weakly coupled MDP. This method is compared with a myopic scheduling heuristic on 1800 problem instances. These numerical experiments show that there was a statistically significant difference in the performance of the two methods in 77% of these instances. Of these statistically significant instances, the Lagrangian method outperformed the myopic method in 97% instances.

Chapter 2 focuses on a class of non-preemptive scheduling problems, where a decision-maker stochastically and dynamically receives requests to work on heterogeneous projects over discrete time. The projects are comprised of precedence-constrained tasks that require multiple resources with limited availabilities. Incomplete projects are held in virtual queues with finite capacities. When a queue is full, an arriving project must be rejected. The projects differ in their stochastic arrival patterns; completion rewards; rejection, waiting and operating costs;

activity-on-node networks and task durations; queue capacities; and resource requirements. The decision-maker's goal is to choose which tasks to start in each time-slot to maximize the infinite-horizon discounted expected profit. The chapter provides a weakly coupled MDP formulation of such dynamic resource-constrained project scheduling problems (DRCPSPs). Unfortunately, existing mathematical programming-based approximate dynamic programming techniques (similar to those in Chapter 1) are computationally tedious for DRCPSPs owing to their exceedingly large scale and complex combinatorial structure. Therefore, the chapter applies a simulation-based policy iteration method that uses least-squares fitting to tune the parameters of a value function approximation. The performance of this method is numerically compared against a myopic scheduling heuristic on 480 randomly generated problem instances. These numerical experiments show that the difference between the two methods was statistically significant in about 60% of the instances. The approximate policy iteration method outperformed the myopic heuristic in 74% of these statistically significant instances.

In Chapters 1 and 2, the decision-maker is assumed to know all parameters that describe the weakly coupled MDPs. Chapter 3 investigates an extension where the decision-maker only has imperfect information about the weakly coupled MDP. Rather than only focusing on weakly coupled MDPs that arise in specific applications as in Chapters 1 and 2, Chapter 3 works with general weakly coupled MDPs. Two different scenarios with imperfect information are studied. In the first case, the transition probabilities for each subproblem are unknown to the decision-maker. In particular, these transition probabilities are parameterized, and the decision-maker does not know the values of these parameters. The decision-maker begins with prior probabilistic beliefs about these parameters and updates these beliefs using Bayes' Theorem as the state evolution is observed. This yields a Bayes-adaptive weakly coupled MDP formulation whose exact solution is intractable. Computationally tractable approximate dynamic programing methods that combine semi-stochastic certainty equivalent

control or Thompson sampling with Lagrangian relaxation are proposed. These ideas are applied to a class of dynamic stochastic resource allocation problems and numerical results are presented. In the second case, the decision-maker cannot observe the actual state of the system, but only receives a noisy signal about it. The decision-maker thus needs to probabilistically infer the actual state. This yields a partially observable weakly coupled MDP formulation whose exact solution is also intractable. Computationally tractable approximate dynamic programming methods rooted in semi-stochastic certainty equivalent control and Thompson sampling are again proposed. These ideas are applied to a restless multi-armed bandit problem and numerical results are presented.

Chapter 4 investigates a class of sequential auction design problems under imperfect information. There, the resource corresponds to the seller's inventory on hand, which is to be allocated to dynamically and stochastically arriving buyers' requests (bids). In particular, the seller needs to decide lot-sizes in a sequential, multi-unit auction setting, where bidder demand and bid distributions are not known in their entirety. The chapter formulates a Bayes-adaptive MDP to study a profit maximization problem in this scenario. The number of bidders is Poisson distributed with a Gamma prior on its mean, and the bid distribution is categorical with a Dirichlet prior. The seller updates these beliefs using data collected over auctions while simultaneously making lot-sizing decisions until all inventory is depleted. Exact solution of this Bayes-adaptive MDP is intractable. The chapter proposes three approximation methods (semi-stochastic certainty equivalent control, knowledge gradient, and Thompson sampling) and compares them via numerical experiments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

MULTI-CLASS, MULTI-RESOURCE, ADVANCE SCHEDULING

The work reported here has appeared in the journal Computers and Operations Research, 67, 90-101, 2016.

1.1 Introduction

We present a formal abstraction, a Markov decision process (MDP) model, and an approximate dynamic programming (ADP) solution algorithm for a class of problems motivated by recent literature reviews about patient scheduling [18, 32, 44]. To understand this class of problems concretely, imagine a scheduler at a hospital, who dynamically and stochastically receives appointment requests for different types of elective surgeries. Distinct types of surgeries require different kinds and amounts of resources. For example, one type of surgery may require a two-hour block of time in the operating room, one surgeon, two surgeon's assistants, a nurse and a certain medical equipment. Another type of surgery may require a two-hour block of time, two surgeons, no assistants, and one nurse. Indeed, Gupta and Denton [44] stated that surgeries require "clinical assistants, nurses, anesthesiologists, surgeons, operating rooms, diagnostic devices, surgical tools and other equipment." Since these surgeries are elective, the requests may be rejected, or scheduled to a future day in a booking horizon. The scheduler may also receive cancellations for surgeries that were booked in advance. Similarly, some scheduled surgeries may turn out to be no-shows "at the last minute." To mitigate the adverse effects of such cancellations and no-shows on system-performance, the scheduler may overbook surgeries with the hope that additional resources may be summoned if and when deemed necessary. The hospital may receive different rewards for performing

different types of surgeries and incur costs/penalties for rejecting arriving request. It may also incur an indirect waiting cost depending on the delay between when the request arrives and the future day for which it is scheduled. Finally, if additional resources are requested to tackle overbookings, the hospital may incur a cost of overtime. The scheduler is interested in booking these dynamic, stochastic surgery requests so as to optimize an appropriate metric of performance over an infinite horizon because there is no known time of extinction for this system. We call this type of problems multi-class, multi-resource, advance scheduling with cancellations, no-shows and overbooking.

These advance scheduling problems also arise in healthcare operations other than elective surgery scheduling. For example, a clinic that employs one or more physicians must also book stochastically and dynamically arriving patient requests and plan for no-shows and cancellations (estimates of no-show rates vary widely in the literature depending on several factors: for instance, 42% [67]; 10% [17, 25, 112]; 15%-30% [28]; 20% [53]; 3%-80% [96]). Similarly, a radiological facility that houses different kinds of imaging machines also needs to book requests for distinct classes of diagnostic exams [83, 106].

Beyond healthcare, such advance scheduling problems arise in revenue management [31]; at maintenance and repair facilities [85]; in manufacturing [69, 104]; in military reconnaissance and combat planning [1]; in communication networks [5, 93, 111]; and in high-performance computing [108].

Despite the prevalence of these problems in a broad range of applications, research on their mathematical models and dynamic stochastic optimization is sparse. In the healthcare context, Cayirli and Veral [18] noted that the use of patient classification in scheduling problems which consider dynamically and stochastically arriving requests was very limited in the literature. They discussed potential advantages of classifying patients by relaxing the usual assumption of homogeneous patients. They also stated that no-shows is a major exogenous factor that affects the performance of any appointment scheduling system. They commented "no rigorous research exists which investigates possible approaches to adjusting the appointment systems in order to minimize the disruptive effects of no-shows, walk-ins,

and/or emergencies." Daggy et al. [26] discussed the importance of considering no-shows and overbooking in patient scheduling models. Gupta and Denton [44] stated that cancellations and no-shows are open challenges in appointment scheduling and that overbooking could be used to tackle this difficulty.

The main reason why most researchers seem to have balked from building mathematical models and developing optimization algorithms for these problems is the well-founded belief that any such model would be inevitably large-scale and hence their exact solution computationally difficult. This concern is appropriately rooted in the three well-known curses of dimensionality in dynamic programming (DP) — the curse of state-space, the curse of action-space, and the curse of expectation [89]. Indeed, Truong [106] stated that "advance scheduling problems have generally been deemed intractable."

In this chapter, we attempt to fill this gap in the literature by making the following contributions:

1. we formally introduce a group of multi-class, multi-resource, advance scheduling problems with no-shows, cancellations, and overbooking; to the best of our knowledge, this class of problems has not yet been studied in the existing literature;

2. we provide an MDP model for this class of problems;

3. we show that this MDP is weakly coupled (see Adelman and Mersereau[3]) — that is, the state- and action-spaces are Cartesian products of finite sets over different job classes, the immediate expected rewards are additively separable over job classes, transition probabilities are multiplicatively separable over job classes, and the only feature that links different job classes is the resource constraints;

4. we argue that this weakly coupled MDP is large-scale; that is, a naive implementation of the standard Lagrangian relaxation method of Adelman and Mersereau [3] would be computationally challenging;

5. we then show how Lagrangian relaxation with constraint generation and affine value function approximation, which was originally developed by Gocgun and Ghate [41] for solving general large-scale weakly coupled MDPs, can be applied to this class of problems;

6. we compare the resulting Lagrangian policy with a myopic heuristic via extensive numerical experiments that include eighteen hundred problem instances;

7. our numerical experiments suggest that the Lagrangian solution outperforms the myopic solution with a statistically significant margin on a large majority of the problem instances.

This chapter is organized as follows. We review related literature in the next section. Our class of advance scheduling problems is formally introduced in Section 1.3. A weakly coupled MDP formulation for this class of problems is described in Section 1.4. An ADP algorithm based on Lagrangian relaxation of this MDP is presented in Section 1.5. Computational experiments are included in Section 1.6.

1.2 Literature review

The problems considered in this study are categorized as advance scheduling problems because stochastically and dynamically arriving jobs are scheduled to future slots within a booking horizon. The first part of our literature survey (Section 1.2.1) therefore focuses on papers that model advance scheduling in this manner; the second part (Section 1.2.2) reviews papers that do not model advance scheduling the way we envision here, but do include no-shows and/or overbooking. These two parts focus on papers that consider healthcare as their main application domain. The advance scheduling problems we study do share some similarities, albeit tenuous, with other problems studied in application areas such as revenue management, machine scheduling, admission control in queues, and inventory allocation. We do not review those papers here but refer the reader to various surveys about these appli-

cation areas: Talluri and van Ryzin [102] for revenue management; Leung [68] for machine scheduling; Green and Savin [43] for admission control in queues; and Ha [45, 46] for inventory allocation. Our advance scheduling framework is in contrast with allocation scheduling, where an arriving job is either rejected or put in a queue and jobs in the queue are dynamically selected for service later [75]. We do not review the vast literature on allocation scheduling but instead refer the reader to Truong [106] for an up-to-date survey.

1.2.1 Literature on advance scheduling

Patrick et al. [83] presented an MDP model and a linear programming based ADP method to book stochastically and dynamically arriving patient requests to future slots at a diagnostic resource in a public healthcare setting. That paper considered a single resource. Gocgun and Ghate [41], in a paper that was based on Gocgun's doctoral dissertation [39], extended the model in Patrick et al. [83] to include multiple resources. Gocgun and Ghate [41] showed that their problem can be formulated as a weakly coupled MDP and proposed an extension of the Lagrangian relaxation method in Adelman and Mersereau [3] and in Hawkins [50] for approximately solving this MDP. Neither the model in Patrick et al. [83] nor that in Gocgun and Ghate [41] included cancellations, no-shows, or overbooking. Saure et al. [99] applied the method in Patrick et al. to scheduling problems in radiation therapy. Gocgun and Puterman [42] applied an extension of Saure et al. [99] to chemotherapy appointment scheduling. This extension considered target dates for appointments and a tolerance around that target in which the scheduler has the flexibility to book an appointment. Again, these papers did not incorporate no-shows, cancellations, overbooking or multiple resources.

An application of advance scheduling to multi-class, multi-resource surgeries is developed in the Master's thesis of Astaraky [8]. That model aims to minimize a combination of waiting time till the surgery date, overtime in the operation room, and congestion in hospital wards. Astaraky's model does not consider no-shows and uses patient-discharge from the booking schedule that can be viewed as a type of cancellation. The state in his model includes the number of patients booked in the booking horizon, the number of patients in post-operative

recovery, and the demand waiting to be scheduled. This state is different from what we need in this chapter. For instance, Astaraky does not need to store information about when a booking decision was made in the state. Astaraky employed a simulation-based approximate dynamic programming method that was based on the idea of pre- and post-decision states as described in [89] and that used policy iteration rooted in least-squares regression.

Liu et al. [72] developed an advance scheduling model with no-shows, cancellations and overtime. They only considered a single class of patients and a single resource. Feldman et al. [34] extended the model in Liu et al. by adding patient preferences.

Patrick [82] provided an MDP model that allows wait-time dependent no-show rates and also uses overbooking. He showed via simulation that using a booking horizon might work better than implementing an open-access policy whose essence is "do today's demand today." Earlier case studies and simulation studies of open-access policies appeared in Murray and Tantau [78]; Kopach et al. [62]; and Robinson and Chen [92]. A variety of challenges in implementing open-access were discussed in Gupta and Denton [44] and in Kopach et al. [62].

Truong [106] studied advance scheduling problems with one emergency class of patients, one regular class of patients, and multiple resources. She provided the first-ever analytical results for this class of problems, remarkably leading to an efficient algorithm for exact calculation of an optimal policy. No-shows, cancellations, or overbooking were not modeled.

Herring and Herrmann [51] constructed a finite horizon stochastic dynamic programming model for allocating operating room capacity over N days. This model shared some similarities with capacity allocation models in the airline revenue management literature. The key feature that distinguished the Herring and Herrmann [51] model from previous revenue management models was that they allowed the lower-priority demand to form a waiting list. This complicated their state-space, but interestingly, they were still able to derive optimal policies that behaved in a manner similar to revenue management. Herring and Herrmann [51] did not model no-shows, cancellations, or overbooking.

A general dynamic capacity allocation problem was modeled by Erdelyi and Topaloglu

in [30]. Their model involved allocating a fixed amount of daily capacity to different jobs that arrive randomly. They did not model no-shows or cancellations.

1.2.2 Literature on no-shows, cancellations, and overbooking

There is a parallel body of literature on patient scheduling that considers no-shows, and/or cancellations, and/or overbooking but without advance scheduling as envisioned in this chapter.

Kim and Giachetti [60] formulated a static, single session appointment scheduling problem. Their goal was to choose the number of appointments to book in this session. A single class of patients was considered. Their objective function included revenue generated by serving patients, physician overtime cost and a penalty cost for patients who leave without being served.

LaGanga and Lawrence [65] also considered a similar static problem in which the goal was to select the number of patients scheduled for each slot in a day. No-show rates were dependent on the time of the day. A single class of patients was included. They analytically derived simple properties of the overbooking solution and then used complete enumeration for solving small problems and employed heuristic search for approximate solution of larger problems. Zacharias and Pinedo [115] studied a problem similar to LaGanga and Lawrence [65] but allowed for heterogeneous patients. Other similar works, separated by idiosyncratic differences in their problem statements and model formulations, include Kaandorp and Koole [58]; Hassin and Mendel [49]; Klassen and Yoogalingam [61]; Begen and Queyranne [12]; Cayirli et al. [19].

Huang and Zuniga [54] developed a simulation method to find a threshold of no-show probability for a patient booked in a particular slot that should induce overbooking for that slot.

Muthuraman and Lawley [79] presented a scheduling model that combined certain features from more traditional static models with those of dynamic models. In their model, patients are booked for slots of fixed lengths in one day through a call-in process that occurs

before the day begins. The scheduler uses an estimate of a caller's no-show probability to schedule him/her for one of the slots before the phone call ends. Service times are exponential and patients who do not complete service in a slot spill over to the next one. Thus, a queue of waiting patients is formed. The objective is to maximize expected revenue, which is composed of a reward for completing service, the patients' waiting cost and an overtime cost. Chakraborty et al. [20] also used a similar approach to sequential clinical scheduling with patient no-shows. Their model extended the model of Muthuraman and Lawley [79] by allowing arbitrary service time distributions. Zeng et al. [116] also extended the work in Muthuraman and Lawley [79] by providing two additional, more sophisticated scheduling algorithms.

Luo et al. [74] considered a fixed time horizon and a single class of patients; at the beginning of this time horizon, the planner wants to determine the number of patients to be scheduled and also the times at which their appointments begin. They allowed for the possibility of patient no-shows and included exponentially distributed service times. An interesting and unique feature of their model was that it allowed for service-interruptions. These interruptions were induced, for example, by events that forced the server to divert attention to other activities such as treating emergency patients. The objective function in Luo et al. [74] included a waiting cost and an overtime cost. Evaluation of this objective called for the solution of Kolmogorov differential equations. This rendered exact solution intractable. Luo et al. [74] therefore used a heuristic method for approximate solution.

Liu and Ziya [71] considered an appointment system where the provider can control the demand arrival rate. Their model included a single server queue, a single class of patients, and state-dependent no-show probabilities. They first assumed that the planner does not have the option to overbook and provided a characterization of the optimal demand arrival rate. Later they incorporated overbooking in a stylized fashion by making the service rate a decision variable in addition to the demand arrival rate. They derived a characterization of optimal solutions to this joint optimization problem.

1.3 Formal problem statement

In light of the broad variety of applications of advance scheduling, we will use the words patients and jobs interchangeably in this chapter. As stated above, since the system under consideration has no predetermined time of extinction, we construct an infinite-horizon model as is common in the literature on advance scheduling. The actual length of each time-period in this model will depend on the application. For expository convenience and brevity, we will nevertheless call each time-period a day.

Our formal problem statement below is best-suited for facilities that operate in the following manner. Stochastically arriving requests for appointments are collected from different types of jobs online or through voice calls throughout a day. Similarly, stochastically arriving cancellations for previously booked jobs are also logged throughout a day. The scheduler reviews these requests before the facility begins its daily operations. The scheduler also knows the numbers and types of jobs that are already booked in the booking horizon, as well as the day in the past when these booking decisions were made. As we shall see, it is essential to know when these booking decisions were made because we allow our cancellation and no-show probabilities to depend on this quantity. Requests for appointments can be booked for any day in the booking horizon or they can be rejected. The scheduler conveys these decisions to the jobs before the beginning of the daily operations at the facility. At this time, the scheduler also decides the numbers and types of jobs that the facility will serve using overtime that day. A job that is scheduled for today, may, with some probability, fail to show up for its appointment without canceling it ahead of time. In order to prepare the facility for potential additional work, overtime decisions for the day have to be made before these stochastic no-shows are realized that day.

Our class of advance scheduling problems is more precisely described below; Table 1.1 summarizes the notation used.

1. (Heterogeneous job types) The index set of job types is denoted by $\mathcal{I} = \{1, 2, \ldots, I\}$.

2. (Stochastic arrivals) A random number $\mathbf{\Delta}_i$ of type-i jobs arrives in one day; $\mathbf{\Delta}_i$ takes values from the set $\{0, 1, \ldots, N_i\}$, where N_i are positive integers. The probability that n_i type-i jobs arrive in one day is $p_i(n_i)$. Arrivals across different types are independent.

3. (Multiple resource constraints) $\mathcal{J} = \{1, \ldots, J\}$ denotes the set of resources and b_j (positive integer) is the total quantity (including overtime) of resource $j \in \mathcal{J}$ available for use in each day. The regular quantity (without overtime) of resource $j \in \mathcal{J}$ available each day is denoted by b'_j. Thus, clearly, $b'_j \leqslant b_j$ for all j. In order to perform each job of type i, a non-negative integer amount a_{ij} of resource j is required. We assume that for each job-type i, there is at least one resource j such that $a_{ij} > 0$. We note here that the word overtime is used somewhat loosely as it may not actually refer to a literal time. The word essentially refers to the act of serving jobs using extra resources than what is normally available at the facility by incurring a cost.

4. (Immediate decisions) The scheduler can either reject or accept an arriving request. An accepted request is booked for one day somewhere in a booking horizon. The booking horizon consists of the current day and T more future days. Our model allows the scheduler to overbook — that is, to book more jobs than can be handled by the regular resource capacities. A decision is thus made about how many and what type of jobs could be served in overtime today.

5. (Stochastic cancellations) Each day, a random number of jobs booked for future days may cancel their appointments. We assume that jobs for which an appointment decision was just made today will not cancel that same day. The term $0 \leqslant q^i_{lk} \leqslant 1$ denotes the probability that a type-i job booked $1 \leqslant l \leqslant T-1$ days earlier for $1 \leqslant k \leqslant T-1$ days from today will cancel their appointment today, independently of everything else. Note here that we allow the cancellation probability to depend on the job-type as well as on how long ago the job was booked and on how far into the future it was booked. The random number of type-i jobs that will cancel today is denoted by $\mathbf{Y}^i_{lk}$, with l

and k as just defined. Note that an l, k combination in this context is meaningful only when $l + k \leqslant T$ because of our booking horizon.

6. (Stochastic no-shows) Some of the jobs that are booked for today might not show up (without canceling their appointment ahead of time) — these are called no-shows. We use $0 \leqslant \theta_l^i \leqslant 1$ to denote the probability that, independently of everything else, a type-i job that was scheduled l days ago for today will show up for its appointment. Thus, $1 - \theta_l^i$ is the no-show probability. Again, we allow this probability to depend on the job-type as well as on how long ago the job was booked. We also define $\mathbf{S}_l^i$, for $1 \leqslant l \leqslant T$, as the random number of type-i jobs that were booked l days ago for today and that will show up for their appointment. We assume that jobs that were just booked today for today will show up for their appointments.

7. (Costs) The following costs are incurred:

   - (Delay cost) The delay cost of scheduling a type-i job k days from today in the booking horizon is given by the non-negative function $C_i(k)$, for $0 \leqslant k \leqslant T$. It is natural to assume that $C_i(k)$ is non-decreasing in k.
   - (Penalty cost of rejection) The planner incurs a penalty cost of G_i per type-i job that is rejected.
   - (Overtime Cost) The cost of serving each type-i job using overtime is h_i.

8. (Revenue) The reward of serving each type-i job is R_i.

9. (Total discounted profit objective) The goal is to book or reject arriving jobs so as to maximize the total discounted expected profit (reward minus cost) over an infinite horizon; we use $0 < \alpha < 1$ to denote the discount factor.

We now present an MDP formulation for this class of problems and establish that this

Notation	description
$\mathcal{I} = \{1, 2, \ldots, I\}$	index set of job types
N_i	maximum possible number of type-i arrivals in one day
$\mathbf{\Delta}_i \in \{0, 1, \ldots, N_i\}$	random number of type-i arrivals in one day
$p_i(n_i)$	probability that n_i type-i arrivals occur in one day
$\mathcal{J} = \{1, 2, \ldots, J\}$	index set of resources
b_j	total amount of resource j available each day including overtime
b'_j	resource j available each day excluding overtime
a_{ij}	resource consumption coefficient for type-i jobs for resource j
T	booking horizon
$\mathbf{Y}^i_{lk}$	random number of type-i cancellations booked l days ago for k days from now
q^i_{lk}	cancellation probability
$\mathbf{S}^i_l$	random number of type-i show ups
θ^i_l	show up probability
$C_i(k)$	delay cost for booking a type-i job for k days from now
G_i	cost of rejecting a type-i job
h_i	cost of serving a type-i job using overtime
R_i	reward for serving a type-i job
α	discount factor

Table 1.1: A summary of notation employed in describing our scheduling problems.

formulation has a weakly coupled structure. This structure is then exploited in Section 1.5 to propose an efficient approximate solution algorithm.

1.4 Weakly coupled MDP formulation

As is standard in MDP models [91, 94], our formulation is composed of several components: states, decisions, state transitions, immediate expected rewards, and Bellman's equations. These attributes are described one-by-one in detail in the next few sections.

1.4.1 MDP states

The state of our MDP model is defined as $x = (x^1, x^2, \ldots, x^I)$, where each x^i is a vector written as $x^i = (x^i_0; x^i_{lk})$, for $1 \leqslant l \leqslant T$ and $1 \leqslant k + l \leqslant T$. In this vector, x^i_0 is the number of appointment requests from type-i jobs. Moreover, x^i_{lk} is the number of type-i jobs that were booked l days earlier for k days from today. An illustration of this state is shown in

Figure 1.1(a) for $T = 5$. Note that $k = 0$ is possible and corresponds to the number of jobs booked for today. However, l cannot be zero because x^i_{lk} denotes the state of the system before decisions are made today. Since the maximum number of type-i jobs that can arrive over a day is N_i, we have that $x^i_0 \leqslant N_i$. We therefore define the set $X^i_0 = \{0, 1, \ldots, N_i\}$. Also let $M_i = \min_{j \in \mathcal{J}} \left\lfloor \frac{b_j}{a_{ij}} \right\rfloor$ be the largest number of type-i jobs that could be booked for any given day (by utilizing the overtime capacity). We define set $X^i_{lk} = \{0, 1, \ldots, M_i\}$ for each combination of l and k, and let $X^i = \{X^i_0,\ X^i_{lk},\ 1 \leqslant l \leqslant T,\ 1 \leqslant l + k \leqslant T\}$. Finally, let $X = X^1 \times X^2 \times \ldots \times X^I$. The set $\bar{X}$ of all feasible states is then given by

$$\bar{X} = \left\{ x \in X : \sum_{i=1}^{I} a_{ij} \sum_{l=1}^{T-k} x^i_{lk} \leqslant b_j,\ j \in \mathcal{J},\ 0 \leqslant k \leqslant T-1 \right\}. \tag{1.1}$$

That is, state x is feasible if and only if the number of jobs of different types that are currently booked for k days from now respect the resource constraint for each resource j.

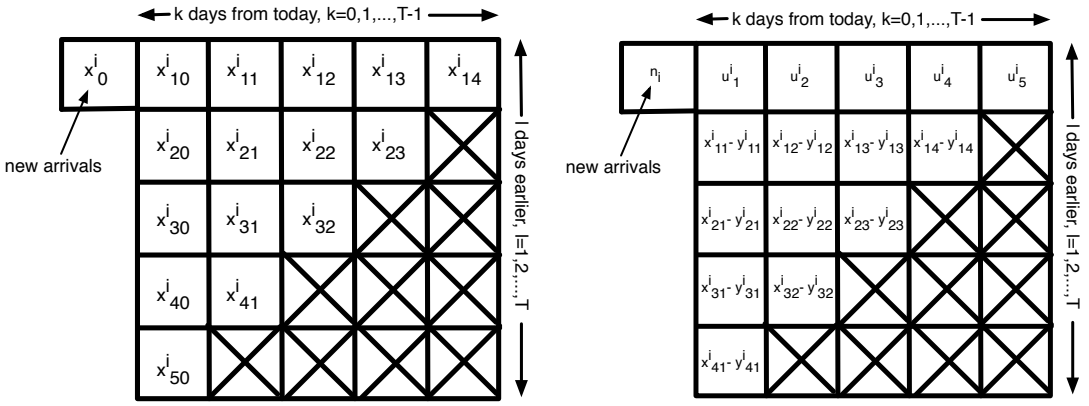

Figure 1.1: (a) Illustration of the state x^i in our MDP for job-type i when $T = 5$. The cells in the 5×5 square grid show x^i_{lk} for $1 \leqslant l \leqslant T$ (rows) and $0 \leqslant k \leqslant T-1$ (columns) such that $l + k \leqslant T$. Combinations l, k such that $l + k > T$ are not meaningful because the booking horizon is not long enough to accommodate them. The corresponding cells have been crossed out. (b) The new state for type-i jobs when a decision $u \in \bar{U}(x)$ is made in state x, the number of cancellations is y^i_{lk}, and the number of new arrivals is n_i.

1.4.2 MDP decisions

The decision vector is $u = (u^1, u^2, \ldots, u^I)$, where each u^i is a $T+2$-dimensional vector written as $u^i = (v^i; u^i_0; u^i_1; \ldots; u^i_T)$. Here, u^i_k, for $0 \leqslant k \leqslant T$, is the number of type-$i$ jobs (among the ones that just arrived) that we choose to schedule for k days from now in the booking horizon. The first component v^i, is the potential number of type-i jobs that we will serve today using overtime. For each $i \in \mathcal{I}$, $x^i \in X^i$, and $0 \leqslant k \leqslant T$, let $U^i_k(x^i) = \{0, 1, \ldots, x^i_0\}$. Also let

$$U^i(x^i) = \Big\{ u^i_k \in U^i_k(x^i),\ 0 \leqslant k \leqslant T,\ v^i : \sum_{k=0}^{T} u^i_k \leqslant x^i_0,\ v^i \leqslant u^i_0 + \sum_{l=1}^{T} x^i_{l0},$$
$$\sum_{l=1}^{T-k} x^i_{lk} + u^i_k \leqslant M_i,\ \text{for } 0 \leqslant k \leqslant T-1,\ \text{and } u^i_T \leqslant M_i \Big\}.$$

In this set definition, the first group of constraints ensures that the number of jobs newly booked is not more than the total number of requests. The second group of constraints ensures that the potential number of jobs served using overtime is not more than the total number of jobs booked for today. The last two groups of constraints ensure that the total number of jobs booked for any one future day in the booking horizon is not more than the maximum possible as determined by the resource capacities. These constraints are included in this set definition to keep this set finite.

Note here that, according to our state and decision definitions, $x^i_0 - \sum_{k=0}^{T} u^i_k \geqslant 0$ is the number of type-i appointment requests that is rejected. Also, for $x \in X$, let $U(x) = U^1(x^1) \times U^2(x^2) \times \ldots \times U^I(x^I)$. The set $\bar{U}(x) \subseteq U(x)$ of all decision vectors feasible in state $x \in \bar{X}$ is then defined as

$$\bar{U}(x) = \Bigg\{ (u^1, u^2, \ldots, u^I) \in U(x) : \sum_{i=1}^{I} a_{ij} \Big(\sum_{l=1}^{T-k} x^i_{lk} + u^i_k \Big) \leqslant b_j,\ j \in \mathcal{J},\ 0 \leqslant k \leqslant T-1,$$
$$\sum_{i=1}^{I} a_{ij} u^i_T \leqslant b_j,\ \text{and}\ \sum_{i=1}^{I} a_{ij} \Big(\sum_{l=1}^{T} x^i_{l0} + u^i_0 - v^i \Big) \leqslant b'_j,\ j \in \mathcal{J} \Bigg\}. \tag{1.2}$$

In this set definition, the first two groups of constraints make sure that the total resource consumption by all jobs booked for today and for any one future day in the booking horizon is not more than the total resource capacities including overtime. The third group of inequalities ensures that the jobs potentially served today in regular time respect the regular time resource constraints.

1.4.3 MDP state transitions and immediate expected rewards

In this subsection, we provide expressions for state transition probabilities and immediate expected rewards as functions of the above states and decisions.

1.4.3.1 State transition probabilities

Suppose the state at the beginning of today before operations begin at the facility is $x = (x^1, x^2, \ldots, x^I)$, where

$$x^i = (x^i_0; x^i_{1,0}, x^i_{2,0}, \ldots, x^i_{T,0}; x^i_{1,1}, x^i_{2,1}, \ldots, x^i_{T-1,1}; \ldots; x^i_{i,t}, x^i_{2,t}, \ldots, x^i_{T-t,t}; \ldots; x^i_{1,T-1}),$$

as was depicted earlier in Figure 1.1(a). Recall that $\mathbf{Y}^i_{lk}$ denotes the random number of type-i jobs that cancel (out of the x^i_{lk} scheduled jobs). Since each such job cancels with probability q^i_{lk}, independently of everything else, $\mathbf{Y}^i_{lk}$ is a binomial random variable with parameters x^i_{lk} and q^i_{lk}. We define $Q^i_{lk}(x^i_{lk}, y^i_{lk})$ as the probability that $0 \leqslant y^i_{lk} \leqslant x^i_{lk}$ jobs are cancelled. We therefore have,

$$Q^i_{lk}(x^i_{lk}, y^i_{lk}) = \binom{x^i_{lk}}{y^i_{lk}} (q^i_{lk})^{y^i_{lk}} (1 - q^i_{lk})^{x^i_{lk} - y^i_{lk}}.$$

A job which was scheduled l days ago for k days from today in the current state, will become a job that was scheduled $l + 1$ days ago for $k - 1$ days from tomorrow in the next state. On selecting decision $u \in \bar{U}(x)$ in state x, the ith component of the next state therefore equals

$$x'^i = (n_i; u^i_1, x^i_{1,1} - y^i_{1,1}, \ldots, x^i_{T-1,1} - y^i_{T-1,1}; u^i_2, x^i_{1,2} - y^i_{1,2}, \ldots, x^i_{T-2,2} - y^i_{T-2,2}; \ldots;$$

$$u^i_{t+1}, x^i_{1,t+1} - y^i_{1,t+1}, \ldots, x^i_{T-t-1,t+1} - y^i_{T-t-1,t+1}; \ldots; u^i_T), \quad (1.3)$$

with probability

$$p_i(n_i) \prod_{k=1}^{T-1} \prod_{l=1}^{T-k} Q^i_{lk}(x^i_{lk}, y^i_{lk}),$$

for $0 \leqslant n_i \leqslant N_i$. This state transition is illustrated in Figure 1.1(b). The next state then is $x' = (x'^1, x'^2, \ldots, x'^I)$ with probability $\prod_{i=1}^{I} p_i(n_i)\Psi^i(x^i, x'^i)$, where we have used the notation

$$\Psi^i(x^i, x'^i) = \prod_{k=1}^{T-1} \prod_{l=1}^{T-k} Q^i_{lk}(x^i_{lk}, x^i_{lk} - x'^i_{lk}) \quad (1.4)$$

for brevity.

1.4.3.2 Immediate expected rewards

The immediate expected profit earned today is denoted by $f(x, u) = \sum_{i=1}^{I} f_i(x^i, u^i)$, where, for all $x^i \in X^i$ and $u^i \in U^i(x^i)$, we define,

$$f_i(x^i, u^i) = R_i\Big(u^i_0 + \sum_{l=1}^{T} x^i_{l,0}\theta^i_l\Big) - G_i\Big(x^i_0 - \sum_{k=0}^{T} u^i_k\Big) - \sum_{k=0}^{T} u^i_k C_i(k) - \alpha h_i v^i. \quad (1.5)$$

This expression includes four terms. The first term accounts for the reward earned for serving type-i jobs today. These jobs include the ones booked today, denoted u^i_0, and also the expected number of jobs (booked previously) that shows up today. Note that because the random number of type-i jobs that shows up today and that were booked l days ago, $\mathbf{S}^i_l$, is a binomial random variable with success probability θ^i_l, its expected value equals $x^i_{l,0}\theta^i_l$. That is why, we have added the term $\sum_{l=1}^{T} x^i_{l,0}\theta^i_l$ in our reward calculation. The second term accounts for the cost of rejecting $x^i_0 - \sum_{k=0}^{T} u^i_k$ jobs. The third term corresponds to the cost of indirect waiting for jobs that are booked today for different days in the booking horizon. The fourth term is the cost of potentially serving v^i jobs of type-i using overtime today.

1.4.4 Bellman's equations and verification of weakly coupled structure

Let $W(x)$ be the maximum infinite-horizon discounted expected profit earned starting in state $x \in \bar{X}$. Then, for all $x \in \bar{X}$, the Bellman's equations for our MDP model are given by

$$W(x) = \max_{u \in \bar{U}(x)} \left\{ f(x,u) + \alpha \sum_{x' \in \bar{X}} \left(\prod_{i=1}^{I} p_i(n_i) \Psi^i(x^i, x'^i) \right) W(x') \right\}, \tag{1.6}$$

where $x' = (x'^1, \ldots, x'^I)$ is given by Equation (1.3). This MDP satisfies all the requirements for a weakly coupled MDP except that the set of feasible states $\bar{X}$ in (1.1) does not have a Cartesian product structure. We use a technique proposed in Gocgun and Ghate [41] to transform this MDP into an equivalent weakly coupled one. This is done by extending the feasible state-space $\bar{X}$ of the original MDP to X by including artificial states $X \backslash \bar{X}$. In each $x^i \in X^i$, we allow a $T+2$-dimensional artificial action vector $\mu^i(x^i)$ whose components are given by $(0; \mu_0^i(x^i); \mu_1^i(x^i); \ldots; 0) = \left(0; -\sum_{l=1}^{T} x_{l,0}^i; -\sum_{l=1}^{T-1} x_{l,1}^i; \ldots; -x_{1,T-1}^i; 0\right)$. This particular choice of artificial actions allows us to conveniently extend the set $\bar{U}(x)$ for $x \in \bar{X}$ to the set $\bar{A}(x)$ given below for $x \in X$. We first define the set $A^i(x^i) = U^i(x^i) \bigcup \mu^i(x^i)$, and for states $x \in X$, we let $A(x) = A^1(x^1) \times A^2(x^2) \times \ldots \times A^I(x^I)$. We then define the set of decisions feasible in state $x \in X$ as

$$\begin{aligned} \bar{A}(x) = \Bigg\{ & (u^1, u^2, \ldots, u^I) \in A(x) : \sum_{i=1}^{I} a_{ij} \Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i \Big) \leqslant b_j, \ j \in \mathcal{J}, \ 0 \leqslant k \leqslant T-1, \\ & \sum_{i=1}^{I} a_{ij} u_T^i \leqslant b_j, \text{ and } \sum_{i=1}^{I} a_{ij} \Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i \Big) \leqslant b'_j, \ j \in \mathcal{J} \Bigg\}. \end{aligned} \tag{1.7}$$

The artificial actions are defined in a way such that $\mu_k^i(x^i) + \sum_{l=1}^{T-k} x_{l,k}^i = 0$ for $0 \leqslant k \leqslant T-1$ and also the first and the last components of the artificial vector are both zero. This implies that (1.7) does not put any unnecessary restrictions on components of u that are not artificial. After choosing a decision $u \in A(x)$ whose ith component is the artificial decision $\mu^i(x^i)$ in a state with ith component $x^i \in X^i$, we artificially set the ith component of the next

state to $x'^i = (n_i; 0; \ldots; 0)$ with probability $p_i(n_i)$. Finally, we set the ith component of the immediate expected profit earned on choosing the artificial action component $\mu^i(x^i)$ in state component x^i to a sufficiently small number B. We emphasize here that our algorithm does not need a specific value for this number as we will explain in detail later in the context of problem (LLPR) toward the end of Section 1.5.1.

Now let $\Phi(x)$ be the maximum infinite-horizon discounted expected profit earned when starting the transformed MDP in state $x \in X$. Bellman's equations for the transformed MDP are given, for $x \in X$, by

$$\Phi(x) = \max_{u \in \bar{A}(x)} \left\{ f(x,u) + \alpha \sum_{x' \in X} \left(\prod_{i=1}^{I} p_i(n_i) \tilde{\Psi}^i(x^i, x'^i, u^i) \right) \Phi(x') \right\}. \tag{1.8}$$

Here, we have used the notation

$$\tilde{\Psi}^i(x^i, x'^i, u^i) = \begin{cases} \Psi^i(x^i, x'^i), \text{ if } u^i \text{ is not artificial;} \\ 1, \text{ if } u^i \text{ is artificial and } x'^i = (n_i; 0; \ldots; 0); \\ 0, \text{ otherwise,} \end{cases} \tag{1.9}$$

so that we could write the transition probabilities for both the original and the artificial actions in a compact manner on the right hand side of these Bellman's equations. Note that when u^i is an artificial action, we have forced $\tilde{\Psi}^i(x^i, x'^i, u^i)$ to equal 1 if the next state is $(n_i; 0; \ldots; 0)$ and to equal 0 if the next state is not $(n_i; 0; \ldots; 0)$. This is consistent with the fact that an artificial action deterministically forces the next state to be $(n_i; 0; \ldots; 0)$ as explained in the paragraph following Equation (1.7). The new MDP is weakly coupled. By using an argument identical to that in Gocgun and Ghate [41], we can show that $W(x) = \Phi(x)$ for every $x \in \bar{X}$. So we can instead consider this new MDP without loss of optimality in the original MDP. In fact, we will show later in Section 1.5.1 that artificial actions can be ignored in our algorithmic procedure.

Note, however, that both the original MDP and the transformed weakly coupled MDP

have state- and action-spaces whose sizes are exponential in I and in T. Therefore, exact solution of these MDPs via standard algorithms such as value iteration or policy iteration (see Puterman [91]) is computationally difficult. Similarly, the number of constraints and variables in a standard linear programming formulation (see Section 6.9.1 of Puterman [91]) would be exponential in I and T. This renders exact solution via the linear programming approach computationally intractable (also see Adelman and Mersereau [3] for a discussion of this issue). In the next section, we show that an extension, proposed by Gocgun and Ghate [41], of the Lagrangian relaxation method in Adelman and Mersereau [3] can be applied to approximately solve (1.8).

1.5 Lagrangian relaxation for approximate solution

The idea in the Lagrangian relaxation approach for approximate solution of weakly coupled MDPs is as follows. The constraints that link different components of states and actions are relaxed by attaching one Lagrange multiplier with each constraint. The resulting Lagrangian penalty function is added to the original immediate expected reward function. Then the linear program (LP) that is equivalent to the Bellman's equations of this relaxed problem is easier to solve as compared to the LP that corresponds to the Bellman's equations of the original problem (see Puterman [91] Section 6.9.1). In particular, Lagrangian relaxation achieves a problem decomposition so that the number of variables and the number of constraints in the LP for the relaxed problem are exponentially smaller than the original problem. It turns out, however, that this standard implementation of Lagrangian relaxation as proposed in Adelman and Mersereau [3], is not adequate to overcome the curse-of-dimensionality in our MDP. We explain this challenge in the next section.

1.5.1 Computational difficulty in a standard implementation

To apply the Lagrangian relaxation approach in Adelman and Mersereau [3] to our MDP, we first recall that there are two types of linking constraints that define the set of feasible

decisions (1.7) in this MDP. The first group of constraints,

$$\sum_{i=1}^{I} a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big) \leqslant b_j,\ j \in \mathcal{J},\ 0 \leqslant k \leqslant T-1,\ \text{and}$$
$$\sum_{i=1}^{I} a_{ij} u_T^i \leqslant b_j,$$

is about the total resource capacities (including overtime) $0 \leqslant k \leqslant T$ days from now. We attach Lagrange multipliers $\eta_k = (\eta_{k,1}, \eta_{k,2}, \ldots, \eta_{kJ}) \geqslant 0$, for $0 \leqslant k \leqslant T$, with these constraints. The second group of constraints,

$$\sum_{i=1}^{I} a_{ij}\Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i\Big) \leqslant b'_j,\ j \in \mathcal{J},$$

is about the regular resource capacities for today. We associate multipliers $\xi = (\xi_1, \xi_2, \ldots, \xi_J) \geqslant 0$ with these constraints. For brevity, we let $\lambda \geqslant 0$ represent the vector $(\xi, \eta_0, \eta_1, \ldots, \eta_T)$ of all Lagrange multipliers. The Lagrangian penalty functions for these linking constraints are then given by

$$\sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}\Big(b_j - \sum_{i=1}^{I} a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big)\Big),$$
$$\sum_{j=1}^{J} \eta_{T,j}\Big(b_j - \sum_{i=1}^{I} a_{ij} u_T^i\Big),\ \text{and}$$
$$\sum_{j=1}^{J} \xi_j\Big(b'_j - \sum_{i=1}^{I} a_{ij}\Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i\Big)\Big).$$

By adding these penalty functions to the immediate expected reward $\sum_{i=1}^{I} f_i(x^i, u^i)$ in (1.8), we obtain the Bellman's equations for the Lagrangian relaxation as

$$\Phi^\lambda(x) = \max_{u \in A(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) + \sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}\Big(b_j - \sum_{i=1}^{I} a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big)\Big) + \sum_{j=1}^{J} \eta_{T,j}\Big(b_j - \sum_{i=1}^{I} a_{ij} u_T^i\Big) \right.$$

$$+\sum_{j=1}^{J}\xi_j\Big(b'_j-\sum_{i=1}^{I}a_{ij}\big(\sum_{l=1}^{T}x^i_{l0}+u^i_0-v^i\big)\Big)+\alpha E_{(x,u)}\Phi^\lambda(\mathbf{X}')\Bigg\}. \tag{1.10}$$

Here, we have used $\mathbf{X}'$ to denote the next stochastic state as described in Equation (1.3) earlier, and $E_{(x,u)}$ denotes the expected value operator given that decision $u \in A(x)$ was chosen in state $x \in X$. According to the general theory of weakly coupled MDPs in Adelmand and Mersereau [3], the Lagrangian value function $\Phi^\lambda(x)$ can be expressed as

$$\Phi^\lambda(x)=\frac{1}{1-\alpha}\left(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}b_j+\sum_{j=1}^{J}\eta_{T,j}b_j+\sum_{j=1}^{J}\xi_j b'_j\right)+\sum_{i=1}^{I}\Phi^\lambda_i(x^i),\ \forall x\in X, \tag{1.11}$$

where

$$\begin{aligned}\Phi^\lambda_i(x^i)=\max_{u^i\in A^i(x^i)}\Bigg\{&f_i(x^i,u^i)-\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k}x^i_{lk}+u^i_k\Big)-\sum_{j=1}^{J}\eta_{T,j}a_{ij}u^i_T \\ &-\sum_{j=1}^{J}\xi_j a_{ij}\Big(\sum_{l=1}^{T}x^i_{l0}+u^i_0-v^i\Big)+\alpha E_{(x^i,u^i)}\Phi^\lambda_i(\mathbf{X}'^i)\Bigg\},\ \forall x^i\in X^i.\end{aligned} \tag{1.12}$$

Here, again, we have used $\mathbf{X}'^i$ to denote the ith component of the next stochastic state $\mathbf{X}$; the operator $E_{(x^i,u^i)}$ denotes the expected value given that decision $u^i \in A^i(x^i)$ was chosen in state $x^i \in X^i$.

It is shown in the general theory of Lagrangian relaxations of weakly coupled MDPs in Adelman and Mersereau [3] that, for any $\lambda \geqslant 0$, such a relaxation yields the bound $\Phi^\lambda(x) \geqslant \Phi(x)$ for each $x \in X$. Moreover, it is standard to work with weighted value functions. Toward this end, let $\beta(x) > 0$ be a sequence of positive numbers indexed by states $x \in X$ such that $\sum_{x\in X}\beta(x) = 1$. We then define $H^\lambda(\beta) = \sum_{x\in X}\beta(x)\Phi^\lambda(x)$ and $H(\beta) = \sum_{x\in X}\beta(x)\Phi(x)$. For any $\lambda \geqslant 0$, we have, $H^\lambda(\beta) \geqslant H(\beta)$. For each $i \in \mathcal{I}$, let $\beta_i(x^i) > 0$ be positive numbers indexed by $x^i \in X^i$ such that $\beta_i(x^i) = \sum_{\{x':x'^i=x^i\}}\beta(x')$. This ensures that

$$\sum_{x^i\in X^i}\beta_i(x^i)=\sum_{x^i\in X^i}\sum_{x':x'^i=x^i}\beta(x')=\sum_{x'\in X}\beta(x')=1.$$

Consequently, it is possible to interpret $\beta_i(x^i)$ as the probability that the initial state of component i is $x^i \in X^i$.

Let $\mathcal{Y}_i = \{(x^i, u^i) : x^i \in X^i, u^i \in A^i(x^i)\}$ for $i \in \mathcal{I}$. Then, for any fixed $\lambda \geqslant 0$, and for each fixed $i \in \mathcal{I}$, the values $\Phi_i^\lambda(x^i)$ for all $x^i \in X^i$, equal the optimal values of variables $\nu_i(x^i)$ in the LP

$$\begin{aligned}
&\min_{\nu_i(\cdot)} \sum_{x^i \in X^i} \beta_i(x^i)\nu_i(x^i) \\
&\quad \nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big) + \sum_{j=1}^{J} \eta_{T,j}a_{ij}u_T^i + \\
&\quad \sum_{j=1}^{J} \xi_j a_{ij}\Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i\Big) \geqslant f_i(x^i, u^i),\ (x^i, u^i) \in \mathcal{Y}_i.
\end{aligned}$$

Then, combining such LPs over all $i \in \mathcal{I}$, we obtain

$$\begin{aligned}
H^\lambda(\beta) = \frac{1}{1-\alpha}\Big(\sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}b_j + \sum_{j=1}^{J} \eta_{T,j}b_j + \sum_{j=1}^{J} \xi_j b'_j\Big) + \min_{\nu_i(\cdot),\ i\in\mathcal{I}} \sum_{i\in\mathcal{I}}\sum_{x^i\in X^i} \beta_i(x^i)\nu_i(x^i) \\
\nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big) + \sum_{j=1}^{J} \eta_{T,j}a_{ij}u_T^i + \\
\sum_{j=1}^{J} \xi_j a_{ij}\Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i\Big) \geqslant f_i(x^i, u^i),\ (x^i, u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I}.
\end{aligned}$$

Finally, to find the tightest upper bound on $H(\beta)$, we perform an outer minimization over all $\lambda \geqslant 0$ to obtain the Lagrangian LP

$$\begin{aligned}
(LLP)\ H^{\lambda*}(\beta) = \min_{\nu_i(\cdot),\ i\in\mathcal{I}}\ \frac{1}{1-\alpha}\Big(\sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}b_j + \sum_{j=1}^{J} \eta_{T,j}b_j + \sum_{j=1}^{J} \xi_j b'_j\Big) + \sum_{i=1}^{I}\sum_{x^i\in X^i} \beta_i(x^i)\nu_i(x^i) \\
\nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{k=0}^{T-1}\sum_{j=1}^{J} \eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k} x_{lk}^i + u_k^i\Big) + \sum_{j=1}^{J} \eta_{T,j}a_{ij}u_T^i + \\
\sum_{j=1}^{J} \xi_j a_{ij}\Big(\sum_{l=1}^{T} x_{l0}^i + u_0^i - v^i\Big) \geqslant f_i(x^i, u^i),\ (x^i, u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I},
\end{aligned}$$

$$\xi_j \geqslant 0,\ j \in \mathcal{J},$$
$$\eta_{kj} \geqslant 0,\ 0 \leqslant k \leqslant T,\ j \in \mathcal{J}.$$

We emphasize that the decision variables in this LP are ξ_j for $j \in \mathcal{J}$; η_{kj}, for $0 \leqslant k \leqslant T$ and $j \in \mathcal{J}$; and $\nu^i(x^i)$ for all $i \in \mathcal{I}$ and $x^i \in X^i$. Optimal values of these variables can be substituted back into (1.12) and (1.11) to obtain an approximation $\Phi^{\lambda^*}(x)$ for any $x \in X$ as needed.

Problem (LLP) has $(T+2)J + \sum_{i=1}^{I}(N_i+1)(M_i+1)^{\frac{T(T+1)}{2}}$ variables and at most $(T+2)J + \sum_{i=1}^{I}(N_i+1)(M_i+1)^{\frac{T(T+1)}{2}}(1+\binom{N_i+T+1}{T+1}(M_i+N_i+1))$ constraints. Thus, the numbers of variables and constraints are linear in I but unfortunately they are still exponential in T. Thus (LLP) is computationally difficult for realistic values of booking horizons T such as a couple of weeks. In Section 1.5.2, we therefore apply the affine value function approximation and constraint generation approach that was proposed in Gocgun and Ghate [41] to tackle such large-scale weakly coupled MDPs.

But first, we simplify the Lagrangian LP by dropping the functional constraints that correspond to artificial actions. For each $i \in \mathcal{I}$, let $\bar{\mathcal{Y}}_i \subset \mathcal{Y}_i$ be the subset given by $\{(x^i,u^i) \in \mathcal{Y}_i : u^i \neq \mu^i(x^i)\}$. That is, $\bar{\mathcal{Y}}_i$ does not include artificial actions and hence is equivalently defined as $\bar{\mathcal{Y}}_i = \{(x^i,u^i) : x^i \in X^i,\ u^i \in U^i(x^i)\}$. Now consider a relaxation of (LLP), which only includes functional constraints over sets $\bar{\mathcal{Y}}_i$ for $i \in \mathcal{I}$. It is given by

$$(LLPR)\ \min \frac{1}{1-\alpha}\Big(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}b_j + \sum_{j=1}^{J}\eta_{T,j}b_j + \sum_{j=1}^{J}\xi_j b'_j\Big) + \sum_{i=1}^{I}\sum_{x^i \in X^i}\beta_i(x^i)\nu_i(x^i)$$
$$\nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k}x^i_{lk} + u^i_k\Big) + \sum_{j=1}^{J}\eta_{T,j}a_{ij}u^i_T +$$
$$\sum_{j=1}^{J}\xi_j a_{ij}\Big(\sum_{l=1}^{T}x^i_{l0} + u^i_0 - v^i\Big) \geqslant f_i(x^i,u^i),\ (x^i,u^i) \in \bar{\mathcal{Y}}_i,\ i \in \mathcal{I},$$
$$\xi_j \geqslant 0,\ j \in \mathcal{J},$$
$$\eta_{kj} \geqslant 0,\ 0 \leqslant k \leqslant T,\ j \in \mathcal{J}.$$

Suppose $\nu_i^*(x^i)$ for $x^i \in X^i$ and $i \in \mathcal{I}$ is an optimal solution of (LLPR). We claim that this solution is also optimal to (LLP). To establish this, we only need to show that this solution satisfies the functional constraints in (LLP) that correspond to $\bigcup_{i\in\mathcal{I}} \left(\mathcal{Y}_i \backslash \bar{\mathcal{Y}}_i\right)$. The left hand side of such a functional constraint corresponding to $x^i \in X^i$ and artificial action $\mu^i(x^i)$ simply equals $\nu_i^*(x^i) - \alpha E_{(x^i,\mu^i(x^i))}\nu_i^*(\mathbf{X}'^i)$. Let $\nu^* = \min_{i\in\mathcal{I}} \left\{ \min_{x^i\in X^i} \left(\nu_i^*(x^i) - \alpha E_{(x^i,\mu^i(x^i))}\nu_i^*(\mathbf{X}'^i)\right)\right\}$ be the smallest among all such left hand side values. Then our claim holds because $f_i(x^i, \mu^i(x^i))$, which equals B, is sufficiently small, and in particular, we assume it to be smaller than ν^*, for all $i \in \mathcal{I}$ and $x^i \in X^i$. In view of this result, we now attempt to solve (LLPR) approximately instead of solving (LLP) approximately.

1.5.2 Affine value function approximation and constraint generation

Following Gocgun and Ghate [41], we simplify (LLPR) using the affine approximation $\nu_i(x^i) \approx \gamma_i + d_i x_0^i + \sum_{k=0}^{T-1}\sum_{l=1}^{T-k} c_{lk}^i x_{lk}^i$, where γ_i, d_i and c_{lk}^i are unknown coefficients. Coefficients d_i and c_{lk}^i can be interpreted as the marginal profit from newly arriving requests and the marginal profit for jobs in the booking horizon respectively. This simplifies (LLPR) to the following approximate Lagrangian LP

$$(ALLP)\ H = \min_{\xi,\eta,c,d}\ \frac{1}{1-\alpha}\left(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}b_j + \sum_{j=1}^{J}\eta_{T,j}b_j + \sum_{j=1}^{J}\xi_j b'_j\right) + \sum_{i=1}^{I}\left(\gamma_i + \Big(\sum_{x^i\in X^i}\beta_i(x^i)x_0^i\Big)d_i\right) + \sum_{i=1}^{I}\sum_{k=0}^{T-1}\sum_{l=1}^{T-k}\Big(\sum_{x^i\in X^i}\beta_i(x^i)x_{lk}^i\Big)c_{lk}^i \tag{1.13}$$

$$(1-\alpha)\gamma_i + (x_0^i - \alpha E\mathbf{\Delta}_i)d_i + \sum_{k=0}^{T-1}\sum_{l=1}^{T-k}\left(x_{lk}^i - \alpha E_{(x^i,u^i)}\mathbf{X}'^i_{lk}\right)c_{lk}^i + \sum_{j=1}^{J}\eta_{T,j}a_{ij}u_T^i + \sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{kj}a_{ij}\Big(\sum_{l=1}^{T-k}x_{lk}^i + u_k^i\Big) + \sum_{j=1}^{J}\xi_j a_{ij}\Big(\sum_{l=1}^{T}x_{l0}^i + u_0^i - v^i\Big) \geqslant f_i(x^i,u^i),\ (x^i,u^i)\in\mathcal{Y}_i,\ i\in\mathcal{I}, \tag{1.14}$$

$$\xi_j \geqslant 0,\ j\in\mathcal{J}, \tag{1.15}$$

$$\eta_{kj} \geqslant 0,\ 0\leqslant k\leqslant T,\ j\in\mathcal{J}, \tag{1.16}$$

$$d_i \geqslant 0,\ i \in \mathcal{I}, \tag{1.17}$$

$$c^i_{lk} \geqslant 0,\ 0 \leqslant k \leqslant T-1; 1 \leqslant k+l \leqslant T;\ i \in \mathcal{I}. \tag{1.18}$$

Table 1.2 summarizes the variables and their interpretations in this problem.

Variable	description/interpretation
γ_i	affine term in the value function approximation
d_i	marginal profit for newly arriving requests in the value function approximation
c^i_{lk}	marginal profit for jobs in the booking horizon in the value function approximation
η_{kj}	Lagrange multipliers for resource constraints including overbooking
ξ_j	Lagrange multipliers for resource constraints excluding overbooking

Table 1.2: A summary of variables in problem ALLP.

The number of variables is linear in I and in J, and it is quadratic in T. However, the number of constraints is still exponential in T although it is linear in I. We therefore use constraint generation, as in Gocgun and Ghate [41], to solve (ALLP). The constraint generation approach involves two main steps. We first need to find an initial set of constraints such that the resulting relaxation of (ALLP) has an optimal solution. Then, we want to find a constraint in (ALLP) that is violated by this optimal solution. We then want to add this constraint and solve the resulting new relaxation of (ALLP) to find its optimal solution. This process is repeated until a stopping criterion is met. We describe the details of these steps in the following.

1.5.2.1 Finding an initial set of constraints

We initiate the constraint generation procedure for (ALLP) with an initial set of constraints $\mathcal{C}_0$ that includes constraints corresponding to $x^i_0 = 0$; $x^i_{lk} = 0$ for $1 \leqslant l \leqslant T$ and $1 \leqslant k+l \leqslant T$; $u^i_k = 0$ for $0 \leqslant k \leqslant T$; and $v^i = 0$, for all $i \in \mathcal{I}$. In this initial set of constraints the following terms will be zero: $f_i(x^i, u^i)$, $\sum_{j=1}^{J} \eta_{T,j} a_{ij} u^i_T$, $\sum_{k=0}^{T-1} \sum_{j=1}^{J} \eta_{kj} a_{ij} \Big(\sum_{l=1}^{T-k} x^i_{lk} + u^i_k \Big)$ and

$\sum_{j=1}^{J} \xi_j a_{ij} \Big(\sum_{l=1}^{T} x^i_{l0} + u^i_0 - v^i \Big)$. Thus, (ALLP) reduces to

$$
\begin{aligned}
(ALLP)_{\mathcal{C}_0} \ \min_{\xi,\eta,c,d} \ & \frac{1}{1-\alpha}\left(\sum_{k=0}^{T-1} \sum_{j=1}^{J} \eta_{kj} b_j + \sum_{j=1}^{J} \eta_{T,j} b_j + \sum_{j=1}^{J} \xi_j b'_j \right) + \\
& \sum_{i=1}^{I} \Big(\gamma_i + \Big(\sum_{x^i \in X^i} \beta_i(x^i) x^i_0 \Big) d_i \Big) + \sum_{i=1}^{I} \sum_{k=0}^{T-1} \sum_{l=1}^{T-k} \Big(\sum_{x^i \in X^i} \beta_i(x^i) x^i_{lk} \Big) c^i_{lk} \\
(1-\alpha)\gamma_i & \geqslant \alpha E \mathbf{\Delta}_i d_i, \ i \in \mathcal{I}, \\
\lambda & \geqslant 0, \\
d_i & \geqslant 0, \ i \in \mathcal{I}, \\
c^i_{lk} & \geqslant 0, \ 0 \leqslant k \leqslant T-1; 1 \leqslant k+l \leqslant T; \ i \in \mathcal{I}.
\end{aligned}
$$

The Lagrange multipliers and all c^i_{lk} variables are non-negative and they only appear in the objective function with positive coefficients. So their optimal values in $(ALLP)_{\mathcal{C}_0}$ are zero. So note that the only remaining term in the objective function is $\sum_{i=1}^{I} \Big(\gamma_i + \Big(\sum_{x^i \in X^i} \beta_i(x^i) x^i_0 \Big) d_i \Big)$. It is also easy to see that the optimal values of d_i and γ_i in this problem are zero because d_i is non-negative. In particular, $(ALLP)_{\mathcal{C}_0}$ has an optimal solution and we use this solution to initiate our constraint generation iterations.

1.5.2.2 Finding the most violated constraint

Now suppose that, during the constraint generation procedure, after solving $(\text{ALLP})_{\mathcal{C}}$ with some set of constraints $\mathcal{C}$, we obtain $\gamma^{\mathcal{C}}, c^{\mathcal{C}}, \lambda^{\mathcal{C}}, d^{\mathcal{C}}$ as its optimal solution with the corresponding optimal value $H_{\mathcal{C}}$. We define $\bar{\mathcal{Y}}_i \triangleq \{(x^i, u^i) : x^i \in X^i, u^i \in U^i(x^i)\}$ as the subset of $\mathcal{Y}_i$ that excludes artificial actions. The violation in the functional constraint corresponding to any $(x^i, u^i) \in \bar{\mathcal{Y}}_i$ for any $i \in \mathcal{I}$ is given by

$$
\mathcal{Z}_i(x^i, u^i) \triangleq f_i(x^i, u^i) - (1-\alpha)\gamma^{\mathcal{C}}_i - (x^i_0 - \alpha E \mathbf{\Delta}_i) d^{\mathcal{C}}_i - \sum_{l=1}^{T} \sum_{k=0}^{T-l} (x^i_{lk} - \alpha E_{(x^i,u^i)} \mathbf{X}'^i_{lk}) c^{i\mathcal{C}}_{lk}
$$

$$- \sum_{k=0}^{T-1} \sum_{j=1}^{J} \eta^{\mathcal{C}}_{k,j} a_{ij} \Big(\sum_{l=1}^{T-k} x^i_{lk} + u^i_k \Big) - \sum_{j=1}^{J} \eta^{\mathcal{C}}_{T,j} a_{ij} u^i_T - \sum_{j=1}^{J} \xi^{\mathcal{C}}_j a_{ij} \Big(\sum_{l=1}^{T} x^i_{l0} + u^i_0 - v^i \Big).$$

To simplify this, recall that $\mathbf{X'}^i_{1,k} = u^i_{k+1}$ for $0 \leqslant k \leqslant T-1$, and $\mathbf{X'}^i_{lk} = x^i_{l-1,k+1} - y^i_{l-1,k+1}$ for $2 \leqslant l \leqslant T$, and $2 \leqslant l + k \leqslant T$. Also recall that $y^i_{l,k}$ is a binomial random variable with parameters $x^i_{l,k}$ and $q^i_{l,k}$; thus, its expected values is $x^i_{l,k} q^i_{l,k}$. Consequently,

$$\sum_{l=1}^{T} \sum_{k=0}^{T-l} c^{i\mathcal{C}}_{lk} E_{(x^i,u^i)} \mathbf{X'}^i_{lk} = \sum_{k=0}^{T-1} c^{i\mathcal{C}}_{1,k} u^i_{k+1} + \sum_{l=2}^{T} \sum_{k=0}^{T-l} c^{i\mathcal{C}}_{lk} x^i_{l-1,k+1} (1 - q^i_{l-1,k+1}).$$

Recalling that $f_i(x^i, u^i) = R_i \Big(u^i_0 + \sum_{l=1}^{T} x^i_{l,0} \theta^i_l \Big) - G_i \Big(x^i_0 - \sum_{k=0}^{T} u^i_k \Big) - \sum_{k=0}^{T} u^i_k C_i(k) - \alpha h_i v^i$, we get

$$\begin{aligned}
\mathcal{Z}_i(x^i, u^i) = {} & R_i \Big(u^i_0 + \sum_{l=1}^{T} x^i_{l,0} \theta^i_l \Big) - G_i \Big(x^i_0 - \sum_{k=0}^{T} u^i_k \Big) - \sum_{k=0}^{T} u^i_k C_i(k) - \alpha h_i v^i - (1-\alpha) \gamma^{\mathcal{C}}_i \\
& - d^{\mathcal{C}}_i (x^i_0 - \alpha E \boldsymbol{\Delta}_i) - \sum_{k=0}^{T-1} (x^i_{1,k} - \alpha u^i_{k+1}) c^{i\mathcal{C}}_{1,k} - \sum_{l=2}^{T} \sum_{k=0}^{T-l} \big(x^i_{lk} - \alpha x^i_{l-1,k+1} (1 - q^i_{l-1,k+1}) \big) c^{i\mathcal{C}}_{lk} \\
& - \sum_{k=0}^{T-1} \sum_{j=1}^{J} \eta^{\mathcal{C}}_{kj} a_{ij} \Big(\sum_{l=1}^{T-k} x^i_{lk} + u^i_k \Big) - \sum_{j=1}^{J} \eta^{\mathcal{C}}_{T,j} a_{ij} u^i_T - \sum_{j=1}^{J} \xi^{\mathcal{C}}_j a_{ij} \Big(\sum_{l=1}^{T} x^i_{l0} + u^i_0 - v^i \Big).
\end{aligned}$$

After algebraic simplification, the above is seen to be of the form $\mathcal{Z}_i(x^i, u^i) = \delta^{i\mathcal{C}}_0 x^i_0 + \sum_{l=1}^{T} \delta^{i\mathcal{C}}_{l,0} x^i_{l,0} + \sum_{k=1}^{T-1} \sum_{l=1}^{T-k} \delta^{i\mathcal{C}}_{lk} x^i_{lk} + \epsilon^{i\mathcal{C}}_0 u^i_0 + \sum_{k=1}^{T} \epsilon^{i\mathcal{C}}_k u^i_k + \delta^{i\mathcal{C}} v^i + \epsilon^{i\mathcal{C}}$, where

$$\begin{aligned}
\delta^{i\mathcal{C}}_0 &\triangleq -G_i - d^{\mathcal{C}}_i, \\
\delta^{i\mathcal{C}}_{l,0} &\triangleq R_i \theta^i_l - c^{i\mathcal{C}}_{l,0} - \sum_{j=1}^{J} \xi^{\mathcal{C}}_j a_{ij} - \sum_{j=1}^{J} \eta^{\mathcal{C}}_{0,j} a_{ij}, \text{ for } l = 1, \ldots, T, \\
\delta^{i\mathcal{C}}_{lk} &\triangleq -c^{i\mathcal{C}}_{lk} + \alpha (1 - q^i_{lk}) c^{i\mathcal{C}}_{l+1,k-1} - \sum_{j=1}^{J} \eta^{\mathcal{C}}_{kj} a_{ij}, \text{ for } k = 1, \ldots, T-1; l = 1, \ldots, T-k, \\
\epsilon^{i\mathcal{C}}_0 &\triangleq R_i + G_i - C_i(0; D_i) - \sum_{j=1}^{J} \xi^{\mathcal{C}}_j a_{ij} - \sum_{j=1}^{J} \eta^{\mathcal{C}}_{0,j} a_{ij},
\end{aligned}$$

$$\epsilon_k^{i\mathcal{C}} \triangleq G_i - C_i(k) + \alpha c_{1,k-1}^{i\mathcal{C}} - \sum_{j=1}^{J} \eta_{kj}^{\mathcal{C}} a_{ij}, \text{ for } k = 1, \ldots, T,$$

$$\delta^{i\mathcal{C}} \triangleq \sum_{j=1}^{J} \xi_j^{\mathcal{C}} a_{ij} - \alpha h_i, \text{ and}$$

$$\epsilon^{i\mathcal{C}} \triangleq -(1-\alpha)\gamma_i^{\mathcal{C}} + \alpha d_i^{\mathcal{C}} E \boldsymbol{\Delta}_i.$$

Thus the maximum constraint violation problem is given by $\mathcal{Z}^* \triangleq \max_{i \in \mathcal{I}} \mathcal{Z}_i^*$, where $\mathcal{Z}_i^*$ is the optimal value of the linear integer program

$$\mathcal{Z}_i^* = \max \ \delta_0^{i\mathcal{C}} x_0^i + \sum_{l=1}^{T} \delta_{l,0}^{i\mathcal{C}} x_{l,0}^i + \sum_{k=1}^{T-1} \sum_{l=1}^{T-k} \delta_{lk}^{i\mathcal{C}} x_{lk}^i + \epsilon_0^{i\mathcal{C}} u_0^i + \sum_{k=1}^{T} \epsilon_k^{i\mathcal{C}} u_k^i + \delta^{i\mathcal{C}} v^i + \epsilon^{i\mathcal{C}} \tag{1.19}$$

$$\begin{aligned}
x_0^i &\leqslant N_i, \\
\sum_{t=0}^{T} u_t^i &\leqslant x_0^i, \\
\sum_{l=1}^{T-k} x_{lk}^i + u_k^i &\leqslant M_i, \text{ for } 0 \leqslant k \leqslant T-1, \\
v^i &\leqslant u_0^i + \sum_{l=1}^{T} x_{l0}^i, \\
u_T^i &\leqslant M_i, \\
x_{lk}^i &\geqslant 0, \text{ and integer, for } 0 \leqslant k \leqslant T-1; 1 \leqslant l \leqslant T-k, \\
u_k^i &\geqslant 0, \text{ and integer, for } 0 \leqslant k \leqslant T, \\
v^i &\geqslant 0, \text{ and integer.}
\end{aligned}$$

This linear integer program includes $T+4$ constraints and the number of variables is quadratic in T. It could, therefore, be computationally difficult to solve when T is large; but we believe that in practical applications, where T is likely to be of the order of a month, it should be solvable relatively easily with modern solvers. Note that if $\mathcal{Z}^* \leqslant 0$ then $\gamma^{\mathcal{C}}, c^{\mathcal{C}}, \lambda^{\mathcal{C}}, d^{\mathcal{C}}$

satisfy all constraints in (ALLP). If, on the other hand, $\mathcal{Z}^* > 0$, then the optimal values of states and actions in this problem tell us which constraints to add to the existing set of constraints $\mathcal{C}$ before resolving the resulting new approximation of (ALLP). This process continues until a stopping criterion is met. Proposition 3.1 in Gocgun and Ghate [41] shows that $H_{\mathcal{C}} \leqslant H \leqslant H_{\mathcal{C}} + \frac{I\mathcal{Z}^*}{1-\alpha}$. This leads to a natural stopping criterion, which is to stop when the absolute percentage gap $\frac{I}{(1-\alpha)}\left|\frac{\mathcal{Z}^*}{H_{\mathcal{C}}}\right|$ between the lower and the upper bounds on H drops below a tolerance.

1.5.3 Retrieving decisions from the approximate value function

Let λ^*; γ_i^* and d_i^* for $i \in \mathcal{I}$; and c_{lk}^{i*} for $0 \leqslant k \leqslant T-1$, $1 \leqslant k+l \leqslant T$, and $i \in \mathcal{I}$ denote the values of variables in (ALLP) after terminating constraint generation. Now suppose that for some state $x \in \bar{X}$, we wish to compute a decision vector that is myopic with respect to the approximation

$$V(x') = \Phi(x') \approx \Phi^{\lambda^*}(x') = \frac{1}{1-\alpha}\Big(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{k,j}^* b_j + \sum_{j=1}^{J}\eta_{T,j}^* b_j + \sum_{j=1}^{J}\xi_j^* b'_j\Big) + \sum_{i=1}^{I}\Phi_i^{\lambda^*}(x'^i)$$
$$\approx \frac{1}{1-\alpha}\Big(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{k,j}^* b_j + \sum_{j=1}^{J}\eta_{T,j}^* b_j + \sum_{j=1}^{J}\xi_j^* b'_j\Big) + \sum_{i=1}^{I}\Big(\gamma_i^* + d_i^* x_0'^i + \sum_{k=0}^{T-1}\sum_{l=1}^{T-k} c_{lk}^{i*} x_{lk}'^i\Big).$$

After substituting this in the right hand side of Bellman's equation (1.6), the decision retrieval problem simplifies to
$\frac{\alpha}{1-\alpha}(\sum_{k=0}^{T-1}\sum_{j=1}^{J}\eta_{k,j}^* b_j + \sum_{j=1}^{J}\eta_{T,j}^* b_j + \sum_{j=1}^{J}\xi_j^* b'_j) + \alpha\sum_{i=1}^{I}\gamma_i^* + \max_{u\in\bar{U}(x)}\Bigg\{\sum_{i=1}^{I} f_i(x^i,u^i) + \alpha\sum_{i=1}^{I}\Big(d_i^* E_{(x^i,u^i)}\mathbf{X}_0'^i + \sum_{k=0}^{T-1}\sum_{l=1}^{T-k} c_{lk}^{i*} E_{(x^i,u^i)}\mathbf{X}_{lk}'^i\Big)\Bigg\}.$
We ignore the terms outside the maximization in the discussion below. Then, after algebraic simplification, we get

$$\max_{u\in\bar{U}(x)}\Bigg\{\sum_{i=1}^{I}\Big(R_i(u_0^i + \sum_{l=1}^{T} x_{l,0}^i\theta_l^i) - G_i(x_0^i - \sum_{k=0}^{T} u_k^i) - \sum_{k=0}^{T} u_k^i C_i(k) - \alpha h_i v^i\Big) +$$

$$\alpha \sum_{i=1}^{I} \left(d_i^* E\boldsymbol{\Delta}_i + \sum_{k=0}^{T-1} c_{1,k}^{i*} u_{k+1}^i + \sum_{l=2}^{T} \sum_{k=0}^{T-l} c_{lk}^{i*} x_{l-1,k+1}^i (1 - q_{l-1,k+1}^i) \right) \Bigg\}.$$

Finally, ignoring the terms in the objective function that do not depend on u and using the definition of $\bar{U}(x)$ in (1.2), we further rewrite this problem as

$$\begin{aligned}
\max \ & \sum_{i=1}^{I} \left(R_i u_0^i + G_i \sum_{k=0}^{T} u_k^i - \sum_{k=0}^{T} u_k^i C_i(k) - \alpha h_i v^i \right) + \alpha \sum_{i=1}^{I} \left(\sum_{k=0}^{T-1} c_{1,k}^{i*} u_{k+1}^i \right) \\
& \sum_{k=0}^{T} u_k^i \leqslant x_0^i, \ i \in \mathcal{I}, \\
& v^i - u_0^i \leqslant \sum_{l=1}^{T} x_{l0}^i, \ i \in \mathcal{I}, \\
& \sum_{i=1}^{I} a_{ij} u_k^i \leqslant b_j - \sum_{i=1}^{I} a_{ij} \sum_{l=1}^{T-k} x_{lk}^i, \ j \in \mathcal{J}, \ 0 \leqslant k \leqslant T-1, \\
& \sum_{i=1}^{I} a_{ij} u_T^i \leqslant b_j, \ j \in \mathcal{J}, \\
& \sum_{i=1}^{I} a_{ij} (u_0^i - v^i) \leqslant b'_j - \sum_{i=1}^{I} a_{ij} \sum_{l=1}^{T} x_{l0}^i, \ j \in \mathcal{J} \\
& u_k^i \geqslant 0 \text{ and integer}, \ i \in \mathcal{I}, \ 0 \leqslant k \leqslant T, \\
& v^i \geqslant 0 \text{ and integer}, \ i \in \mathcal{I}.
\end{aligned}$$

The decision vector obtained by solving this problem is called the Lagrangian decision. The overall computational process is summarized in Algorithm 1.

In the next section, we apply this methodology to a large group of test problems.

1.6 Computational results

We compared the performance of Lagrangian decisions with that of myopic decisions. In any state $x \in \bar{X}$, a myopic decision was obtained by solving the problem $\max_{u \in \bar{U}(x)} f(x,u)$, which was again a linear integer program.

Algorithm 1 A constraint generation algorithm for approximately solving (ALLP)

1: Initialize the iteration counter $t = 0$ and choose a small tolerance $\epsilon > 0$.
2: Solve $(\text{ALLP})_{\mathcal{C}_t}$ and let $H_{\mathcal{C}_t}$ denote its optimal value.
3: Use the optimal values of the variables in $(\text{ALLP})_{\mathcal{C}_t}$ as listed in Table 1.2 to solve the maximum violated constraint problem (1.19) and obtain its optimal value $\mathcal{Z}_i^*$, for $i \in \mathcal{I}$.
4: Let $\mathcal{Z}^* = \max_{i \in \mathcal{I}} \mathcal{Z}_i^*$ and $i^* = \operatorname{argmax}_{i \in \mathcal{I}} \mathcal{Z}_i^*$. Save $(x^{i^*}, u^{i^*}) \in \mathcal{Y}_{i^*}$.
5: Stop if $\mathcal{Z}^* \leqslant 0$.
6: **if** $\frac{I\mathcal{Z}^*}{(1-\alpha)H_{\mathcal{C}_t}} < \epsilon$ **then** Stop
7: **else**
8: Add constraint (1.14) for (x^{i^*}, u^{i^*}) to the set $\mathcal{C}_t$ and update $t \leftarrow t + 1$.
9: **end if**
10: Report as final output the optimal values of the variables in $(\text{ALLP})_{\mathcal{C}_t}$.

Our numerical experiments were performed via computer simulations on 1800 problems that were randomly generated using an extension of the problem generation technique from Gocgun and Ghate [41]. All simulations were performed on a 3.1 GHz iMac desktop with 16 GB RAM and an Intel Core i7 chip running Mac OS X 10.9.3. All linear and integer programs were solved using CPLEX 12 from IBM via a MATLAB R2014a front-end.

The problem generation technique in Gocgun and Ghate [41] was in turn an extension of a standard method to create multi-dimensional knapsack problems [24] (note that advanced scheduling problems can be viewed as a version of dynamic-stochastic multi-dimensional knapsack problems). Since Gocgun and Ghate [41] did not incorporate no-shows, cancellations, and overbooking, we had to modify their problem generation approach to accommodate these features as explained next.

In the empirical literature in healthcare, the probability of patient attendance has been observed to depend on patient-type and also on the wait time [26, 47, 62]. Muthuraman and Lawley in [79] and Chakraborty et al. in [20] generated patient-type dependent no-shows probabilities. LaGanga and Lawrence in [65] and Nan Liu et al. in [72] also generated data for cancellation and no-show probabilities that are dependent on wait time. In our more general model, the cancellation probability q_{lk}^i is dependent on three factors: job-type i; how long

ago the appointment decision was made (l); and how far into the future the appointment is booked (k). Specifically, q^i_{lk} is given by $\text{unif}_i \times \frac{\kappa^i \times l + \sigma^i \times k}{T \times \sigma^i}$ where unif_i is a uniformly distributed random number between zero and one; this scaling by a random fraction was employed to avoid generating cancellation probabilities too close to one. The two coefficients κ^i and σ^i were set to $DU[1, 10]$ (uniformly distributed integer between 1 and 10). We forced $\sigma^i > \kappa^i$ because, intuitively, we wanted the dependence on k to be more pronounced than that on l. The probability that a type-i patient shows up, θ^i_l, was given by $1 - \text{unif}_i \frac{\pi^i \times l}{T \times \max\limits_{i \in \mathcal{I}} \pi^i}$, where the π^i values were sampled from $DU[1, 10]$. This functional form has the intuitive property that the no-show probability $1 - \theta^i_l$ is increasing in l. The rest of the parameters were generated in a manner identical to Gocgun and Ghate [41] as described in the next paragraph.

The number of job types was fixed at $I = 5$. The max arrival N_i was set to $DU[1, 20]$ and this N_i was used to obtain arrival probabilities $p_i(0), p_i(1), \ldots, p_i(N_i)$ as $N_i + 1$ uniform $(0, 1)$ random variables that are then normalized. The unit resource consumption $a_{i,j}$ was set to $DU[1, 100]$ and the regular resource availability b'_j equaled $I \times \rho \times \sum\limits_{i=1}^{I} a_{ij}$ for all j; here, ρ is called tightness ratio. The tightness ratio and number resources were set to $\rho = 0.25, 0.35, 0.45, 0.55, 0.65, 0.75$ and $J = 1, 2$, respectively. The total resource capacities including overtime were generated as $b = \lfloor (1 + 0.25\overrightarrow{\text{unif}}_i(J)) \cdot \times b' \rfloor$, where b' is the regular resource availability vector, and $\overrightarrow{\text{unif}}_i(J)$ is a J-dimensional vector that contains one uniform $(0, 1)$ random number for each resource. Here, we multiplied $\vec{\text{unif}}_i(J)$ by 0.25 so that the overtime resource does not exceed regular resource by more than 25% and $\cdot\times$ denotes componentwise product. We used $T = 5, 10, 15$. The delay cost was

$$C_i(k) = F_i \times \max(k - D_i, 0), \text{ for } 0 \leqslant k \leqslant T,$$

as in [41, 83] with $F_i = \text{DU}[1, 100]$; here D_i is interpreted as the "deadline" associated with type-i jobs, $0 \leqslant D_i \leqslant T$. The deadline was set to $D_i = DU[0, T]$. The rejection cost was set to $G_i = r_i C_i(T; D_i)$, where r_i was a uniform $(0, 2)$ random variable. The revenue R_i was set to $DU[1, 1000]$. The overtime cost h_i was set to $\text{unif}_i \times G_i$. The discount factor was 0.85.

We created a total of 36 advance scheduling problem sets by changing J, ρ, T values. For each problem set, we generated 50 problem instances.

The initial state distribution $\beta_i(x^i)$ was chosen to be uniform over X^i. Constraint generation was terminated after the absolute gap $\frac{I}{1-\alpha}\left|\frac{\mathcal{Z}^*}{H_C}\right|$ dropped below 0.5%. For each problem instance, we performed 200 independent simulations each with 50 periods. The Lagrangian and myopic methods were compared using the average profit over these 200 simulations. Each simulation started with an empty state and a Lagrangian and a myopic decision was obtained in each visited state using the approach described earlier. The next state was then sampled by appropriately sampling a random number of new arrivals, a random number of no-shows, and a random number of cancellations. A summary of our results is shown in Tables 1.3, 1.4 and 1.5 for $T = 5, 10, 15$, respectively. Each row in a table corresponds to one problem set, that is, one fixed combination of J and ρ. In each problem set, the number of times out of 50 that the difference between the two methods was statistically significant according to a paired t-test (at significance threshold 0.05) is listed in the third column. Out of the instances in which the difference was significant, the number of instances in which the Lagrangian method generated a higher profit than the myopic method is listed in the fourth column. The average improvement achieved by the Lagrangian method over the myopic method over these instances is listed in the fifth column. The average CPU time (in seconds) needed to approximately solve (ALLP) using Algorithm 1 is reported in the last column. We note that this is essentially the only additional CPU time needed for the Lagrangian method over and above what is needed for the myopic method. We emphasize that in practice this CPU time is spent only one time as a preprocessing step. This is because problem (ALLP) is solved only once to obtain the values of the variables listed in Table 1.2 in order to characterize our affine value function approximation. This approximation is then used during a real system-run to recover Lagrangian decisions only in the system-states visited as explained in Section 1.5.3.

J	ρ	# significant	# $L > M$	% improvement	average CPU time for ALLP (sec.)
1	0.25	39	39	9.55	0.78
1	0.35	38	37	6.11	2.64
1	0.45	44	43	4.25	3.85
1	0.55	48	46	3.58	4.01
1	0.65	45	43	2.50	4.73
1	0.75	44	38	2.34	6.17
2	0.25	24	24	13.43	2.83
2	0.35	30	30	6.53	1.34
2	0.45	35	35	5.93	1.97
2	0.55	37	35	4.37	2.79
2	0.65	43	42	3.62	3.56
2	0.75	42	41	1.70	4.17

Table 1.3: Results for 12 problem sets with $T = 5$. For each problem set, we generated 50 test instances. Thus the total number of problems summarized in this table is 600.

J	ρ	# significant	# $L > M$	% improvement	average CPU time for ALLP (sec.)
1	0.25	45	44	12.82	8.89
1	0.35	43	41	8.70	14.47
1	0.45	49	49	4.10	18.12
1	0.55	44	44	2.22	14.32
1	0.65	41	41	2.54	18.72
1	0.75	42	42	1.67	18.58
2	0.25	27	26	8.03	8.43
2	0.35	34	30	6.84	10.92
2	0.45	38	37	4.47	14.08
2	0.55	41	40	3.39	14.36
2	0.65	42	41	2.36	15.76
2	0.75	39	38	2.46	20.40

Table 1.4: Results for 12 problem sets with $T = 10$. For each problem set, we generated 50 test instances. Thus the total number of problems summarized in this table is 600.

J	ρ	# significant	# $L > M$	% improvement	average CPU time for ALLP (sec.)
1	0.25	36	36	8.17	52.48
1	0.35	37	37	5.38	85.44
1	0.45	42	42	3.33	92.52
1	0.55	39	39	2.76	147.10
1	0.65	37	37	1.99	138.69
1	0.75	39	37	1.67	213.25
2	0.25	30	29	8.53	88.68
2	0.35	38	38	10.63	125.82
2	0.45	34	33	3.72	164.71
2	0.55	47	44	3.84	94.91
2	0.65	43	43	2.56	101.41
2	0.75	45	43	1.98	126.12

Table 1.5: Results for 12 problem sets with $T = 15$. For each problem set, we generated 50 test instances. Thus the total number of problems summarized in this table is 600.

The three tables show that out of the 1800 problem instances listed, the difference between the two methods was statistically significant in 1391, that is, in 77% of the problem instances. Of these 1391 instances where the difference was significant, the Lagrangian method outperformed the myopic method in 1354, that is, in 97% of the instances. In these instances, the Lagrangian method outperformed the myopic method by about 5% on average. The tables also suggest that the improvement achieved by the Lagrangian method decreases as the resource capacities, that is, ρ values, increase — this is intuitive because when resources are plentiful, a simple heuristic is likely to perform as well as a more sophisticated approach.

The next chapter focuses on a dynamic resource allocation problem that arises in project scheduling.

Chapter 2

DYNAMIC RESOURCE-CONSTRAINED PROJECT SCHEDULING

The work reported here has appeared in the journal Operations Research Letters, 45(5), 442-447, 2017.

2.1 Introduction and literature review

Systematic mathematical studies on project scheduling date at least as far back as the 1950s to the Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT) [86]. Both CPM and PERT assume unlimited resource availability. Early work on resource-constrained project scheduling problems (RCPSPs) can perhaps be traced to the zero-one programming model in [90]. That paper studied a discrete-time problem and incorporated multiple projects with precedence-constrained tasks and multiple resources. Task durations were deterministic and no new projects arrived over time. The binary decision variables corresponded to whether or not a task is completed in a certain period. Such static-deterministic RCPSPs started receiving more attention in the 1980s. For instance, Blazewicz et al. [15] showed in 1983 that such RCPSPs are NP-hard. More recent literature surveys of static-deterministic RCPSPs are available in Herroelen et al. [52], and in Hartmann and Brikson [48]. Hartmann and Brikson stated that most of the literature available at the time focused on rather simplistic models and ignored two features that are important in practice: stochastic task durations and dynamic arrivals of new projects.

Choi et al. [22] formulated an infinite-horizon, discrete-time MDP model for RCPSPs with stochastic task durations, uncertain task outcomes (success or failure), and uncertain costs. In their problem setting, only one resource was needed to perform one task. They

focused on linear activity-on-node (that is, precedence constraint) networks and did not model dynamic arrivals of new projects. The state in their MDP included the status of each project, that is, which tasks are complete and which are ongoing; information about whether or not the latest task in each project was a success was also stored in the state; in addition, the state included the number of time-slots for which a resource has been used for the currently ongoing task. Since their activity-on-node network was linear, at most one task in a project can be started in one time-period. The decisions in their MDP were therefore binary, representing whether or not to start a particular task in a particular project. The resulting MDP still suffered from the curse of dimensionality. A simulation-based approximate dynamic programming (ADP) algorithm was therefore applied and compared against simple heuristics.

Choi et al. [23] extended the earlier MDP model in Choi et al. [22] by allowing new project arrivals from a certain pre-determined group of projects. Consequently, their state included an additional variable to represent the realized arrival time of each new project. The decisions were identical to those in Choi et al. [22]. They implemented a variation of the Q-learning algorithm [113] on the resulting large-scale MDP.

Melchiors [76] formulated an infinite-horizon, continuous-time MDP model for scheduling dynamically arriving projects with stochastic task durations. Project arrivals followed a Poisson process. Their model was not restricted to linear activity-on-node networks. However, as in the above two papers by Choi et al., Melchiors also made the simplifying assumption that each task needed only one resource. The state in this MDP included information about waiting and ongoing tasks in each project. The decision-maker chose the tasks to work on in each period. Again, a simulation-based ADP algorithm that employed value-function approximation was implemented.

We study infinite-horizon, discrete-time RCPSPs with dynamic arrivals of new heterogeneous projects. We allow for arbitrary arrival distributions and any (not necessarily linear) activity-on-node networks. One task may simultaneously require multiple resources. These features generalize the corresponding components of the problems studied in Choi et al.

[22, 23] and in Melchiors [76]. We focus on deterministic task durations and provide an MDP formulation of such dynamic resource constrained project scheduling problems (DRCPSPs). It turns out that this MDP is weakly coupled [3]. That is, the immediate expected profits are additively separable over project-types and transition probabilities are multiplicatively separable. Decisions about distinct project-types are only linked by resource-availability constraints. Exact solution of this MDP is intractable owing to the curse of dimensionality. Standard approaches for approximate solution of such weakly coupled MDPs include Lagrangian relaxation and approximate linear programming which is used in previous chapter [3, 41]. Unfortunately, owing to the complicated state-evolution process in our MDP, such mathematical programming-based methods are computationally difficult to implement. We therefore apply a simulation-based approximate policy iteration algorithm [8, 89] to this MDP. This method employs a value-function approximation whose parameters are tuned via simulation using least-squares fitting. We perform extensive numerical experiments on 480 randomly generated problem instances to compare the performance of this algorithm against a myopic policy. The performance of the two methods is statistically different in about 60% of the instances. The policy iteration method outperformed the myopic policy in about 74% of these statistically significant instances.

This chapter is organized as follows. Our problem setting is described precisely in the next section. An MDP formulation for this problem is developed and challenges in its exact solution are outlined in Section 2.3. A simulation-based ADP method for tackling this MDP is described in Section 2.4. Numerical results are presented in Section 2.5.

2.2 Problem statement

Consider a projection scheduling problem that evolves over discrete-time periods $t = 1, 2, 3, \ldots$. We will use the following notation.

1. $\mathcal{I} = \{1, 2, \ldots, I\}$ is the index set of project types.

2. Up to D^i new projects of type $i \in \mathcal{I}$ may arrive during one time-period. Let $p^i(m)$

denote the probability that $m \in \{0, 1, \ldots, D^i\}$ new projects of type $i \in \mathcal{I}$ arrive during one time-period.

3. For each project of type $i \in \mathcal{I}$, $\mathcal{N}^i = \{1, \ldots, N^i\}$ denotes the finite set of tasks that need to be accomplished in order to complete the project. Tasks are performed in a non-preemptive manner; i.e., once started, a task cannot be interrupted until it is complete.

4. For each task $n \in \mathcal{N}^i$, $\mathcal{M}_n^i \subset \mathcal{N}^i$ denotes the set of tasks that must be completed before starting task n. This set is called the set of predecessors of task n in project-type i.

5. We assume that task durations are deterministic. Task $n \in \mathcal{N}^i$ takes Δ_n^i time-periods to complete.

6. $\mathcal{J} = \{1, \ldots, J\}$ denotes the set of resources available. B^j is the total (integer) quantity of resource $j \in \mathcal{J}$ available in each time-period. Task $n \in \mathcal{N}^i$ requires an integer amount $b_n^{ij} \geqslant 0$ of resource $j \in \mathcal{J}$.

7. By the beginning of any time-period, a project that arrived earlier may have been completed, may be incomplete or may be waiting inception. The projects that have not been completed, i.e., the ones that are incomplete or waiting inception form a "queue". $W^i < \infty$ denotes the queue capacity for incomplete and waiting projects of type i.

8. The following rewards and costs are obtained or incurred:

   - Projects of type $i \in \mathcal{I}$ that arrive when W^i projects of that type are in the queue are rejected incurring a penalty cost G^i at the end of the time-period.
   - A reward R^i is received on completing a project of type $i \in \mathcal{I}$ at the end of a time-period.

- A cost c_n^i per time-period is charged at the end of that time period for performing task n in project-type $i \in \mathcal{I}$.
- An incomplete stalled project (no ongoing tasks) of type $i \in \mathcal{I}$ incurs a cost Q^i per period. This cost is charged at the end of the time-period.
- A holding cost H^i per project of type $i \in \mathcal{I}$ is incurred in each time-period where such a project is waiting inception. We assume that this cost is incurred at the beginning of the time-period.

9. The goal is to maximize the total discounted expected profit (rewards minus costs) over an infinite-horizon where the per-period discount factor is $0 < \alpha < 1$.

An MDP model for this class of DRCPSPs is developed next.

2.3 A Markov decision process model

The **states** of our MDP are given by $X = (X^1, \ldots, X^I)$, where matrix X^i stores information about all incomplete and waiting type-i projects (i.e., projects in queue). The number of rows in matrix X^i equals the number of type-i projects in queue and hence cannot exceed W^i. Let rows(X^i) denote the set of rows in matrix X^i. Recall that $\mathcal{N}^i = \{1, \ldots, N^i\}$ is the set of tasks in type-i projects. Each row in X^i is of length N^i. The entries X^i_{lk} in the lth row and kth column of matrix X^i are defined as follows:

$$X^i_{lk} = -1 \text{ if task } k \in \mathcal{N}^i \text{ in the } l\text{th type } i \text{ project in queue has not yet started;}$$
$$X^i_{lk} = \Delta^i_k \text{ if task } k \in \mathcal{N}^i \text{ in the } l\text{th type-}i \text{ project in queue has been completed;}$$
$$0 < X^i_{lk} < \Delta^i_k \text{ if task } k \in \mathcal{N}^i \text{ in the } l\text{th type-}i \text{ project has started but not complete.}$$

When $0 < X^i_{lk} < \Delta^i_k$, X^i_{lk} denotes the number of time-periods since the inception of task k. Let $\mathcal{X}^i$ denote the set of all state matrices that are feasible with respect to precedence constraints for type-i projects. Also let $\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \ldots \times \mathcal{X}^I$. We define the set

$\mathcal{C}_l^i(X^i) = \{k : 0 < X_{lk}^i < \Delta_k^i\}$ of tasks that have started but are incomplete. Owing to our non-preemptive setting, tasks in $\mathcal{C}_l^i(X^i)$ must be processed in the current period. The set of all feasible states is then given by

$$\bar{\mathcal{X}} = \left\{ X \in \mathcal{X} : \sum_{i=1}^{I} \sum_{l \in \text{rows}(X^i)} \sum_{k \in \mathcal{C}_l^i(X^i)} b_k^{ij} \leqslant B^j, j \in \mathcal{J} \right\}. \tag{2.1}$$

The vector A of **action** matrices in our MDP is written as $A = (A^1, \ldots, A^I)$. Here, A^i is a matrix of zeros and ones, equal in size to X^i, and indicates which tasks will be started next. $A_{lk}^i = 1$ implies that we choose to begin task k in the lth type-i project in queue; A_{lk}^i is zero otherwise. A feasible action must satisfy the following logical restrictions:

$$A_{lk}^i \;\; = \;\; 1 \;\text{ only if }\; X_{lk}^i = -1 \;\text{ and }\; X_{ln}^i = \Delta_n^i, \;\forall n \in \mathcal{M}_k^i. \tag{2.2}$$

This ensures that task k can be started only if it had not begun or completed earlier and if all of its predecessors have been completed. The set of all such action matrices for state matrix X^i is denoted by $\mathcal{U}^i(X^i)$. Also let $\mathcal{U}(X) = \mathcal{U}^1(X^1) \times \mathcal{U}^2(X^2) \times \ldots \times \mathcal{U}^I(X^I)$. Given the state-action pair (X^i, A^i), the matrix $X^i + A^i$ provides valuable information. In particular, we define the set of ongoing tasks in the lth type-i project as $\mathcal{V}_l^i(X^i, A^i) = \{k : 0 \leqslant X_{lk}^i + A_{lk}^i < \Delta_k^i\}$. Note that $\mathcal{C}_l^i(X^i) \subseteq \mathcal{V}_l^i(X^i, A^i)$ for every $A^i \in \mathcal{U}^i(X^i)$. The amount of resource $j \in \mathcal{J}$ consumed by all ongoing tasks from type-i projects equals $\sum_{l \in \text{rows}(X^i)} \sum_{k \in \mathcal{V}_l^i(X^i, A^i)} b_k^{ij}$ and is denoted by $B^{ij}\left(X^i, A^i\right)$. The set of all vectors of action matrices that are feasible in state $X \in \bar{\mathcal{X}}$ is defined as

$$\bar{\mathcal{U}}(X) = \left\{ (A^1, A^2, \ldots, A^I) \in \mathcal{U}(X) : \sum_{i=1}^{I} B^{ij}\left(X^i, A^i\right) \leqslant B^j, \;\; j = 1, 2, \ldots, J \right\}. \tag{2.3}$$

Given the state-action pair X^i, A^i, **transitions** into the new state X'^i are defined as follows. For the lth type-i project in queue, the set of tasks waiting inception is defined as $\mathcal{E}_l^i(X^i, A^i) = \{k : X_{lk}^i + A_{lk}^i = -1\}$. The set of type-$i$ projects with no tasks waiting inception

(i. e., all tasks either completed or ongoing) is defined as $\mathcal{F}^i(X^i, A^i) = \{l \in \text{rows}(X^i) : \mathcal{E}^i_l(X^i, A^i) = \varnothing\}$. A project $l \in \mathcal{F}^i(X^i, A^i)$ is completed by the end of the current time-period if all of its ongoing tasks are completed by then. The subset $\bar{\mathcal{F}}^i(X^i, A^i) \subseteq \mathcal{F}^i(X^i, A^i)$ denotes the type-i projects that will be completed by the end of the current time-period. These projects do not appear in the next state. Recall that m new type-i projects arrive at the beginning of the current time-period with probability $p^i(m)$. Depending on how full the queue is at the time, none, some, or all of these will be added to the queue. The number of rows in the next state matrix X'^i therefore equals $|\text{rows}(X'^i)| = |\text{rows}(X^i)| - |\bar{\mathcal{F}}^i(X^i, A^i)| + m - \left[|\text{rows}(X^i)| - |\bar{\mathcal{F}}^i(X^i, A^i)| + m - W^i\right]^+$, where the notation $|\cdot|$ is used throughout this chapter for set cardinalities. For all tasks k in all type-i projects $l \in \text{rows}(X'^i)$ that newly joined the queue in this manner, we have, $X'^i_{lk} = -1$. For all ongoing tasks k in each type-i project $l \in \text{rows}(X^i)$, we have, $X'^i_{lk} = X^i_{lk} + 1$.

The **immediate expected profit** earned for type-i projects on choosing a feasible action matrix A^i in state matrix X^i is given by

$$\phi^i(X^i, A^i) = \alpha R^i |\bar{\mathcal{F}}^i(X^i, A^i)| - \alpha \sum_{m=0}^{D^i} p^i(m) \left[|\text{rows}(X^i)| - |\bar{\mathcal{F}}^i(X^i, A^i)| + m - W^i\right]^+ G^i$$
$$- |\mathcal{W}^i(X^i, A^i)| H^i - \alpha |\mathcal{Z}^i(X^i, A^i)| Q^i - \alpha \sum_{l \in \text{rows}(X^i)} \sum_{k \in \mathcal{V}^i_l(X^i, A^i)} c^i_k.$$

Here, the first term $\alpha R^i |\bar{\mathcal{F}}^i(X^i, A^i)|$ equals the discounted reward earned for completing projects in the set $\bar{\mathcal{F}}^i(X^i, A^i)$. The second term equals the discounted cost of rejecting projects that cannot fit into the queue. In the third term, $\mathcal{W}^i(X^i, A^i) = \{l \in \text{rows}(X^i) : X^i_{lk} + A^i_{lk} = -1 \ \forall k \in \mathcal{N}^i\}$ is the set of type-i projects waiting inception. The third term thus accounts for the holding cost of projects that are waiting inception. In the fourth term, $\mathcal{Z}^i(X^i, A^i) = \{l : \mathcal{V}^i_l(X^i, A^i) = \varnothing,\ \mathcal{E}^i_l(X^i, A^i) \neq \varnothing,\ \mathcal{E}^i_l(X^i, A^i) \neq \mathcal{N}^i\}$ is the set of stalled type-i projects. The fourth term thus equals the discounted cost of stalled projects. The last term is the discounted operating cost of all ongoing projects. The total immediate expected profit is denoted by $\phi(X, A) = \sum_{i=1}^{I} \phi^i(X^i, A^i)$.

Let $\sigma(X)$ be the maximum infinite-horizon discounted expected profit earned starting in state $X \in \bar{\mathcal{X}}$. Then, for all $X \in \bar{\mathcal{X}}$, Bellman's equations for our MDP are given by

$$\sigma(X) = \max_{A \in \bar{\mathcal{U}}(x)} \left\{ \phi(X, A) + \alpha \sum_{m^1=0}^{D^1} \cdots \sum_{m^I=0}^{D^I} \left(\prod_{i \in \mathcal{I}} p^i(m^i) \right) \sigma(X') \right\}. \tag{2.4}$$

Since the state and action sizes are exponential in I, in W^i and in N^i for $i \in \mathcal{I}$, exact solution of this MDP is computationally intractable.

Notice that the immediate expected profit $\phi(X, A)$ in this MDP is additively separable over $\mathcal{I}$, and the transition probabilities $\prod_{i \in \mathcal{I}} p^i(m^i)$ are multiplicatively separable over $\mathcal{I}$. State components X^i and decisions A^i for projects of distinct types i from $\mathcal{I}$ are only connected through the linking state feasibility constraints (2.1) and resource constraints (2.3). Thus, this MDP is weakly coupled [3]. There are two main methods for approximate solution of weakly coupled MDPs: Lagrangian relaxation and linear programming. Unfortunately, owing to the large scale and highly combinatorial state transitions in our weakly coupled MDP, an efficient implementation of such mathematical programming-based methods or of their variations as it is used in previous chapter and [41] does not seem possible. We instead resort to a simulation-based ADP method as described in the next section.

2.4 A simulation-based ADP method

There are four main ideas in the ADP method we describe here. First, we employ the notions of pre- and post-decision states as in [89] to partly handle the curse of dimensionality. Specifically, the state transitions from X^i to X'^i described above are partitioned into two pieces. The first piece corresponds to the state-transformation that results only from choosing action A^i. The second piece incorporates random project arrivals that occur after A^i is selected. This considerably simplifies the Bellman's equations in (2.4). We then define an approximation to the value function of the post-decision states. Parameters of this value function are tuned via a simulation-based policy iteration approach that relies on least-squares fitting as in [8, 89]. The resulting approximate value function is then substituted in

the right hand side of the Bellman's equations to retrieve decisions in states that are visited in run-time. Mathematical details of this approach are described in the next three sections.

2.4.1 Pre- and post-decision states, and value function approximation

Given the pre-decision state X^i and action A^i, the post-decision state $X^i(A^i)$ is defined as follows. The completed projects, that is, the projects in set $\bar{\mathcal{F}}^i(X^i, A^i)$, do not appear in the post-decision state. For all ongoing tasks k in each type-i project $l \in \text{rows}(X^i)$, we have, $X^i_{lk}(A^i) = X^i_{lk} + 1$. The effect of random project arrivals is then incorporated into this post-decision state to create the next pre-decision state X'^i. Let $X(A) = (X^1(A^1), \ldots, X^I(A^I))$. Then, based on this notion of a post-decision state, the Bellman's equations in (2.4) can be rewritten as

$$\sigma(X) = \max_{A \in \mathcal{U}(X)} \left\{ \phi(X, A) + \alpha \sigma^A(X(A)) \right\}, \tag{2.5}$$

where $\sigma^A(X(A)) = E[\sigma(X')|X(A)]$. Here, the expectation is taken with respect to the probabilistic project arrival process. If the post-decision value function $\sigma^A(X(A))$ were known, the Bellman's equations in (2.5) only require the solution of a deterministic optimization problem. This is precisely the benefit of the concept of a post-decision state. As explained in [89], however, the post-decision value function needs to be typically approximated in practice. In this chapter, we employ the following parameterized approximation:

$$\widetilde{\sigma}^A(X(A); \vec{r}) = r_0 + \sum_{i \in \mathcal{I}} \left[|\bar{\mathcal{F}}^i(X^i(A^i))| r^i_1 + |\mathcal{W}^i(X^i(A^i))| r^i_2 \right]. \tag{2.6}$$

Here, $\bar{\mathcal{F}}^i(X^i(A^i))$ is the set of type-i projects that will be completed at the end of the next period given that the post-decision state in this period is $X^i(A^i)$. Moreover, $\mathcal{W}^i(X^i(A^i))$ is the set of type-i projects that will be waiting inception in the next pre-decision state given the post-decision state $X^i(A^i)$. Finally, $\vec{r} = (r_0; (r^1_1, r^1_2); \ldots; (r^I_1, r^I_2))$ is a vector of unknown parameters. In the next section, we apply an approximate policy iteration approach from

[8, 89] that uses least-squares fitting to adaptively tune these unknown parameters.

2.4.2 Implementation of least-squares approximate policy iteration

The policy iteration algorithm is initialized with an arbitrary parameter vector for the value-function approximation, which is updated in every iteration. Each iteration of the algorithm includes two main parts. The first part consists of creating K sample paths each including τ time-steps. The kth sample path begins at an initial post-decision state. This kth sample path is generated by following a policy that is optimal with respect to the current value-function approximation. The discounted expected profit accumulated over τ steps in this sample path is calculated. In the second part of the algorithm, these K profits then serve as input for a least-squares data fitting problem, where an optimal value-function parameter vector is derived. The new parameter vector is then calculated as a convex combination of the old and this optimal parameter vector before starting the next iteration. Details of this algorithm are listed below.

- **Step** 0**: Initialization**.
  - Step 0a. Initialize the iteration counter n to 1; fix positive integers τ, K, N; and fix a step-size parameter $\gamma > 1$.
  - Step 0b. Initialize a parameter vector $\vec{r}^{\,1}$ arbitrarily.
- **Step** 1**: Policy evaluation through simulation**.
  - Step 1a. Set replication counter $k = 1$.
  - Step 1b. Warm-up: generate a post-decision state X_k^* and the next pre-decision state X_0. Set $t = 0$.
  - Step 1c.

* Find an optimal decision A_t by solving

$$A_t = \underset{A \in \bar{\mathcal{U}}(X_t)}{\operatorname{argmax}} \left\{ \phi(X_t, A) + \alpha \widetilde{\sigma}^A(X_t(A), \vec{r}^{\,n}) \right\}. \tag{2.7}$$

* Compute $\phi(X_t, A_t)$.
* Obtain the next pre-decision state X_{t+1} from X_t and A_t.
* If $t < \tau$, increment t by one and go to Step 1c.

– Step 1*d*. Compute discounted expected profit accumulated in periods $t = 0, 1, 2, \ldots, \tau$, by starting in state X_k^* as $\psi(k) = \sum_{t=0}^{\tau} \alpha^t \phi(X_t, A_t)$.

– Step 1*e*. If $k < K$, increment q and go to Step 1*b*.

- **Step 2: Value-function approximation update**

  – Step 2*a*. Find a parameter vector by solving the least-squares fitting problem $\vec{r}^{\,*} = \underset{\vec{r}}{\operatorname{argmin}} \left\{ \sum_{k=1}^{K} \left\{ \psi(k) - \widetilde{\sigma}(X_k^*, \vec{r}^{\,n}) \right\}^2 \right\}$.

  – Step 2*b*. Let $\beta_n = \gamma/(\gamma + n - 1)$, and update the parameter vector as $\vec{r}^{\,n+1} = (1 - \beta_n)\vec{r}^{\,n} + \beta_n \vec{r}^{\,*}$.

- **Step 3.** If $n < N$, increment n by one and go to Step 1.

In our numerical results below, we implemented a myopic policy starting in an empty state to deliver the initial post-decision state X_k^* in Step 1b of this algorithm. The step-size formula $\beta_n = \gamma/(\gamma + n - 1)$ is from [89]. Finally, the deterministic optimization problem in (2.7) is identical in form to the problems we need to solve for online retrieval of decisions. Our solution method for solving these problems is described next.

2.4.3 Online retrieval of decisions

Let $\vec{r}^{\,*}$ be the final parameter vector after which the above algorithm is terminated. Suppose that we want to compute a decision vector that is optimal in a pre-decision state $X \in \bar{\mathcal{X}}$ with

respect to the value function approximation (2.6). This is done by substituting $\widetilde{\phi}^A(X(A);\vec{r}^*)$ from (2.6) in place of $\phi^A(X(A))$ in the right hand side of (2.5). This yields the following deterministic optimization problem:

$$\max_{A\in\bar{\mathcal{U}}(X)}\left\{\phi(X,A)+\alpha\left(r_0^*+\sum_{i\in\mathcal{I}}\left[|\bar{\mathcal{F}}^i(X^i(A^i))|r_1^{*i}+|\mathcal{W}^i(X^i(A^i))|r_2^{*i}\right]\right)\right\}. \tag{2.8}$$

The constant term αr_0^* from this problem can be ignored. By recalling the definitions of $\phi(X,A)$, $|\bar{\mathcal{F}}^i(X^i(A^i))|$ and $|\mathcal{W}^i(X^i(A^i))|$, it can be seen that they are not linear in components of A. By defining additional decision variables and associated logical constraints, we reformulate problem (2.8) so that it becomes linear. We first describe some new set notations.

Let $\mathcal{V}_l^i(X^i)$ denote the set of ongoing tasks in the lth type-i project in state X^i. We use $\mathcal{E}_l^i(X^i)$ to denote the set of tasks in the lth type-i project in state X^i, which have not been started yet. Similarly, let $\bar{\mathcal{E}}_l^i(X^i)$ denote the subset of tasks in $\mathcal{E}_l^i(X^i)$ that *could* be started because their predecessor tasks have been completed. Moreover, we use $\mathcal{W}^i(X^i)$ to denote the set of type-i projects that are waiting inception in state X^i. Similarly, let $\mathcal{Z}^i(X^i)$ be the set of type-i projects that are stalled in state X^i. Let $\mathcal{F}^i(X^i)$ denote the set of type-i projects in which all tasks have been started in state X^i. Also let $\bar{\mathcal{F}}_1^i(X^i)\subseteq\mathcal{F}^i(X^i)$ be the type-i projects that will be completed by the end of the current time-period because all incomplete tasks in these projects only need one time-period of additional work. Moreover, $\bar{\mathcal{F}}_2^i(X^i)$ is the set of type-i projects in state X^i that have the following properties: (i) predecessors of all tasks that have not been started yet are complete, and durations of all such tasks equal one time-period; and (ii) all ongoing tasks need only one time-period of additional work. If we chose to start all remaining tasks in a project in $\bar{\mathcal{F}}_2^i(X^i)$, then that project will be completed by the end of the current time-period. Similarly, let $\bar{\mathcal{F}}_3^i(X^i)\subseteq\mathcal{F}^i(X^i)$ be the type-i projects that will be completed by the end of the next time-period because all incomplete tasks in these projects need at most two time-periods of additional work with at least one which needs two time-periods of additional work. Finally, let $\bar{\mathcal{F}}_4^i(X^i)$ be the set of type-i in state

X^i with the following properties: (i) predecessors of all tasks that have not been started yet are complete, and durations of all such tasks equal either one or two time-periods; and (ii) all ongoing tasks need one or two time-periods of additional work; (iii) $\bar{\mathcal{F}}_2^i(X^i) \cap \bar{\mathcal{F}}_4^i(X^i) = \varnothing$.

We now describe new decision variables that can in fact be recovered from a decision-matrix A, but defining them separately helps in linearizing the above problem. For every type-i project l in state X^i, we define binary decision variables f_l^i, $\bar{f}_l^i$, w_l^i, and z_l^i. The decision variable f_l^i defines whether (= 1) or not (= 0) the lth type-i project will be completed at the end of the *current* time-period. Similarly, $\bar{f}_l^i$ defines whether (= 1) or not (= 0) the lth type-i project will be completed at the end of the *next* time-period. If the lth type-i project is waiting inception, then the decision variable w_l^i is 1 if we begin that project in this time-period; w_l^i is 0 otherwise. Similarly, if the lth type-i project is stalled, decision variable z_l^i is 1 if that project resumes in this time-period; z_l^i is 0 otherwise.

Finally, for every $m \in \{0, \ldots, D^i\}$, let $g_m^i = \max\{0, |\text{rows}(X^i)| - (|\bar{\mathcal{F}}_1^i(X^i)| + \sum_{l \in \bar{\mathcal{F}}_2^i(X^i)} f_l^i) + m - W^i\}$. Problem (2.8) can then be rewritten as

$$\max_{A,f,\bar{f},w,z,g} \sum_{i=1}^{I} \Bigg(\alpha R^i (|\bar{\mathcal{F}}_1^i(X^i)| + \sum_{l \in \bar{\mathcal{F}}_2^i(X^i)} f_l^i) - \alpha \sum_{m=0}^{D^i} p^i(m) g_m^i G^i - (|\mathcal{W}^i(X^i)| - \sum_{l \in \mathcal{W}^i(X^i)} w_l^i) H^i$$
$$- (|\mathcal{Z}^i(X^i)| - \sum_{l \in \mathcal{Z}^i(X^i)} z_l^i) Q^i - \alpha \sum_{l \in \text{rows}(X^i)} \Big(\sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i c_k^i + \sum_{k \in \mathcal{V}_l^i(X^i)} c_k^i \Big) \Bigg) +$$
$$\alpha \sum_{i=1}^{I} \Bigg\{ r_1^{*i} \big(|\bar{\mathcal{F}}_3^i(X^i)| + \sum_{l \in \bar{\mathcal{F}}_4^i(X^i)} \bar{f}_l^i \big) + r_2^{*i} \big(|\mathcal{W}^i(X^i)| - \sum_{l \in \mathcal{W}^i(X^i)} w_l^i \big) \Bigg\}$$

subject to:

$$\sum_{i=1}^{I} \Big(\sum_{l \in \text{rows}(X^i)} \Big(\sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i b_k^{ij} + \sum_{k \in \mathcal{V}_l^i(X^i)} b_k^{ij} \Big) \leqslant B^j, \ j \in \mathcal{J}, \tag{2.9}$$

$$f_l^i \leqslant A_{lk}^i, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_2^i(X^i) \ \text{ and } \ k \in \bar{\mathcal{E}}_l^i(X^i), \tag{2.10}$$

$$f_l^i \geqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i - |\bar{\mathcal{E}}_l^i(X^i)| + 1, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_2^i(X^i), \tag{2.11}$$

$$\bar{f}_l^i \leqslant A_{lk}^i, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_4^i(X^i) \ \text{ and } \ k \in \bar{\mathcal{E}}_l^i(X^i), \tag{2.12}$$

$$\bar{f}_l^i \geqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i - |\bar{\mathcal{E}}_l^i(X^i)| + 1, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_4^i(X^i), \tag{2.13}$$

$$w_l^i \leqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i, i \in \mathcal{I} \text{ and } l \in \mathcal{W}^i(X^i), \tag{2.14}$$

$$w_l^i \geqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i / |\bar{\mathcal{E}}_l^i(X^i)|, i \in \mathcal{I} \text{ and } l \in \mathcal{W}^i(X^i), \tag{2.15}$$

$$z_l^i \leqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i, i \in \mathcal{I} \text{ and } l \in \mathcal{Z}^i(X^i), \tag{2.16}$$

$$z_l^i \geqslant \sum_{k \in \bar{\mathcal{E}}_l^i(X^i)} A_{lk}^i / |\bar{\mathcal{E}}_l^i(X^i)|, i \in \mathcal{I} \text{ and } l \in \mathcal{Z}^i(X^i), \tag{2.17}$$

$$g_m^i \geqslant |\text{rows}(X^i)| - (|\bar{\mathcal{F}}_1^i(X^i)| + \sum_{l \in \bar{\mathcal{F}}_2^i(X^i)} f_l^i) + m - W^i, \quad i \in \mathcal{I} \text{ and } m \in \{0, \ldots, D^i\}, \tag{2.18}$$

$$g_m^i \geqslant 0, \text{ integer}, \quad i \in \mathcal{I} \text{ and } m \in \{0, \ldots, D^i\}, \tag{2.19}$$

$$A_{lk}^i, \text{binary}, i \in \mathcal{I}, l \in \text{rows}(X^i) \text{ and } k \in \bar{\mathcal{E}}_l^i(X^i), \tag{2.20}$$

$$f_l^i, \text{binary}, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_2^i(X^i), \tag{2.21}$$

$$\bar{f}_l^i, \text{binary}, i \in \mathcal{I}, l \in \bar{\mathcal{F}}_4^i(X^i), \tag{2.22}$$

$$w_l^i, \text{binary}, i \in \mathcal{I}, l \in \mathcal{W}^i(X^i), \tag{2.23}$$

$$z_l^i, \text{binary}, i \in \mathcal{I}, l \in \mathcal{Z}^i(X^i). \tag{2.24}$$

In this problem (2.9) are resource constraints. Constraints (2.10)-(2.17) are logical restrictions that relate various decision variables as described next. Specifically, constraints (2.10)-(2.11) are forcing constraints that relate f_l^i and A_{lk}^i for $l \in \bar{\mathcal{F}}_2^i(X^i)$ and $k \in \bar{\mathcal{E}}_l^i(X^i)$. Similarly, constraints (2.12)-(2.13) are forcing constraints that relate $\bar{f}_l^i$ and A_{lk}^i for $l \in \bar{\mathcal{F}}_4^i(X^i)$ and $k \in \bar{\mathcal{E}}_l^i(X^i)$. Constraints (2.14)-(2.15) are forcing constraints that relate w_l^i to A_{lk}^i for $l \in \mathcal{W}^i(X^i)$. Constraints (2.16)-(2.17) are forcing constraints that relate z_l^i to A_{lk}^i for $l \in \mathcal{Z}^i(X^i)$. We use this linear integer program to recover decisions that are greedy with respect to the value function approximation parameterized by $\vec{r}^*$ in every visited pre-decision state. The same process is also employed to solve the deterministic optimization problem (2.7) in Step 1c of the policy iteration algorithm. In the next section, we present numerical

results on 480 test problems using this approach.

2.5 Computational results

All simulations were performed on a 3.1 GHz iMac desktop with 16 GB RAM and an Intel Core i7 chip running Mac OS X 10.9.3. All linear and linear integer programs were solved using CPLEX 12 from IBM via a MATLAB R2016a front-end. We compared the performance of the simulation-based approximate policy iteration algorithm with a standard myopic policy. In any state $X \in \bar{\mathcal{X}}$, a myopic decision was obtained by solving the problem $\max_{A \in \bar{\mathcal{U}}(X)} \phi(X, A)$, which was a linear integer program.

Our numerical experiments were performed on randomly generated problem instances as is often the case in the scheduling literature [41, 76, 84] and it is also applied in previous chapter. We generated our test instances using a technique from chapter 1 and [41], which in turn, was an extension of a standard method to create multi-dimensional knapsack problems [24]. Since we have several new problem parameters that were not relevant in these earlier papers and previous chapter, we had to modify their problem generation approach as described next.

The discount factor was set to 0.85 as in chapter 1 and [3, 41]. The number of job types was fixed at $I = 5$ and $I = 10$ in two different sets of problems. The number of resources was set to $J = 2$ in all instances. The number of tasks N^i in type-i projects was a randomly sampled integer between 5 and 25; we henceforth describe this process as $N^i \sim \text{DU}[5, 25]$. The arrival probabilities were generated as in previous chapter and [41]. Some of the other parameters were generated as follows: $W^i \sim \text{DU}[1, 5]$; $R^i \sim \text{DU}[5000, 10000]$; $G_i \sim \text{DU}[1, 100]$; $H_i \sim \text{DU}[1, 5]$; $Q_i \sim \text{DU}[1, 5]$; $c^i_n \sim \text{DU}[1, 3]$; $\Delta^i_n \sim \text{DU}[1, 4]$. The activity-on-node network for each project-type was generated as a random lower triangular adjacency matrix with zeros on its diagonal. We also used $b^{ij}_n \sim \text{DU}[1, 7]$. The resource availability B^j was then set to $\lfloor I \times \rho \times \sum_{i=1}^{I} \sum_{n=1}^{N^i} b^{ij}_n \rfloor$ as in chapter 1 and [24, 41]. Here, ρ is commonly called the tightness ratio. For each I, we used tightness ratios $\rho = 0.01, 0.02, 0.03, 0.04, 0.05, 0.07, 0.09, 0.1$. For each I, ρ combination, 30 problem instances were generated randomly.

Parameters for the simulation-based policy iteration method were set as follows. The step-size parameter γ was set to 10 as in Figure 6.3 from [89]. The replication parameter Q was set to 50 as in [8], and the simulation horizon was $\tau = 50$ within each replication. The number of iterations was set to $N = 100$. The warm-up duration at the beginning of every replication to find X_q^* was set to 50 periods.

The simulation-based policy iteration algorithm and the myopic method were compared using the average profit over 200 independent simulations with 50 period each. Each of these simulations was started in an empty state, and a simulation-based and a myopic decision was obtained in each visited state. The next state was then found by sampling a random number of new arrivals.

Tables 2.1 and 2.2 summarize our results for $I = 5, 10$, respectively. Different rows in each table correspond to distinct values of the tightness ratio ρ listed in the first column. In each row, we report results for 30 randomly generated test instances. The second column lists the number of problem instances (out of these 30) in which the average profits (over 200 simulations) obtained by the two methods were statistically different as per a paired t-test at significance threshold 0.05. Out of these statistically significant cases, the number of instances in which the simulation-based method generated a higher average profit than the myopic approach is listed in the third column. The average percentage improvement achieved by the simulation-based method over the myopic method in these instances is reported in the fourth column.

ρ	# significant	# $S > M$	% improvement
0.01	22	8	2.6
0.02	21	12	6.9
0.03	17	11	7.9
0.04	7	6	7.8
0.05	11	8	6.23
0.07	9	5	2.3
0.09	13	6	0.09
0.1	12	8	4.4

Table 2.1: Results for 8 problem sets (with different values of ρ) when $I = 5$. Each row lists results for 30 randomly generated test instances.

ρ	# significant	# $S > M$	% improvement
0.01	20	13	4.1
0.02	18	13	3.12
0.03	20	18	1.5
0.04	25	22	2.4
0.05	21	19	9.5
0.07	25	24	2.6
0.09	26	24	5.4
0.1	25	20	7.37

Table 2.2: Results for 8 problem sets (with different values of ρ) when $I = 10$. Each row lists results for 30 randomly generated test instances.

The two tables show that out of the 480 problem instances listed, the difference between the two methods was statistically significant in 292, that is, in 60% of the problem instances. Of these 292 instances where the difference was significant, the simulation-based method outperformed the myopic approach in 217, that is, in 74% of the instances. In these instances, the simulation-based method outperformed the myopic method by about 4.7% on average. The tables also suggest that the simulation-based method performs better for $I = 10$ than for $I = 5$. This work here suggests that simulation-based policy iteration is a viable method for solving DRCPSPs.

The next chapter studies weakly coupled MDPs with imperfect information. Two different settings for imperfect information are considered: in the first case, the transition probabilities are unknown to the decision-maker, and in the second case, the state of the system is not observable.

Chapter 3

WEAKLY COUPLED MARKOV DECISION PROCESSES WITH IMPERFECT INFORMATION

3.1 Introduction

The previous two chapters studied two classes of scheduling problems that can be formulated as weakly coupled MDPs, and employed approximate dynamic programming methods for their solution. In this chapter, we revert back to general weakly coupled MDPs, and consider the scenario where a decision-maker only has partial knowledge about the system's behavior. Two closely related special cases of this situation are considered.

In the first case (Section 3.4), the decision-maker does not know the state transition probabilities. In particular, the transition probabilities for each subproblem in the weakly coupled MDP are parameterized, and the decision-maker does not know these parameters. The decision-maker begins with a belief distribution on each of these parameters, and updates these beliefs as the state of the system evolves. The resulting problem is called a Bayes-adaptive weakly coupled MDP. Exact solution of this formulation is intractable. Approximation methods rooted in semi-stochastic certainty equivalent control and Thompson sampling are proposed. This methodology is applied to a dynamic resource allocation problem where state transitions are characterized by a geometric distribution with parameters that are unknown to the decision-maker. Numerical experiments on test instances of this resource allocation problem are performed.

In the second case (Section 3.5), the decision-maker cannot "see" the actual states of the MDP subproblems, but can only probabilistically infer them based on noisy observation signals. The decision-maker maintains a probabilistic belief about the actual state of the system and updates this belief as noisy observations are made. The resulting problem is

a partially observable weakly coupled MDP. Again, its exact solution is intractable, and approximation methods rooted in semi-stochastic certainty equivalent control and Thompson sampling are proposed. These algorithms are applied to a restless multi-armed bandit problem with noisy state observations. Two other algorithms based on Lagrangian relaxation and state-discretization, which are particularly suitable for these restless multi-armed bandit problems, are also proposed. These four algorithms are compared via numerical experiments on test instances of the restless multi-armed bandit problem.

3.2 Background on weakly coupled MDPs

Weakly coupled MDP formulations of various scheduling; resource allocation; queueing; supply chain; and mutli-armed and restless bandit problem are presented in [3, 14, 8, 29, 40, 41, 50, 83, 114]. A formal description of weakly coupled MDPs is recalled here from [3, 50]. In a weakly coupled MDP, $\mathcal{I} = \{1, 2, \ldots, I\}$ denotes the set of sub-MDPs, indexed by i. The finite state-space for the ith MDP is denoted by the set S_i. The state-space of the weakly coupled MDP is then $S = \underset{i=1}{\overset{I}{\times}} S_i$. The finite action-space for the ith sub-MDP in state $s_i \in S_i$ is $A_i(s_i)$. The set of actions in state $s = (s_1, \ldots, s_I)$ of the weakly coupled MDP is $A(s) = \underset{i=1}{\overset{I}{\times}} A_i(s_i)$. The set of feasible actions is given by $\bar{A}(s) = \{a \in A(s) : \sum_{i=1}^{I} D_i(s_i, a_i) \leqslant b\}$, where, for each $i = 1, \ldots, I$, $D_i(s_i, a_i)$ are m-dimensional (column) vector functions of (s_i, a_i), and b is an m-dimensional (column) vector. The sub-MDPs are joined through these linking constraints. Upon choosing an action $a \in A(s)$ in state $s \in S$, the decision-maker receives a reward $r(s, a) = \sum_{i=1}^{I} r_i(s_i, a_i)$, and the system transitions to state $s' = (s'_1, \ldots, s'_I)$ with probability $p(s'|s, a) = \prod_{i=1}^{I} p_i(s'_i|s_i, a_i)$. That is, the rewards are additively separable and the transition probabilities are multiplicatively separable. The decision-maker's goal is to maximize the total expected discounted reward over an infinite horizon, with discount factor $0 < \alpha < 1$.

Bellman's equations for such a weakly coupled MDP are given by

$$J(s) = \max_{a \in \bar{A}(s)} \left\{ \sum_{i \in \mathcal{I}} r_i(s_i, a_i) + \alpha \sum_{s' \in S} p(s'|s,a) J(s') \right\}, s \in S. \tag{3.1}$$

Exact solution of these Bellman's equations by standard techniques such as value iteration, policy iteration, or linear programming is computationally intractable owing to the curse-of-dimensionality rooted in the I-dimensional state- and action-spaces of the problem. One approximate solution technique based on Lagrangian relaxation was described in [3, 50]. It is recalled here briefly.

The first step in the Lagrangian relaxation approach is to relax the linking constraints and add a corresponding penalty to the objective function using nonnegative Lagrange multipliers $\lambda \in \Re_+^m$ (row vector). This yields the relaxed Bellman's equations

$$J^\lambda(s) = \max_{a \in A(s)} \left\{ \sum_{i \in \mathcal{I}} r_i(s_i, a_i) + \lambda(b - \sum_{i=1}^{I} D_i(s_i, a_i)) + \alpha \sum_{s' \in S} p(s'|s,a) J^\lambda(s') \right\}. \tag{3.2}$$

It is shown in [3] that

$$J^\lambda(s) = \frac{1}{1-\alpha} \lambda b + \sum_{i=1}^{I} V_i^\lambda(s_i), \tag{3.3}$$

where, for $s_i \in S_i$,

$$V_i^\lambda(s_i) = \max_{a_i \in A_i(s_i)} \left\{ r_i(s_i, a_i) - \lambda D_i(s_i, a_i) + \alpha \sum_{s_i' \in S_i} p_i(s_i'|s_i, a_i) V_i^\lambda(s_i') \right\}. \tag{3.4}$$

The aggregate value function is defined as $H^\lambda(\beta) = \sum_{s \in S} \beta(s) J^\lambda(s)$, where β is the initial state-distribution. Then, applying the linear programming approach (see Puterman [91]) to Bellman's equations (3.2), we can obtain the best λ and $V_i(\cdot)$, for $i = 1, 2, \ldots, I$. That is, we have,

$$H^{\lambda *}(\beta) = \min_{V(.), \lambda} \frac{\lambda b}{1-\alpha} + \sum_{i=1}^{I} \sum_{s_i \in S_i} \beta_i(s_i) V_i(s_i)$$

$$V_i(s_i) \geqslant r_i(s_i, a_i) - \lambda D_i(s_i, a_i) + \alpha \sum_{s_i' \in S_i} p_i(s_i'|s_i, a_i) V_i(s_i'), (s_i, a_i) \in \mathcal{A}_i, i \in \{1, \ldots, I\},$$

$$\lambda \geqslant 0. \tag{3.5}$$

Here, $\mathcal{A}_i$ is the set of all state-action pairs for sub-MDP i.

Note that in problem (3.5), the number of constraints grows linearly in I, and solving this linear program is computationally tractable in some applications [3, 40, 41]. After an optimal (or sub-optimal) λ^* and $V_i^{\lambda^*}(\cdot)$, for $i = 1, 2, \ldots, I$, are available by solving problem (3.5) exactly or approximately, these can be substituted back in the right hand side of (3.1) via the formula (3.3). This enables the decision-maker to retrieve decisions in every visited state during runtime by solving (3.1).

Section 3.4 considers an extension of the above weakly coupled MDP where the transition probabilities $p_i(s_i'|s_i, a_i)$, for $i = 1, \ldots, I$, are unknown to the decision-maker. Section 3.5 studies an extension where the decision-maker cannot observe states s_i, for $i = 1, \ldots, I$.

3.3 Literature review

A majority of the existing literature on weakly coupled MDPs with imperfect information focuses on a very special case of the problems studied in this chapter. This literature considers sequential problems where the decision-maker may not know some of the parameters. The decision-maker starts with a prior belief about these parameters and updates this belief as the problem evolves. This belief distribution defines a so-called "information state" for the problem, which evolves with time. The resulting problem can be formulated as a Bayes-adaptive MDP, which may turn out to be weakly coupled in the components of the information state. Crucially, these problems do not possess a physical state. Perhaps the most famous example is the multi-armed bandit problem studied by Gittins and Jones [38], where they prove the optimality of an index policy. Some researchers have applied this multi-armed bandit theory to the design of clinical trials and to treatment planning (see [4, 80, 109] for some recent examples). Caro and Gallien [16] use a multi-armed bandit framework to model

a dynamic assortment problem with unknown demand for seasonal consumer goods. Again, none of these problems include a physical state and hence are considerably easier to solve than the ones in this chapter.

Liu et al. in [70] study a regret minimization approach for a restless multi-armed bandit problem with unknown transition probabilities. Variations of this approach are pursued for other multi-armed bandit problems in [59, 107]. The analytical results in these papers carefully exploit the structure of their problems, and do not seem to be generalizable to our Bayes-adaptive weakly coupled MDPs.

Meshram and Gopalan [77] provide an index policy for two-state restless multi-armed bandit problems with partially observable states. They outline applications to multi-channel communication systems and to advertisement placement systems. Their work is rooted in the classic idea of Whittle index [114], but does not seem to generalize to our partially observable weakly coupled MDP setting. The recent book by Krishnamurthy [63] also defines a partially observable MDP for such multi-armed bandit problems (even with more than two states). The book mentions that index policies are not tractable for such high-dimensional problems.

3.4 Bayes-adaptive weakly coupled MDPs

Suppose that the transition probabilities for the ith subproblem are characterized by a parameter θ_i. To emphasize this, we denote these transition probabilities by $\psi^i_{\theta_i}(s'_i|s_i, a_i)$. This parameter can take K_i possible values from the set $\{\theta_i^1, \theta_i^2, \ldots, \theta_i^{K_i}\}$. The decision-maker does not know the actual value of this parameter, but begins with a prior probability mass function on its possible values. This prior is updated as state observations are made. In particular, suppose x_i is a vector of size $K_i - 1$ such that x_i^k is the probability that $\theta_i = \theta_i^k$, for $k \in \{1, 2, \ldots, K_i - 1\}$. Note that the probability that $\theta_i = \theta_i^{K_i}$ can be calculated simply as $1 - \sum_{k=1}^{K_i-1} x_i^k$ and hence need not be stored. This x_i is called the information state of the problem. If the decision-maker chooses action a_i in physical state s_i and observes physical

state s'_i, the information state is updated according to the formula

$$x'^k_i = \frac{x^k_i \psi^i_{\theta^k_i}(s'_i|s_i,a_i)}{\sum_{k=1}^{K_i-1} x^k_i \psi^i_{\theta^k_i}(s'_i|s_i,a_i) + (1-\sum_{k=1}^{K_i-1} x^k_i)\psi^i_{\theta^{K_i}_i}(s'_i|s_i,a_i)}, \; k \in \{1,\ldots,K_i-1\}. \tag{3.6}$$

The physical-state, information-state pair (s_i, x_i) is a sufficient statistic for the decision-maker's reward maximization problem [13]. This allows us to write the reward maximization problem as a Bayes-adaptive MDP. The state in this MDP is (s_i, x_i) and the transition probabilities are

$$\begin{aligned} p_i(s'_i|s_i,x_i,a_i) &= x^1_i \psi^i_{\theta^1_i}(s'_i|s_i,a_i) + x^2_i \psi^i_{\theta^2_i}(s'_i|s_i,a_i) + \ldots + x^{K_i-1}_i \psi^i_{\theta^{K_i-1}_i}(s'_i|s_i,a_i) \\ &+ (1-\sum_{k=1}^{K_i-1} x^k_i)\psi^i_{\theta^{K_i}_i}(s'_i|s_i,a_i). \end{aligned} \tag{3.7}$$

The expected reward is

$$\begin{aligned} r^i(s_i,x_i,a_i) &= x^1_i r_{i,\theta^1_i}(s_i,a_i) + x^2_i r_{i,\theta^2_i}(s_i,a_i) + \ldots + x^{K_i-1}_i r_{i,\theta^{K_i-1}_i}(s_i,a_i) \\ &+ (1-\sum_{k=1}^{K_i-1} x^k_i) r_{i,\theta^{K_i}_i}(s_i,a_i). \end{aligned} \tag{3.8}$$

Note that we allow the expected rewards $r_{i,\theta^1_i}(s_i,a_i)$ to depend on the true value of parameter θ_i (via the transition probabilities). The Bellman equation for this Bayes-adaptive MDP is given by

$$J(s,x) = \max_{a \in \bar{A}(s)} \left\{ \sum_{i=1}^{I} r_i(a_i,s_i,x_i) + \alpha \sum_{s'} \prod_{i=1}^{I} p_i(s'_i|s_i,x_i,a_i) J(s',x') \right\}, s \in S, x \in \bigtimes_{i \in \mathcal{I}} [0,1]^{K_i-1}. \tag{3.9}$$

This Bayes-adaptive MDP is weakly coupled. Its exact solution is intractable. The standard approximation based on Lagrangian relaxation as discussed in Section 3.2 is not applicable because the information state is continuous. This chapter compares two approximation

techniques for this problem.

3.4.1 Semi-stochastic CEC

In semi-stochastic CEC, (some of) the random variables in a decision problem are replaced by their nominal (often expected) values [13]. For our model, at every time-step, the parameter of the transition probability for problem i is set to equal the expectation of the corresponding prior. That is, in state (s, x), the decision-maker wishes to find an action that solves

$$J_{\bar{\theta}(x)}(s) = \max_{a \in \bar{A}(s)} \left\{ \sum_{i=1}^{I} r_{i,\bar{\theta}_i(x_i)}(a_i, s_i) + \alpha \sum_{s'} \prod_{i=1}^{I} \psi^i_{\bar{\theta}_i(x_i)}(s'_i | s_i, a_i) J_{\bar{\theta}(x)}(s') \right\}, s \in S. \tag{3.10}$$

Here, $\bar{\theta}_i(x_i)$ is the expectation of the prior distribution x_i. This problem is solved approximately using the usual Lagrangian relaxation approach for weakly coupled MDPs and the resulting action is implemented. The physical state evolves to a new state, the information state is updated as described above, and the process continues.

3.4.2 Thompson sampling

Thompson sampling is a classic algorithm [101, 103] that can be implemented in our context as follows. The decision-maker in state (s, x) samples the parameters of the transition probabilities according to the prior distribution x. We denote these sampled parameters by $\hat{\theta}_i(x_i)$, for $i = 1, 2, \ldots, I$. The decision-maker then uses the standard Lagrangian relaxation approach to approximately solve the resulting weakly coupled MDP

$$J_{\hat{\theta}(x)}(s) = \max_{a \in \bar{A}(s)} \left\{ \sum_{i=1}^{I} r_{i,\hat{\theta}_i(x_i)}(a_i, s_i) + \alpha \sum_{s'} \prod_{i=1}^{I} \psi^i_{\hat{\theta}_i(x_i)}(s'_i | s_i, a_i) J_{\hat{\theta}(x)}(s') \right\}, s \in S. \tag{3.11}$$

The decision-maker implements the action prescribed by this Lagrangian approach, the system evolves to a new physical state, the information state is updated, and the process continues.

3.4.3 Application to dynamic resource allocation

This section applies the above ideas to an imperfect information extension of a dynamic resource allocation problem that was examined in [40]. The problem definition in the case of perfect information is first recalled verbatim from [40] below.

1. The set $\mathcal{I} = \{1, 2, \ldots, I\}$ denotes the heterogeneous job types.

2. For each job type $i \in \mathcal{I}$ there is a maximum possible number of new arrivals denoted by $0 < N^i < \infty$ per period. The parameter $p^i(m)$ is the probability that $1 \leqslant m \leqslant N^i$ new jobs of type $i \in \mathcal{I}$ arrive at every period. Note that the arrival probability is independent among different job types.

3. Type i jobs need to be served for a geometrically distributed number of time periods before they are complete. These geometric job durations are independent random variables with success probability $0 < q^i \leqslant 1$.

4. The $\mathcal{J} = \{1, \ldots, J\}$ is the set of resources with $0 < b^j < \infty$ as the resource availability for resource $j \in \mathcal{J}$ in each period. The resource consumption of a unit job type $i \in \mathcal{I}$ from resource type $j \in \mathcal{J}$ is $a^{ij} \geqslant 0$ at every period. For each job type $i \in \mathcal{I}$, there is at least one resource $j \in \mathcal{J}$ that $a^{ij} > 0$.

5. There is a queue for each job type that holds the incomplete jobs. The queue capacity for every job type $i \in \mathcal{I}$ is $0 < W^i < \infty$. Note that an ongoing job can be interrupted and resumed later.

6. The following lists the reward and cost at every period:

   - After completing a type $i \in \mathcal{I}$ job at one period, the reward R^i is collected at the end of the time period.

- At the beginning of every time period, after selecting jobs to serve, a holding cost of H^i for the unit remaining job $i \in \mathcal{I}$ in queue is accrued.
- There is a rejection cost G^i per job $i \in \mathcal{I}$ request which is rejected at every time period. The rejection will accrue only if the type $i \in \mathcal{I}$ jobs in queue reaches the maximum capacity W^i. This cost is calculated at the end of the time period.
- The decision-maker needs to decide how many jobs of each type to serve in each time-period. The goal is to maximize the total expected discounted profit over an infinite horizon. The discount factor is $0 < \alpha < 1$.

A weakly coupled MDP formulation and an approximate solution procedure based on Lagrangian relaxation for this problem was developed in [40]. Here, we consider an imperfect information variation, where the decision-maker does not know the success probabilities q^i. The decision- maker knows that these probabilities take one value each from the sets $Q^i = \{\theta^i_1, \ldots \theta^i_{K^i}\}$, for $i = 1, 2, \ldots, I$. The definition of the physical state and its evolution, actions, and rewards and costs are identical to the model in [40]. It is recalled below for completeness. For our imperfect information case, we also need an information state, which stores the probability mass function of the decision-maker's belief about the values of the success probabilities q^i. Specifically, we use information states $y^i_k = P(q^i = \theta^i_k)$, for $k = 1, 2, \ldots, K_i - 1$, and for $i = 1, 2, \ldots, I$; and the $K_i - 1$-dimensional information state is written simply as y^i. Moreover, $Y^i = [0, 1]^{K_i - 1}$ denotes the set of all possible information states for type-i. The set of all information states is denoted by $Y = \underset{i \in \mathcal{I}}{\times} Y^i$.

Following [40], the physical state of our MDP model is defined as $x = (x^1, x^2, \ldots, x^I)$, where x^i, for $i \in \mathcal{I}$, is the number of incomplete type i jobs in queue at the beginning of a time period. Let $X^i = \{0, 1, \ldots, W^i\}$ for $i \in \mathcal{I}$. The set X of all possible physical states is then defined as the Cartesian product $X = X^1 \times X^2 \times \ldots \times X^I$. The set $\mathcal{X}$ stores all possible physical and information states and is defined as $\mathcal{X} = X \times Y$. The decision vector is represented by $u = (u^1, u^2, ..., u^I)$, where u^i, for $i \in \mathcal{I}$, is the number of type-i jobs that we choose to serve in a time-period after observing the state (x, y). Let $U^i(x^i) = \{0, 1, \ldots, x^i\}$

and $U(x) = \underset{i\in\mathcal{I}}{\times} U^i(x^i)$. The set $\bar{U}(x) \subset U(x)$ of all feasible decision vectors in state x is defined by the resource constraints

$$\bar{U}(x) = \{(u^1, u^2, ..., u^I) \in U(x) : \sum_{i\in\mathcal{I}} a^{ij}u^i \leqslant b^j, j = 1, \ldots, J\}. \tag{3.12}$$

Also define $f(x, y, u) = \sum_{i\in I} f_i(x^i, y^i, u^i)$, where $f_i(x^i, y^i, u^i)$ is the expected reward given state (x^i, y^i) and action u^i. It is defined as

$$\begin{aligned} f_i(x^i, y^i, u^i) = \Bigg\{ & \alpha \sum_{k=1}^{K^i} y_k^i \theta_k^i u^i R^i - H^i(x^i - u^i) \\ & - \sum_{k=1}^{K^i} \sum_{n_i=0}^{N^i} \sum_{\eta^i=0}^{u^i} y_k^i p^i(n_i) \binom{u^i}{\eta^i} (\theta_k^i)^{\eta^i} (1-\theta_k^i)^{u^i-\eta^i} G^i(\max\{(x^i - \eta^i) + n_i - W^i, 0\}) \Bigg\}. \end{aligned} \tag{3.13}$$

Thus Bellman equations for all $(x, y) \in \mathcal{X}$ are given by

$$\begin{aligned} V(x,y) = \max_{u\in\bar{U}(x,y)} \Bigg\{ & \sum_{i\in I} f_i(x^i, y^i, u^i) + \\ & \alpha \sum_{k_1=1}^{K^1} \cdots \sum_{k_I=1}^{K^I} \sum_{n_1=0}^{N^1} \cdots \sum_{n_I=0}^{N^I} \sum_{\eta^1=0}^{u^1} \cdots \sum_{\eta^I=0}^{u^I} \Big(\prod_{i\in I} y_{k_i}^i p^i(n_i) \binom{u^i}{\eta^i} (\theta_{k_i}^i)^{\eta^i} (1-\theta_{k_i}^i)^{u^i-\eta^i} \Big) V(x', y') \Bigg\}. \end{aligned} \tag{3.14}$$

Here, $x^{i'} = \min\{(x^i - \eta^i) + n^i, W^i\}$, for $i \in \mathcal{I}$. Also, for all $i \in \mathcal{I}$ and $k = \{1, 2, \ldots, K^i - 1\}$, the information state is updated according to

$$y_k'^i = \frac{y_k^i \binom{u^i}{\eta^i} (\theta_k^i)^{\eta^i} (1-\theta_k^i)^{u^i-\eta^i}}{\sum_{l=1}^{K^i-1} y_l^i \binom{u^i}{\eta^i} (\theta_l^i)^{\eta^i} (1-\theta_l^i)^{u^i-\eta^i} + (1 - \sum_{l=1}^{K^i-1} y_l^i) \binom{u^i}{\eta^i} (\theta_{K^i}^i)^{\eta^i} (1-\theta_{K^i}^i)^{u^i-\eta^i}}. \tag{3.15}$$

Suppose either semi-stochastic CEC or Thompson calculates the success probabilities as $\tilde{q}^i$,

for $i \in \mathcal{I}$ in physical state x. Then the decision-maker approximately solves Bellman equation

$$V(x) = \max_{u \in \bar{U}(x)} \Bigg\{ \sum_{i \in I} f_i(x^i, u^i) + \\ \alpha \sum_{n_1=0}^{N^1} \cdots \sum_{n_I=0}^{N^I} \sum_{\eta^1=0}^{u^1} \cdots \sum_{\eta^I=0}^{u^I} \Big(\prod_{i \in I} p^i(n_i) \binom{u^i}{\eta^i} (\tilde{q}^i)^{\eta^i} (1-\tilde{q}^i)^{u^i - \eta^i} \Big) V(x') \Bigg\}, \tag{3.16}$$

using Lagrangian relaxation. Here, $x^{i'} = \min\{(x^i - \eta^i) + n^i, W^i\}$, for $i \in \mathcal{I}$ and,

$$f_i(x^i, u^i) = \big\{ \alpha u^i \tilde{q}^i R^i - H^i(x^i - u^i) \\ - \sum_{n_i=0}^{N^i} \sum_{\eta^i=0}^{u^i} p^i(n_i) \binom{u^i}{\eta^i} (\tilde{q}^i)^{\eta^i} (1-\tilde{q}^i)^{u^i - \eta^i} G^i(\max\{(x^i - \eta^i) + n_i - W^i, 0\}) \big\}.$$

The Lagrangian relaxation approach for this Bellman's equation is outlined in detail in [40] and we do not repeat it here. Numerical results for this problem are presented next.

3.4.3.1 Computational results for dynamic resource allocation

Similar to the previous chapters, we generated problem sets randomly and ran simulations to examine the performance of semi-stochastic CEC and Thompson sampling. The simulations were performed on a 3.1 GHz iMac desktop with 16 GB RAM and an Intel Core i7 chip running Mac OS X 10.9.3. All linear and integer programs were solved using CPLEX 12 from IBM via a MATLAB R2015b front-end.

We randomly generated problem instances as follows. The number of job-types for the problem sets in Tables 3.1 and 3.2 was set to $I = 6$ and $I = 12$, respectively. The maximum number of new arrivals in each period, which is denoted N^i for every i, was set to a random uniformly distributed integer in interval $[1, 4]$. The probability $p^i(m)$ for $1 \leqslant m \leqslant N^i$ was generated by normalizing N^i uniform $(0, 1)$ random variables. We assumed for simplicity that the set of possible success probability values, $Q^i = \{\theta^i_1, \ldots \theta^i_{K^i}\}$, is identical for all i. We set it to $Q^i = \{0.1, 0.2, \ldots, 0.9\}$. Thus, $K^i = 9$ for all i. The number of shared resources

was set to $J = 1$. The resource availability and resource consumption was generated similar to the approach in Chapter 1. The resource consumption for each job type is a random uniformly distributed integer value from set $[1, 2]$. The resource availability is the summation of resource consumption of all job types multiplied by the total number of job types I and the tightness ratio denoted by ρ in the first column of Tables 3.1 and 3.2. That is, after generating the resource consumption values, the resource availability is derived as $I \times \rho \times \sum_{i=1}^{I} a_{ij}$. Similar to Chapter 1, by changing the tightness ratio, we can alter the level of resource scarcity. The maximum capacity of the queue denoted by W^i for each i is set to a random uniformly distributed integer value from the interval $[1, 5]$. Similarly, for the reward R^i, rejection cost G^i, and holding cost H^i, a random uniformly distributed integer value from the intervals $[1, 100]$, $[1, 4]$, and $[1, 10]$ is sampled, respectively. The discount factor α is set to 0.99.

For every problem instance, we estimated the performance of different methods by averaging total discounted profit over 200 sample paths with 50 time-steps each. For examining the effect of learning with semi-stochastic CEC and Thompson sampling, we also implemented a third policy in our numerical experiments. We call this the "no-learning" policy, wherein the decision-maker does not update the initial belief and assumes that the transition probabilities are given by the expectation with respect to the initial belief as in (3.7). This yields a weakly coupled MDP with physical states only. Lagrangian relaxation is then employed to obtain an approximate value function.

For each row in Tables 3.1 and 3.2, we generated 30 test instances randomly. For each test instance we run the simulation using Thompson sampling, semi-stochastic CEC and no-learning. For each of the 200 independent sample path replications for each problem instance, we started with a uniform belief over Q^i, and a random physical state. The initial physical and belief states is the same for all three policies. Note that for every replication we have a discounted profit generated with three different policies over 50 time steps. So for every problem generated in every row we have a vector of size 200 which we can use to see whether semi-stochastic CEC or Thompson sampling performs statistically different as compared to no-learning. In Tables 3.1 and 3.2, the second and third columns report

the number of test instances out of 30 that semi-stochastic CEC and Thompson sampling is statistically different from no-leaning, respectively. To check this, we used t-test at the significance threshold of 0.05. For test instances where learning policies are statistically different from no-learning, we calculated the average discounted profit over 200 replications for the no-learning policy and divide that with the corresponding value using the learning policy. The average ratio over all test instances that the no-learning and learning policies are statistically different is reported in the last two columns of tables 3.1 and 3.2.

The tables show that there is an improvement in the profit generated by using semi-stochastic CEC and Thompson sampling as compared to no-learning. For this dynamic resource allocation problem, there is no significant difference between semi-stochastic CEC and Thompson sampling. The improvement achieved by the learning methods is less significant when resource availability is very low or very high. In the middle range of resource availability, the performance of learning policies improves when less resource is available. That is, the ratio of no-learning over learning decreases as tightness ratio decreases in the middle range. This is intuitive because when more resource is available, a simple heuristic could perform as well as a more sophisticated approach.

ρ	# significant Semi-CEC vs no-learning	# significant Thompson vs no-learning	No-learning over Semi-CEC	No-learning over Thompson
0.1	5	7	0.66	0.79
0.15	22	26	0.72	0.80
0.2	25	24	0.84	0.85
0.25	23	24	0.91	0.92
0.35	15	16	0.97	0.97
0.45	7	6	0.95	0.95

Table 3.1: Results of 6 problem sets. For each problem set, we generated 30 test instances. Total number of problems is 180. Number of jobs is 6.

ρ	# significant Semi-CEC vs no-learning	# significant Thompson vs no-learning	No-learning over Semi-CEC	No-learning over Thompson
0.05	5	6	0.89	0.92
0.08	28	27	0.83	0.85
0.1	29	29	0.82	0.84
0.13	26	24	0.92	0.93
0.15	21	18	0.97	0.96
0.2	7	6	0.98	0.98

Table 3.2: Results of 6 problem sets. For each problem set, we generated 30 test instances. Total number of problems is 180. Number of jobs is 12.

3.5 Partially observable weakly coupled MDP

Please refer to Krishnamurthy [63] for standard terminology on partially observable MDPs. We define a partially observable weakly coupled MDP as follows.

1. We have I subproblems, denoted by the set $\mathcal{I} = \{1, 2, \ldots, I\}$.

2. System state-space is $S = \underset{i \in \mathcal{I}}{\times} S_i$, where $S_i = \{1, 2, \ldots, n_i\}$ is the state-space for type-i.

3. Define $\vec{x}_i = \{x_i(1), x_i(2), \ldots, x_i(n_i)\}$, where $x_i(j)$ is the decision maker's probabilistic belief that system i is in state $j \in S_i$.

4. The belief state-space is given by $\Delta^i = \{\vec{x}^i \in \mathcal{R}_+^{n_i} | \sum_{j=1}^{n_i} x_i(j) = 1\}$.

5. The action-space for subproblem i is $A_i = \{1, 2, \ldots, m_i\}$. Let $A = \underset{i \in \mathcal{I}}{\times} A_i$.

6. Observation-space for subproblem i is $\mathcal{O}_i = \{1, 2, \ldots, O_i\}$. Let $\mathcal{O} = \underset{i \in \mathcal{I}}{\times} \mathcal{O}_i$.

7. State transition probabilities for subproblem i are $p_i = (s_i'|s_i, a_i)$, and $p(s'|s, a) = \prod_{i \in \mathcal{I}} p_i(s_i'|s_i, a_i)$.

8. Measurement outcome probabilities for subproblem i are $f_i(o_i|s_i, a_i)$. Let $f(o|s, a) = \prod_{i\in\mathcal{I}} f_i(o_i|s_i, a_i)$.

9. Rewards are given by $r(s'|a, s) = \sum_{i\in\mathcal{I}} r_i(s'_i|a_i, s_i)$

10. Coupling constraints: $\sum_{i\in\mathcal{I}} D_i(a_i) \leqslant K$.
    Here, $D_i(a_i)$ is a vector function of decisions a_i. Note that we do not allow the constraints to depend on the system state s_i, because this system state is not observable by the decision-maker.

11. Discount factor is $0 < \alpha < 1$.

Define $\phi_i(o'_i|\vec{x}_i, a_i)$ as the probability of observing o'_i given current belief state $\vec{x}_i$ and action a_i for subproblem i. We have,

$$\phi_i(o'_i|\vec{x}_i, a_i) = \sum_{s'_i\in S_i} f_i(o'_i|s'_i, a^i) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i)x_i(s_i). \tag{3.17}$$

Then define $\phi(o'|\vec{x}, a) = \prod_{i\in\mathcal{I}} \phi_i(o'_i|\vec{x}_i, a_i)$. Using Bayes Theorem, the decision-maker update belief vector for subproblem $i \in \mathcal{I}$, given current belief $\vec{x}_i$, action a_i and new observation o'_i. This update is defined by the formula

$$x'_i(s'_i) = \frac{f_i(o'_i|s'_i, a_i) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i)x_i(s_i)}{\sum_{s'_i\in S_i} f_i(o'_i|s'_i, a_i) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i)x_i(s_i)}, \ \forall s'_i \in S_i. \tag{3.18}$$

Then the Bellman equation for this partially observable weakly coupled MDP are given by

$$V(\vec{x}^1, \vec{x}^2, \ldots, \vec{x}^I) = \max_{\vec{a}\in A, \sum_{i\in\mathcal{I}} D_i(a_i)\leqslant K} \left\{ \sum_{i\in\mathcal{I}} \sum_{s_i\in S_i} x_i(s_i)r_i(s_i, a_i) + \right. \tag{3.19}$$
$$\left. \alpha \sum_{\vec{o}'\in\mathcal{O}} \prod_{i\in\mathcal{I}} \phi_i(o'_i|\vec{x}_i, a_i)V(\vec{x'}^1, \vec{x'}^2, \ldots, \vec{x'}^I) \right\}, \forall \vec{x}_i \in \Delta^i.$$

The next three subsections describe three algorithms for approximate solution of this partially observable weakly coupled MDP.

3.5.1 Semi-stochastic CEC

In semi-stochastic CEC, we calculate the expected value of the belief state $\vec{x}_i$ for each i. Let $\bar{s}_i$ be the system state closest to this expected value. We then apply Lagrangian relaxation to solve the following Bellman's equations for this "estimated" state

$$V(\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_I) = \max_{\vec{a}\in A, \sum_{i\in\mathcal{I}} D_i(a_i)\leqslant K} \Bigg\{ \sum_{i\in\mathcal{I}} r_i(\bar{s}_i, a_i) + \alpha \sum_{s'\in\mathcal{S}} \prod_{i\in\mathcal{I}} p_i(s'_i|\bar{s}_i, a_i) V(s'_1, s'_2, \ldots, s'_I) \Bigg\}. \tag{3.20}$$

3.5.2 Thompson sampling

In Thompson sampling, we sample a state $\hat{s}_i$ from the belief distribution $\vec{x^i}$, for each i.Now we can solve the following bellman equation. We then apply Lagrangian relaxation to solve the following Bellman's equations for this "estimated" state

$$V(\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_I) = \max_{\vec{a}\in A, \sum_{i\in\mathcal{I}} D_i(a_i)\leqslant K} \Bigg\{ \sum_{i\in\mathcal{I}} r_i(\hat{s}_i, a_i) + \alpha \sum_{s'\in\mathcal{S}} \prod_{i\in\mathcal{I}} p_i(s'_i|\hat{s}_i, a_i) V(s'_1, s'_2, \ldots, s'_I) \Bigg\}. \tag{3.21}$$

3.5.3 Discretization for partially observable weakly coupled MDP

Discretization simply means that we solve (3.19) by using a discretization $\vec{\sigma}_i$ of the continuous belief state $\vec{x}_i$, for each $i \in \mathcal{I}$. That is, the new Bellman equation is

$$V(\vec{\sigma}_1, \vec{\sigma}_2, \ldots, \vec{\sigma}_I) = \max_{\vec{a}\in A, \sum_{i\in\mathcal{I}} D_i(a_i)\leqslant K} \Bigg\{ \sum_{i\in\mathcal{I}} \sum_{s_i\in S_i} \sigma_i(s_i) r_i(s_i, a_i) + \alpha \sum_{\vec{o'}\in\mathcal{O}} \prod_{i\in\mathcal{I}} \phi_i(o'_i|\vec{\sigma}_i, a_i) V(\vec{\sigma'}_1, \vec{\sigma'}_2, \ldots, \vec{\sigma'}_I) \Bigg\}, \forall \sigma \in \bigtimes_{i\in\mathcal{I}} \mathcal{X}^{n_i}, \tag{3.22}$$

where $\vec{\sigma'}_i$ is the rounded future belief state.

3.5.4 Application to restless multi-armed bandit problems with partial state observations

In a restless multi-armed bandit problem [50, 63] there are I distinct types of projects. Projects of type-i have a state-space $S_i = \{1, 2, \ldots, n_i\}$. At every time-step, the decision-maker chooses K projects out of these I projects to work on for one period. A reward is collected for the active projects. The states of both active and inactive projects can evolve. There is no reward collected from inactive projects. More specific details are as follows.

1. Let a^i be a binary decision variable, which is 1 if project i is chosen and is 0 otherwise. The coupling constraint is $\sum_{i=1}^{I} a_i = K$.

2. The reward vector for each project i is $\vec{r_i} \in \mathcal{R}_+^{n_i}$. The discount factor is α.

3. The states evolve according to transition probabilities $p_i(s_i'|s_i, a_i)$.

4. States s_i are not observable. Instead, the decision-maker observes o_i. Define $\mathcal{O}_i = \{1, 2, \ldots, O_i\}$ as the set of of all possible values for o_i.

5. Define $f_i(o_i|s_i, a_i)$ as the observation outcome probability.

The Bellman equation for this problem is given by

$$V(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_I) = \max_{\vec{a}=\{0,1\}^I, \sum_{i\in\mathcal{I}} a_i = K} \left\{ \sum_{i\in\mathcal{I}} a_i \vec{x}_i . \vec{r}_i + \alpha \sum_{\vec{o'}\in\mathcal{O}} \prod_{i\in\mathcal{I}} \phi_i(o_i'|\vec{x}_i, a_i) V(\vec{x'}_1, \vec{x'}_2, \ldots, \vec{x'}_I) \right\}. \tag{3.23}$$

For semi-stochastic CEC and Thompson sampling, we first get an estimated $\tilde{s}_i$ from the information vector $\vec{x}_i$ as described above. Then we apply Lagrangian relaxation to approximately

solve

$$V(\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_I) = \max_{\vec{a}=\{0,1\}^I, \sum_{i\in\mathcal{I}} a_i = K} \left\{ \sum_{i\in\mathcal{I}} a_i \vec{r}_i(\tilde{s}_i) + \alpha \sum_{\vec{s'}\in S} \prod_{i\in\mathcal{I}} p_i(s'_i|\tilde{s}_i, a_i) V(s'_1, s'_2, \ldots, s'_I) \right\}. \tag{3.24}$$

The steps involved in this Lagrangian relaxation method are briefly described next.

3.5.4.1 Lagrangian relaxation within semi-stochastic CEC or Thompson sampling for restless multi-armed bandit

The relaxed counterpart of (3.24) is written as

$$V^\lambda(\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_I) = \max_{\vec{a}=\{0,1\}^I} \left\{ \sum_{i\in\mathcal{I}} a_i \vec{r}_i(\tilde{s}_i) + (\sum_{i\in\mathcal{I}} a_i - K)\lambda + \right.$$
$$\left. \alpha \sum_{\vec{s'}\in S} \prod_{i\in\mathcal{I}} p_i(s'_i|\tilde{s}_i, a_i) V^\lambda(s'_1, s'_2, \ldots, s'_I) \right\},$$

for any λ. As described in Section 3.2, we know that, for all $s \in S$,

$$V^\lambda(s) = \frac{1}{1-\alpha}\lambda K + \sum_{i=1}^{I} L_i^\lambda(s_i), \tag{3.25}$$

where

$$L_i^\lambda(s_i) = \max_{a_i=\{0,1\}} a_i \vec{r}_i(s_i) - \lambda a_i + \alpha \sum_{s'_i\in S_i} p_i(s'_i|s_i, a_i) L_i^\lambda(s'_i).$$

We need to solve the following LP to get the optimal λ, and value function approximations $L_i(s_i)$ for all $s_i \in S_i$ and $i \in \mathcal{I}$.

$$\min_{\lambda, L} \sum_{i\in\mathcal{I}} \sum_{s_i\in S_i} \beta_i L_i(s_i) + \lambda \frac{K}{1-\alpha}$$

$$\text{st: } L_i(s_i) \geqslant a_i \vec{r}_i(s_i) - \lambda a_i + \alpha \sum_{s_i' \in S_i} p_i(s_i'|s_i, a_i) L_i(s_i'), \quad \forall s_i \in S_i \text{ and } a_i = \{0,1\}. \tag{3.26}$$

Since this LP does not depend on the current information state x, we solve it only once to get λ^* and value function L^*. The problem that does need to be solved at every time-step is (3.24). For that purpose, we use $V(s') \approx \frac{K\lambda^*}{1-\alpha} + \sum_{i \in \mathcal{I}} L_i^*(s_i')$ as the approximate value function. This yields the following linear binary problem

$$\max_{\vec{a}=\{0,1\}^I, \sum_{i\in\mathcal{I}} a_i = k} \Bigg\{ \sum_{i \in \mathcal{I}} a_i \vec{r}_i(\tilde{s}^i) + \alpha \sum_{i \in \mathcal{I}} \sum_{\vec{s_i'} \in S_i} \left[a_i p_i(s_i'|\tilde{s}_i, a_i = 1) L_i^*(s'^i) + (1 - a_i) p_i(s_i'|\tilde{s}_i, a_i = 0) L_i^*(s'^i) \right] \Bigg\}.$$

The resulting optimal action is implemented, and the belief state is updated after making a new observation. This process then continues.

3.5.4.2 Discretization for restless multi-armed bandit

The relaxed Lagrangian Bellman equation counterpart of (3.22) is given by:

$$V^\lambda(\vec{\sigma}_1, \vec{\sigma}_2, \ldots, \vec{\sigma}_I) = \max_{\vec{a}\{0,1\}^I} \Bigg\{ \sum_{i \in \mathcal{I}} \sum_{s_i \in S_i} a_i \sigma_i(s_i) \vec{r}_i(s_i) + (\sum_{i \in \mathcal{I}} a_i - K)\lambda + \alpha \sum_{\vec{o'} \in \mathcal{O}} \prod_{i \in \mathcal{I}} \phi_i(o_i'|\vec{\sigma}_i, a_i) V^\lambda(\vec{\sigma'}_1, \vec{\sigma'}_2, \ldots, \vec{\sigma'}_I) \Bigg\}, \forall \vec{\sigma}_i \in \mathcal{X}^{n_i}.$$

The LP below needs to be solved to find the best λ and approximate value function $L_i(\vec{\sigma}_i)$.

$$\min_{\lambda, L} \sum_{i \in \mathcal{I}} \sum_{\vec{\sigma}_i \in \mathcal{X}^{n_i}} \beta_i L_i(\vec{\sigma}_i) + \lambda \frac{K}{1-\alpha}$$
$$L_i(\vec{\sigma}_i) \geqslant \sum_{s_i \in S_i} \vec{\sigma}_i(s_i) a_i \vec{r}_i(s_i) - \lambda a_i + \alpha \sum_{o_i' \in O_i} \phi_i(o_i'|\vec{\sigma}_i, a_i) L_i(\vec{\sigma'}_i), \quad \forall \vec{\sigma}_i \in \mathcal{X}^{n_i} \text{ and } a_i = \{0,1\}. \tag{3.27}$$

Problem (3.27) is only solved once since it does not depend on the current information state. The resulting λ^* and L^* are employed to approximately retrieve a decision. At every time-step, problem (3.23) needs to be solved by using $V(\vec{x'}_1, \vec{x'}_2, \ldots, \vec{x'}_I) \approx \frac{K\lambda^*}{1-\alpha} + \sum_{i\in\mathcal{I}} L_i^*(\vec{\sigma'}_i)$. Note that $\vec{\sigma'}_i$ is the rounded value of $\vec{x'}_i$. Specifically, the following decision retrieval problem is solved at every time-step given state $(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_I)$.

$$\max_{\vec{a}=\{0,1\}^I, \sum_{i\in\mathcal{I}} a_i = k} \Bigg\{ \sum_{i\in\mathcal{I}} a_i \vec{r}_i \vec{x}_i + \alpha \sum_{i\in\mathcal{I}} \sum_{o'_i\in O_i} [a_i \phi_i(o'_i|\vec{x}_i, a_i = 1) L_i^*(\vec{\sigma'}_i) + (1-a_i)\phi_i(o'_i|\vec{x}_i, a_i = 0) L_i^*(\vec{\sigma'}_i)] \Bigg\}.$$

In the above, for every $o'_i \in O_i$, the corresponding $\vec{\sigma'}_i$ is calculated using the update rule 3.18 and rounding.

The number of constraints in problem (3.27) grows exponentially in n_i so its solution is slow for large problems. In the computational results in Section 3.5.4.3 below, we tried the affine approximation and constraint generation approach, which was proposed for large-scale weakly coupled MDPS in [41], to solve (3.27). That is, similar to Chapter 1, we defined an affine approximation for $L_i(\vec{x_i})$ and used constraint generation to solve the resulting approximation of (3.27). The number of constraints then grows linearly in I and n_i. So (3.27) was computationally much faster to solve. The affine approximation was then used to retrieve decisions. The retrieval problem was a linear binary problem.

In the following the brief procedure for affine approximation and constraint generation is presented. First we assume the approximate value function $L_i(.)$ is from the set of affine functions $f : \mathbb{R}^{n_i} \to \mathbb{R}$. Assume there is a vector $\vec{x} \in \mathcal{X}^{n_i}$ or $\vec{x} \in \Delta^i$, then $L_i(\vec{x}) = d^i + \sum_{s_i\in S_i} x(s_i) c^i_{s_i}$. The optimal coefficients $d_i \in \mathbb{R}$ and $c^i \in \mathbb{R}^{n_i}$ can be found from solving (3.27). By using the affine approximation for L_i and also substituting update rule (3.18) and

observation probability definition (3.17) in (3.27) we will have:

$$\begin{aligned}
&\min_{\lambda,c,d} \sum_{i\in\mathcal{I}} \sum_{\vec{\sigma}_i\in\mathcal{X}^{n_i}} \beta_i[d^i + \sum_{s_i\in S_i} \sigma_i(s_i)c^i_{s_i}] + \lambda\frac{K}{1-\alpha} \\
&\text{st}: d^i + \sum_{s_i\in S_i} \sigma_i(s_i)c^i_{s_i} \geqslant \sum_{s_i\in S_i} \sigma_i(s_i)a_i\vec{r}_i(s_i) - \lambda a_i + \alpha \sum_{o'_i\in O_i} \phi_i(o'_i|\vec{\sigma}_i, a_i)d^i + \\
&\alpha \sum_{o'_i\in O_i} \sum_{s'_i\in S_i} c^i_{s_i}\Big(f_i(o'_i|s'_i, a_i) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i)\sigma_i(s_i)\Big), \quad \forall i \in \mathcal{I}, \quad \forall\vec{\sigma}_i \in \mathcal{X}^{n_i} \text{ and } a_i \in \{0, 1\}
\end{aligned} \tag{3.28}$$

Now we will apply constraints generation to (3.28). First we relax all the constraints in (3.28) and try to find initial set of constraints such that there is a feasible region. In the numerical implementation we randomly generated a vector $\vec{\sigma}_i \in \mathcal{X}^{n_i}$ and we added the corresponding constraints considering $a_i \in \{0, 1\}$. We repeat this process until an initial feasible region is found. Using the feasible region and the objective in (3.28), we can solve an optimization problem to get λ^*, d^* and c^*. Using λ^*, d^* and c^* we can solve a maximum violation problem to find the set of constraint among the constraints of (3.28) which is most violated by the current optimal values of λ^*, d^* and c^*. For every $i \in \mathcal{I}$ and every $a_i \in \{0, 1\}$ we solve the following maximum violation problem. Note that $\vec{\sigma}_i$ is a variable in the maximum violation problem.

$$\begin{aligned}
&\max_{\vec{\sigma}_i\in\mathcal{X}^{n_i}} \sum_{s_i\in S_i} \vec{\sigma}_i(s_i)a_i\vec{r}_i(s_i) - \lambda^* a_i + \alpha \sum_{o'_i\in O_i} \phi_i(o'_i|\vec{\sigma}_i, a_i)d^{*i} + \\
&\alpha \sum_{o'_i\in O_i} \sum_{s'_i\in S_i} c^{*i}_{s_i}\Big(f_i(o'_i|s'_i, a_i) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i)\sigma_i(s_i)\Big) - \Big(d^{*i} + \sum_{s_i\in S_i} \sigma_i(s_i)c^{*i}_{s_i}\Big)
\end{aligned}$$

For each $i \in \mathcal{I}$ and each $a_i \in \{0, 1\}$ the set of constraint which is violated the most will be calculated by the above problem. The optimal objective values for each $i \in \mathcal{I}$ and each $a_i \in \{0, 1\}$ is saved and then the maximum among all of them is chosen as the maximum violation and the corresponding constraint will be added to the initial set of constraints. The optimization problem using the new feasible region and objective function of (3.28) is

solved to get new coefficients. Using new coefficients we repeat the process of finding the maximum violated constraint. This is repeated until the stopping criteria which is discussed in [41] and chapter 1 is met. The stopping criteria is when the value $\frac{\mathcal{I}Z^*}{(1-\alpha H_c)}$ lies under some tolerance parameter. In the formulation Z^* is the recent maximum violation objective value and H_c is the optimal value of the optimization problem with the recent feasible region and objective function of (3.28). For every problem setting this iterative procedure is done only onces and at the end we have the affine value function approximation coefficients. The following decision retrieval at every state $(\vec{x}^1, \vec{x}^2, \ldots, \vec{x}^I)$ is solved. Note that the update rule (3.18) and observation probability definition (3.17) is used along with the affine value function approximation to drive the retrieval problem below.

$$\max_{\vec{a}=\{0,1\}^I, \sum\limits_{i\in\mathcal{I}} a_i=k} \left\{ \sum_{i\in\mathcal{I}} a_i \vec{r}_i \vec{x}_i + \alpha \sum_{i\in\mathcal{I}} \sum_{o'_i\in O_i} [a_i \phi_i(o'_i|\vec{x}_i, a_i=1) d^{*i} + (1-a_i)\phi_i(o'_i|\vec{x}_i, a_i=0) d^{*i}] + \right.$$
$$\alpha \sum_{i\in\mathcal{I}} \sum_{o'_i\in O_i} \sum_{s'_i\in S_i} \left[a_i c^{*i}_{s'_i} \Big(f_i(o'_i|s'_i, a_i=1) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i=1) x_i(s_i) \Big) + \right.$$
$$\left. \left. (1-a_i) c^{*i}_{s'_i} \Big(f_i(o'_i|s'_i, a_i=0) \sum_{s_i\in S_i} p_i(s'_i|s_i, a_i=0) x_i(s_i) \Big) \right] \right\}$$

The above linear binary problem will give us the optimal decision at every state.

3.5.4.3 Computational results for restless multi-armed bandit

Similar to previous sections, we randomly generated test instances of restless multi-armed bandit problems. The first column of Table 3.3 defines the four parameters that are set before generating these problem instances. The first one is I, the number of projects. We assume for simplicity that the number of possible project states is equal across all projects. That is, $n_i = N$, for all $i \in \mathcal{I}$. Thus, N is the second parameter we used for test problem generation. Recall that K equals the number of projects selected by the decision-maker at each time-step. This is the third parameter. We assume for simplicity that the number of possible observations $O_i = O$ for each i. Thus, O is the fourth parameter of our test instances. We

generated 30 problem instances for each one of 6 different (I, N, K, O) combinations. This produced 180 different problem instances.

The reward vector $\vec{r}_i$, which is of size N, stores the reward obtained by selecting project i in every possible state of project i. These reward numbers were set to equal random uniformly distributed integers from the interval $[1, 100]$. The transition probability p_i is generated randomly both for the case where the arm is chosen and not chosen. For every arm and for every chosen or not chosen case, the probability distribution of transiting from a given state to all others is a set of normalized random values between $(0, 1)$. The similar approach is taken for generating random outcome measurements f_i. For every arm and for every chosen or not chosen case, the probability distribution of observing the observable states $\{1, 2, \ldots, O\}$ from a given state s_i is a set of normalized random values between $(0, 1)$. The discount factor α is set to 0.99.

We ran simulations over 200 independent sample paths each with 50 times steps. We started every sample path with a uniform belief vector for the actual state of every project. We also started each path by randomly sampling a physical state for each project. Four policies were compared: semi-stochastic CEC, Thompson sampling, discretized, and affine-discretized. The first three are as discussed in sections 3.5.1, 3.5.2 and 3.5.3, respectively. The "affine-discretized" obtains a policy via discretization as discussed in 3.5.3, but by using an affine value function approximation along with constraint generation for the relaxed LP problem. The affine-discretized policy is computationally tractable for large problems whereas discretization (only) is not. For each problem instance, the performance of all policies is normalized with respect to the same restless multi-armed bandit problem but with perfect state observations. This latter is called "true-MDP." This true-MDP serves as an idealized "best-case" scenario since we expect the performance of various approximation methods to be worse in the partially observable case than with perfect observations. In particular, for every test instance, the average discounted profit over 200 sample paths using semi-stochastic CEC is divided by the corresponding value for the true-MDP. The average of all such values over 30 test instances is reported in the second column of 3.3. A similar

calculation was performed and is reported for Thompson sampling, discretization, and affine-discretization in columns three, four, and five, respectively. Note that for the case of $I = 10$, discretization (only) was not applied since it was computationally slow.

The table shows that Thompson sampling performed better than the other three policies in our numerical experiments.

Problem setting (I,N,K,O)	Semi-CEC over true-MDP	Thompson over true-MDP	Discretized over true-MDP	Affine-Discretized over true-MDP
(10,7,5,6)	0.60	0.77	-	0.63
(10,7,1,6)	0.56	0.85	-	0.56
(5,5,4,4)	0.67	0.89	0.72	0.71
(5,5,3,4)	0.69	0.89	0.72	0.71
(5,5,2,4)	0.74	0.94	0.71	0.77
(5,5,1,4)	0.66	0.90	0.69	0.71

Table 3.3: Results of 6 problem sets. For each problem set, we generated 30 test instances. Total number of problems is 180.

In the next chapter, an MDP model with imperfect information is employed to model a business-analytics problem. It is a multi-unit sequential-auction design problem, where the auctioneer does not know the parameters of the demand and bid value distributions. Although the problem is not weakly coupled, we show that solution methods similar to this chapter can still be applied for its approximate solution.

Chapter 4

LOT-SIZING IN SEQUENTIAL AUCTIONS WHILE LEARNING BID AND DEMAND DISTRIBUTIONS

The work reported here has appeared in Proceedings of the 2016 Winter Simulation Conference, 895-906, 2016.

4.1 Introduction

Sellers often use sequential, multi-unit, online auctions of identical items as a revenue generation and inventory clearing tool [21, 87, 88, 98, 110].

Lot-size is a key design variable in multi-unit auctions. A large lot-size reduces bidder competition and hence may reduce the clearing price. The total revenue might still be high because the number of units sold is large. A small lot-size increases bidder competition and hence the clearing price, but the total revenue might still be limited because just a few units are sold and a higher holding cost is incurred. Bidders in different auctions also compete indirectly through the opportunity cost of inventory. Owing to the uncertainty in the number of bidders and also in the bidders' posted bids, inventory evolves stochastically and hence optimal lot-sizes should vary dynamically. The cost of holding inventory also affects optimal lot-sizes. Finally, in auctions of fashion goods, of seasonal products, of new products, and where the seller is new to the market, the seller may not know *a priori* the distributions of the number of bidders and their bids, in their entirety [6, 10, 9, 37, 33]. The seller could then learn these sequentially by observing the number and values of posted bids. We emphasize here that the interaction between learning and decision-making in this problem is somewhat different from the standard multi-armed bandit problems. Here, the lot-size decision in an auction does not affect the number of bids and the values of the bids received in that auction.

Instead, a higher lot-size in one auction may correspond to a fewer total number of auctions and hence fewer opportunities to learn. Thus, the learning and decision-making aspects of the problem interact across auctions but not so much within an auction. We present a Bayes-adaptive MDP [64] model of this simultaneous learning and optimization problem.

Pinker et al. in [88] considered multi-unit, second-price sequential auctions with a fixed, deterministic constant number of bidders in each auction, and uniformly distributed bids. They were able to derive a closed-form formula for an optimal lot-size using dynamic programming. They also presented a Bayesian framework to learn the spread of the uniform bid distributions.

Tripathi et al. in [105] also assumed a fixed, deterministic constant number of bidders in each auction and uniformly distributed bids. They worked with a multi-unit Dutch mechanism. They assumed that the lot-size did not change across auctions and derived a simple closed-form formula for an optimal lot-size. They presented a goal programming method to learn bid distributions from online auction data.

Chen et al. in [21] generalized the models in Pinker et al. and Tripathi et al. in several ways. First, they allowed for a random number of bidders in each auction and arbitrary bid distributions. They also did not restrict their formulation to any specific auction mechanism. Instead, they provided a sufficient condition (that involved the expected revenue function and the probability mass function (pmf) of the number of bidders) under which a staircase with unit jumps policy was optimal for lot-sizes. According to this policy, if it is optimal to auction x units when the inventory on hand is i, then it is optimal either to auction x units or to auction $x + 1$ units when the inventory level is $i + 1$. Our clairvoyant MDP model in Section 4.2 below is similar to the model in Chen et al. barring minor idiosyncratic differences. Our research in this chapter extends that clairvoyant model to accommodate learning over a sequence of auctions.

There is a parallel body of literature on sequential auctions, where the seller does not announce lot-sizes, but instead uses reserve prices for releasing units. Examples include Vulcano et al. [110] and Vulcano and van Ryzin [98]. These papers did not incorporate

learning. Ghate in [37] recently incorporated demand learning in sequential, *single-unit*, second-price auctions where the seller optimizes her reserve price for this unit in each auction. Ghate assumed that the number of bidders in each auction is Poisson distributed with an unknown mean. The seller then uses a mixture-of-Gamma prior on this mean. This resulted in a high-dimensional Bayes-adaptive MDP, that was solved approximately by using semi-stochastic certainty equivalent control (CEC) and Q-function approximation.

Several papers in the inventory control and dynamic pricing areas have studied demand learning. Examples on the inventory control side include Scarf [100], Iglehart [55], Azoury [11], Lovejoy [73], Lariviere and Porteus [66]. For instance, Scarf generalized the classic inventory model of Arrow et al. [7] to accommodate demand distributions from an exponential family with one unknown parameter; other papers listed above studied variations of this learning problem. Examples on the dynamic pricing side include Araman and Caldentey [6], Aviv and Pazgal [9], and Farias and Van Roy [33]. For instance, Aviv and Pazgal [9] included demand learning in the Poisson intensity control model of Gallego and van Ryzin [36]. They assumed that the seller's prior belief about the Poisson rate parameter was Gamma, and used the CEC heuristic to approximately solve the resulting problem. Farias and Van Roy [33] revisited the Poisson intensity control problem, where the seller's prior belief on the Poisson rate parameter was a mixture-of-Gamma distributions. They proposed a new approximation approach called load-balancing, and showed that it outperformed CEC.

We next begin with a clairvoyant MDP model similar to Chen et al. [21], where the seller is assumed to know the demand and the bid distributions.

4.2 Clairvoyant MDP

Consider a seller who initially holds $I \geqslant 1$ units in her inventory. The seller uses a sequence of online, multi-unit auctions to clear inventory and generate revenue. A cost of $h \geqslant 0$ is incurred per unit held in inventory over the duration of an auction and is charged at the beginning of the auction. At the end of each auction, the inventory level drops by a quantity that either equals the lot-size (if the number of bids received is strictly larger than

the lot-size) or the number of bids received (if the number of bids is less than or equal to the lot-size). This process continues until all inventory is cleared. The discount factor over the duration of each auction is $0 < \delta < 1$. The seller's goal is to find a rule for lot-size decisions to maximize total discounted expected profit over all auctions.

We assume that the numbers of bidders across different auctions are independent and identically distributed (iid) random variables. Similarly, bids across different auctions are iid random variables. In addition, bids of different bidders in one auction are also iid random variables and are independent of the number of bids posted in that and in other auctions. These assumptions are standard in the aforementioned literature on sequential auctions.

The random number of bids in any auction is denoted by N. Let $p(n)$ denote the Poisson pmf of N with mean $\lambda > 0$. Moreover, for $n = 0, 1, \ldots$, we use $\bar{P}(n)$ to denote the probability that more than n bids are received.

A bid is denoted by the random variable Y, which takes values from the finite set of non-negative integers $\{0, 1, 2, \ldots, B\}$. Let $f_y \triangleq P(Y = y)$, for $y \in \{0, 1, 2, \ldots, B\}$, denote the categorical pmf of Y. We write it compactly as $\vec{f} \triangleq (f_0, f_1, \ldots, f_B)$.

Our first task is to obtain an expression for the expected revenue in a single auction with lot-size x. When $n \geqslant x + 1$ bids are posted, the seller's expected revenue in a second-price auction with x units equals the expected value of the $x + 1$st largest among n iid categorical random variables defined above. We denote this expected value of the $x + 1$st largest among n iid categorical random variables by $\psi(x; n)$.

Lemma 4.2.1. *The expected value of the $x + 1$st largest among n iid categorical random variables is given by*

$$\psi(x; n) \triangleq \sum_{y=0}^{B} \sum_{j=0}^{n-x-1} \binom{n}{j} (\eta(y))^j (1 - \eta(y))^{n-j}, \tag{4.1}$$

where $\eta(y) = P(Y < y)$, for $y = 0, 1, \ldots, B$.

Proof. For any positive integer $n \geqslant 1$, and $1 \leqslant k \leqslant n$, let $Y_{(k)}(n)$ denote the kth smallest

among n iid categorical random variables as defined above. Then, $E[Y_{(k)}(n)] = \sum_{y=0}^{B} P(Y_{(k)}(n) \geqslant y)$ by Problem 1.1 in Ross [95]. Note that $P(Y_{(k)}(n) \geqslant y)$, which represents the probability that the kth smallest among n iid random variables is at least y, is equal to the probability that at most $k-1$ among these random variables are strictly smaller than y. Thus, $P(Y_{(k)}(n) \geqslant y) = \sum_{j=0}^{k-1} \binom{n}{j} (\eta(y))^j (1-\eta(y))^{n-j}$. The result then follows by observing that the $x+1$st largest among n numbers is the $(n-x)$th smallest among these numbers. □

Thus, the seller's expected revenue with a Poisson distributed number of bids equals $\phi(x) \triangleq x \sum_{n=x+1}^{\infty} p(n)\psi(x;n)$. Here, we consider second-price auctions for concreteness; as in Chen et al. [21], other variations such as first-price auctions can also be handled by our approach by using the appropriate expected order statistic formula in (4.1).

We now model the seller's lot-size decision problem as a stationary, discounted MDP [91, 94]. The state of this MDP equals the inventory on hand. For such an MDP, a stationary policy is optimal. That is, it suffices to find, irrespective of the stochastic history of inventory evolution and of the auction number, the lot-size x_i in each inventory level $1 \leqslant i \leqslant I$. For $0 \leqslant i \leqslant I$, we let $V(i)$ denote the maximum total discounted expected profit earned from all future auctions starting with inventory i on hand. In dynamic programming parlance, function V is called the optimal value function.

For inventory levels $i = 0, 1, \ldots, I$, the optimal values $V(i)$ uniquely satisfy Bellman's equations given by

$$V(i) = \max_{x \in \{0,1,\ldots,i\}} \Big[-hi + \delta\phi(x) + \delta \sum_{n=0}^{x} p(n)V(i-n) + \delta\bar{P}(x)V(i-x) \Big]. \qquad (4.2)$$

Here, the right hand side of (4.2) includes three terms. The first term $-hi$ is the cost of holding i units in inventory. The second term, $\delta\phi(x)$, accounts for the seller's discounted expected revenue from a single auction with x units. The third term corresponds to the optimal discounted expected profit earned through all future auctions given that the seller announced a lot-size x in the current auction but only up to x bids were posted. The forth

term accounts for the optimal discounted expected profit generated through all remaining auctions given that the seller announced a lot-size x in the current auction and at least $x+1$ bids were posted. The lot-sizes x_i, for $i = 0, 1, \ldots, I$, that achieve the maxima in (4.2) define an optimal stationary policy [91, 94].

Value iteration [91, 94] is a standard algorithm for solving Bellman's equations and obtaining the corresponding optimal policy for MDPs. In our case, value iteration would proceed as follows. We start with $V^1(i) = 0$ for all $0 \leqslant i \leqslant I$, and use the update formula

$$V^{t+1}(i) = \max_{x \in \{0,1,\ldots,i\}} \left[-hi + \delta\phi(x) + \delta \sum_{n=0}^{x} p(n) V^t(i-n) + \delta \bar{P}(x) V^t(i-x) \right],$$

for $t = 1, 2, 3, \ldots$. We generate these updates until $\max\limits_{0 \leqslant i \leqslant I} |V^{t+1}(i) - V^t(i)|$ drops below a tolerance. From the general theory of MDPs [91, 94], we know that $\lim\limits_{t \to \infty} V^t(i) = V(i)$ for all $0 \leqslant i \leqslant I$. The function V^t thus provides an approximation of the optimal value function V and we obtain an approximate optimal policy by substituting V^t as a surrogate for V in the right hand side of (4.2) on termination.

4.3 Bayesian learning with Gamma and Dirichlet priors

The goal in this chapter is to study a variation of the above clairvoyant problem wherein the seller is uncertain about (i) the mean λ of the Poisson demand distribution, and (ii) the categorical pmf $\vec{f}$ of the bids. To emphasize the dependence of quantities such as $p(n)$, $\bar{P}(n)$, $\psi(x; n)$, and $\phi(x)$ from Section 4.2 on λ and $\vec{f}$, we denote them instead by $p_\lambda(n)$, $\bar{P}_\lambda(n)$, $\psi_{\vec{f}}(x; n)$, and $\phi_{\lambda,\vec{f}}(x)$, respectively, in the rest of this chapter. We similarly write the clairvoyant value function as $V_{\lambda,\vec{f}}$.

4.3.1 A Gamma prior on mean Poisson demand

We assume that (i) the seller views λ as a random variable, which we denote by Λ, and (ii) the seller has a Gamma prior belief about Λ. The shape and rate parameters of this prior are denoted by α, β, respectively. In particular, the seller believes *a priori* that the

probability density function $r(\lambda;\alpha,\beta)$ of Λ is given by $r(\lambda;\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}$. A benefit of this approach is that a Gamma prior is conjugate to Poisson. That is, the seller's posterior belief about the unknown mean of the Poisson number of bidders after observing the number of bids posted in one auction is also Gamma. Specifically, if $n \geqslant 0$ bids are posted in an auction, then the seller's belief about Λ is updated according to $\alpha^n \leftarrow \alpha + n$ and $\beta^n \leftarrow \beta + 1$. This conjugate property and the resulting straightforward parameter update considerably simplify the Bayes-adaptive MDP formulation of the seller's decision problem.

4.3.2 A Dirichlet prior on categorical bids

We assume that (i) the seller views the probabilities $\vec{f} = (f_0, f_1, \ldots, f_B)$ as random variables, and (ii) the seller has a Dirichlet prior belief with hyperparameters $\vec{a} \triangleq (a_0, a_1, \ldots, a_B) \in \Re_{++}^{B+1}$ about these random variables. Specifically, the prior probability density function of the categorical pmf $\vec{f}$ is given by $q(\vec{f};\vec{a}) \triangleq \frac{1}{\mathcal{B}(\vec{a})}\prod_{j=0}^{B}(f_j)^{a_j-1}$, where $\mathcal{B}(\vec{a})$ is the multinomial Beta function that is expressed in terms of Gamma functions as $\mathcal{B}(\vec{a}) \triangleq \frac{\prod_{j=0}^{B}\Gamma(a_j)}{\Gamma(\sum_{j=0}^{B}a_j)}$. Recall here that the Gamma function is defined by $\Gamma(z) = \int_0^\infty e^{-t}t^{z-1}dt$ [2]. Again, a benefit of this approach is that the Dirichlet prior is conjugate to categorical. That is, the seller's posterior belief about the unknown pmf of the categorical bid distribution after observing the bids posted in one auction remains Dirichlet. Specifically, if bids $\vec{y}^{\,n} \triangleq (y_1, y_2, \ldots, y_n)$ are posted in an auction, then the seller's belief about $\vec{f}$ is updated according to $a_j \leftarrow a_j + \sum_{k=1}^{n}\mathcal{I}_j(y_k),\ j = 0, 1, \ldots, B$. Here, $\mathcal{I}_j(y_k)$ is the indicator function that equals one when $y_k = j$ and equals zero otherwise. We compactly write the new Dirichlet hyperparameters as $\vec{a}(\vec{y}^{\,n})$. Again, this conjugate property and the resulting straightforward parameter update considerably simplify the Bayes-adaptive MDP formulation of the seller's decision problem.

4.3.3 Bayes-adaptive MDP formulation

We use $z \triangleq (\alpha, \beta) \in \Re^2_{++}$ and $\vec{a} \in \Re^{B+1}_{++}$ to denote the information states at the beginning of an auction. The seller's decision problem can now be formulated as a Bayes-adaptive MDP that is described next.

Given the information state $(z, \vec{a})$, the seller believes that her expected revenue from one auction with lot-size x is given by

$$\begin{aligned}\phi(x; z, \vec{a}) &\triangleq \int\int r(\lambda; z) q(\vec{f}; \vec{a}) \phi_{\lambda, \vec{f}}(x) d\lambda df_0 df_1 \dots df_B \\ &= \int\int r(\lambda; z) q(\vec{f}; \vec{a}) \left(\sum_{n=x+1}^{\infty} p_\lambda(n) \psi_{\vec{f}}(x; n) \right) d\lambda df_0 df_1 \dots df_B. \end{aligned} \tag{4.3}$$

Note that in the above double integrals, the first integral in over $\{\vec{f} \geqslant 0 : \sum_{j=0}^{B} f_j = 1\}$ and the second one is over $\lambda \geqslant 0$. Similarly, the seller believes that the probability that $n \geqslant 0$ bids will be posted is given by $p(n; z) \triangleq \int_{\lambda \geqslant 0} r(\lambda; z) p_\lambda(n) d\lambda$. Also, when the categorical pmf of bids is $\vec{f}$, let $\omega_{\vec{f}}(\vec{y}^{\,n})$ denote the probability that the posted bids will equal $\vec{y}^{\,n} = (y_1, y_2, \dots, y_n) \in \{0, 1, \dots, B\}^n$, given that $n \geqslant 1$ bids are posted. Here, $\{0, 1, \dots, B\}^n$ denotes the set of all permutations of n elements from the set $\{0, 1, \dots, B\}$. Thus, if $n \geqslant 1$ bids are posted, the seller believes that they will equal $\vec{y}^{\,n} \in \{0, 1, \dots, B\}^n$ with probability $\omega(\vec{y}^{\,n}; \vec{a}) \triangleq \int_{\{\vec{f} \geqslant 0 : \sum_{j=0}^{B} f_j = 1\}} \omega_{\vec{f}}(\vec{y}^{\,n}) q(\vec{f}; \vec{a}) df_0 df_1 \dots df_B$. We assume, as a matter of convention, that $\sum_{\vec{y}^{\,0} \in \{0,1,\dots,B\}^0} \omega(\vec{y}^{\,0}; \vec{a}) = 1$. For all $0 \leqslant i \leqslant I$, all $z \in \Re^2_{++}$, and all $\vec{a} \in \Re^{B+1}_{++}$, let the optimal value function $\mathcal{V}(i; z, \vec{a})$ denote the maximum total discounted expected profit earned from all future auctions when the inventory at the beginning of an auction is i and the information state is $z, \vec{a}$. We use z^n to denote the updated information state (α^n, β^n) as obtained from the information state $z = (\alpha, \beta)$, when $n \geqslant 0$ is the number of bids posted in an auction. Similarly, we use $\vec{a}(\vec{y}^{\,n})$ to denote the updated information state obtained from the information state $\vec{a}$, when bids $\vec{y}^{\,n}$ are posted in an auction. Then, Bellman's equations

for the seller's Bayes-adaptive MDP are given by

$$\mathcal{V}(i;z,\vec{a}) = \max_{x\in\{0,1,\ldots,i\}} \Big\{ -hi + \delta\phi(x;z,\vec{a}) + \delta \sum_{n=0}^{x} p(n;z) \sum_{\vec{y}^{\,n}\in\{0,1,\ldots,B\}^n} \omega(\vec{y}^{\,n};\vec{a})\mathcal{V}(i-n;z^n,\vec{a}(\vec{y}^{\,n})) + \delta \sum_{n=x+1}^{\infty} p(n;z) \sum_{\vec{y}^{\,n}\in\{0,1,\ldots,B\}^n} \omega(\vec{y}^{\,n};\vec{a})\mathcal{V}(i-x;z^n,\vec{a}(\vec{y}^{\,n}))\Big\}.e \tag{4.4}$$

Exact solution of this MDP is computationally intractable owing to the multi-dimensional state-space. We therefore investigate three methods for its approximate solution in the next section.

4.4 Algorithms for approximate solution of the Bayes-adaptive MDP

4.4.1 Semi-stochastic certainty equivalent control

Recall that in semi-stochastic certainty equivalent control (CEC), (some of) the random variables in a decision problem are replaced by their expected values [13]. In the current context, if the information state is $z, \vec{a}$, then the seller makes a lot-size decision by assuming that (i) the number of bidders is Poisson distributed with mean equal to the mean of the Gamma belief; and (ii) the bids are distributed according to a categorical distribution whose pmf is equal to the component means of the Dirichlet belief. We denote these means by $\lambda(z)$ and $\mu_j(\vec{a})$, for $j = 0, 1, \ldots, B$, respectively. Note that $\lambda(z) = \alpha/\beta$, and $\mu_j(\vec{a}) = a_j/(\sum_{j=0}^{B} a_j)$. We use the shorthand $\vec{\mu}(\vec{a}) \triangleq (\mu_0(\vec{a}), \mu_1(\vec{a}), \ldots, \mu_B(\vec{a}))$ to denote the vector of means. The inventory level and the information state then stochastically evolve to their new values and this process repeats until the end of all auctions. In particular, in state $(i; z, \vec{a})$, the lot-size is obtained by solving

$$\max_{x\in\{0,1,\ldots,i\}} -hi + \delta\phi_{\lambda(z),\vec{\mu}(\vec{a})}(x) + \delta \sum_{n=0}^{x} p_{\lambda(z)}(n) V_{\lambda(z),\vec{\mu}(\vec{a})}(i-n) + \delta \bar{P}_{\lambda(z)}(x) V_{\lambda(z),\vec{\mu}(\vec{a})}(i-x).$$

The certainty equivalent policy depends on the seller's beliefs only through their expectations $\lambda(z)$ and $\vec{\mu}(\vec{a})$. We therefore propose alternative approximation methods in the next two

sections.

4.4.2 Knowledge gradient algorithm

The knowledge gradient algorithm is often employed to balance the exploration-exploitation tradeoff in sequential information collection problems [35, 97]. Roughly speaking, the basic idea is to choose an alternative that would be optimal if the current period were the final opportunity for learning although profits would continue to accrue in future periods. In our case, this amounts to choosing lot-sizes in state $(i; z, \vec{a})$ by solving

$$\max_{x\in\{0,1,\ldots,i\}} -hi + \delta\phi(x; z, \vec{a}) + \delta \sum_{n=0}^{x} p(n; z) \sum_{\vec{y}^{\,n}\in\{0,1,\ldots,B\}^n} \omega(\vec{y}^{\,n}; \vec{a})\tilde{V}(i-n; z^n, \vec{a}(\vec{y}^{\,n}))$$
$$+ \delta \sum_{n=x+1}^{\infty} p(n; z) \sum_{\vec{y}^{\,n}\in\{0,1,\ldots,B\}^n} \omega(\vec{y}^{\,n}; \vec{a})\tilde{V}(i-x; z^n, \vec{a}(\vec{y}^{\,n})), \tag{4.5}$$

where

$$\tilde{V}(i; z^n, \vec{a}(\vec{y}^{\,n})) \triangleq \int\int V_{\lambda,\vec{f}}(i) r(\lambda; z^n) q(\vec{f}; \vec{a}(\vec{y}^{\,n})) d\lambda df_0 df_1 \ldots df_B, \tag{4.6}$$

for $0 \leqslant i \leqslant I$ and the first and second integrals are over $\{\vec{f} \geqslant 0 : \sum_{j=0}^{B} f_j = 1\}$ and $\lambda \geqslant 0$ respectively. In our implementation of this method, we estimate $\phi(x; z, \vec{a})$ via Monte Carlo simulation. Specifically, we sample K values of λ, and K values of the vector $\vec{f}$ independently according to density functions $r(\lambda; z)$ and $q(\vec{f}; \vec{a})$, respectively. We denote these values by $\hat{\lambda}_k$ and $\hat{\vec{f}}(k)$, for $k = 1, 2, \ldots, K$. We then estimate $\phi(x; z, \vec{a})$ using the sample average $\phi(x; z, \vec{a}) \approx \frac{\sum_{k=1}^{K} \phi_{\hat{\lambda}_k, \hat{\vec{f}}(k)}(x)}{K}$. The expectations in (4.5) are also estimated using sample average approximation as follows. For every $\hat{\lambda}_k$ and $\hat{\vec{f}}(k)$ we sample an n and a $\vec{y}^{\,n}$ from a Poisson and a categorical distribution, and then calculate the associated $(z^n, \vec{a}(\vec{y}^{\,n}))$. Using K values of $(z^n, \vec{a}(\vec{y}^{\,n}))$, we then use sample average to approximate the expectations in (4.5). Equation (4.6) is approximated using $\tilde{V}(i; z^n, \vec{a}(\vec{y}^{\,n})) \approx V_{\lambda(z^n),\vec{f}(\vec{a}(\vec{y}^{\,n}))}(i)$, where $\lambda(z^n)$ and $\vec{f}(\vec{a}(\vec{y}^{\,n}))$ are expectations of densities $r(\lambda; z^n)$ and $q(\vec{f}; \vec{a}(\vec{y}^{\,n}))$. Finally, these approximations are

employed in the sample average calculation. Then equation (4.5) will be approximated by

$$\max_{x\in\{0,1,\ldots,i\}} -hi+\delta\frac{\sum_{k=1}^{K}\phi_{\hat{\lambda}_k,\hat{\vec{f}}(k)}(x)}{K}+\delta\frac{\sum_{0\leqslant n_k\leqslant x}V_{\lambda(z^{n_k}),\vec{f}(\vec{a}(\vec{y}^{\,n_k}))}(i-n_k)+\sum_{n_k>x}V_{\lambda(z^{n_k}),\vec{f}(\vec{a}(\vec{y}^{\,n_k}))}(i-x)}{K}. \tag{4.7}$$

4.4.3 Thompson sampling

In the current context, Thompson sampling reduces to the following approach. When the inventory is i and the information state is $(z,\vec{a})$, the seller samples a λ according to the Gamma density $r(\lambda;z)$ and the bid pmf $\vec{f}$ according to the Dirichlet density $q(\vec{f};\vec{a})$. Let $\hat{\lambda}$ and $\hat{\vec{f}}$ denote these sampled values. The seller then chooses a lot-size that maximizes the profit assuming that the demand and bid distribution parameters are $\hat{\lambda}$ and $\hat{\vec{f}}$, respectively.

The computational effort required for the above three approximation methods is roughly as follows. For finding the optimal lot-size in every state, the Bellman equation (4.2) needs to be solved using value iteration when CEC and Thompson sampling approximations are applied. For knowledge gradient, the Bellman equation (4.2) needs to be solved K times in every state in contrast. This causes knowledge gradient to be slower than CEC and Thompson sampling. This behavior was also observed in our numerical results in the next section.

4.5 Computational results

We created different problem instances by changing the true λ (columns of our tables), and initial inventory I (rows). The discount factor was $\delta = 0.99$. The sample size K was set to 50 in all sample average approximations. Parameters of the initial Gamma prior were fixed at $\alpha = 5$ and $\beta = 1$. The initial Dirichlet parameters were set to $\vec{a} = (1, 1, \ldots, 1)$. The maximum bid value B was set to 430 and the holding cost was $h = 10$. The true categorical bid distribution was set to equal a discretized, truncated Weibull distribution. The scale parameter of this Weibull distribution was fixed at $B/2$. Its shape parameter was 2 for

Tables 4.1-4.4, and 4 for Tables 4.5-4.7. We refer to these two shapes as wide and narrow bid distributions, respectively. In each row of each table, we report averages over 50 independent simulations. Each simulation terminated when the inventory was completely depleted. Semi-stochastic CEC, knowledge gradient, and Thompson sampling need, as subroutines, value functions of MDPs where the mean demand and the bid pmf are fixed at various values. These were approximated by the value function of the staircase with unit jumps policy to significantly speed up our simulations. In all our simulations, the clairvoyant optimal policy is a hypothetical, ideal policy that is assumed to know the true λ and the true categorical bid distribution.

The numbers in Table 4.1 equals the profit made by a lot-sizing policy that does not update the seller's beliefs, reported as a percentage of the clairvoyant optimal profit. This "no learning" policy finds lot-sizes by solving Bellman's equations where the expected revenue and the demand pmf are calculated by taking expectations with respect to the initial Gamma and Dirichlet priors. In our approximate implementation of this process, these expectations were estimated as sample averages. The boldface numbers in Table 4.1 indicate that the profit made by the no learning policy was statistically different from the clairvoyant optimal profit (as inferred from a t-test at the significance threshold of 0.05).

The first number in each column of Tables 4.2, 4.3, and 4.4 equals the profits made by semi-stochastic CEC, knowledge gradient, and Thompson sampling, reported as a percentage of the clairvoyant optimal profit, respectively. The boldface numbers in these tables indicate that the profits made by these three learning methods were statistically different from those made by the no learning policy (again, as inferred from a t-test at the significance threshold of 0.05).

The second number in each column of Tables 4.2, 4.3, and 4.4 refers to the increase or decrease in profit compared to the no learning policy. This is reported as 0.00 when the learning and no learning policies are not statistically different; otherwise, it is reported by subtracting from the first number either the first number in Table 4.1 or 100 depending on whether or not the profit without learning is statistically different form the clairvoyant

optimal profit.

Table 4.1: The percentage of optimal profit reached with no learning for the wide bid distribution.

	λ			
I	5	10	15	20
20	104.31	**95.71**	**96.49**	**94.64**
25	99.09	**96.71**	**95.79**	**93.16**
30	96.08	**96.29**	**92.24**	**92.61**
35	96.38	**94.42**	**92.97**	**91.89**
40	96.99	**95.21**	**90.07**	**91.75**
45	97.60	**91.60**	**91.43**	**90.30**
50	91.54	**93.42**	**89.64**	**89.28**
55	89.52	**90.86**	**90.81**	**87.21**
60	112.86	**92.37**	**89.89**	**86.10**

Table 4.3: The percentage of optimal profit reached by knowledge gradient for the wide bid distribution.

	λ			
I	5	10	15	20
20	**102.04**,(2.04)	**98.03**,(2.32)	**98.08**,(1.59)	**98.15**,(3.51)
25	**97.44**,(-2.56)	97.74,(0.00)	**99.50**,(3.71)	**98.04**,(4.88)
30	**95.36**,(-4.64)	**98.17**,(1.88)	**97.90**,(5.66)	**97.64**,(5.03)
35	94.56,(0.00)	**96.05**,(1.63)	**97.92**,(4.95)	**97.79**,(5.9)
40	**93.15**,(-6.85)	**96.74**,(1.53)	**96.01**,(5.94)	**96.97**,(5.22)
45	**94.01**,(-5.99)	**94.68**,(3.08)	**96.47**,(5.04)	**96.88**,(6.58)
50	**85.45**,(-14.55)	**94.71**,(1.29)	**96.06**,(6.42)	**96.39**,(7.11)
55	90.21,(0.00)	**93.56**,(2.7)	**96.76**,(5.95)	**96.01**,(8.8)
60	**93.65**,(-6.35)	**93.97**,(1.6)	**95.39**,(5.5)	**95.87**,(9.77)

Table 4.2: The percentage of optimal profit reached by semi-stochastic CEC for the wide bid distribution.

	λ			
I	5	10	15	20
20	**102.41**,(2.41)	**98.15**,(2.44)	**98.17**,(1.68)	**98.39**,(3.75)
25	**96.68**,(-3.32)	**98.06**,(1.35)	**99.28**,(3.49)	**98.40**,(5.24)
30	95.92,(0.00)	**98.24** ,(1.95)	**98.03**,(5.79)	**97.77**,(5.16)
35	95.07,(0.00)	**96.28**,(1.86)	**97.33**,(4.36)	**97.28**,(5.39)
40	**92.92**,(-7.08)	**97.20**,(1.99)	**96.08**,(6.01)	**97.12**,(5.37)
45	**94.18**,(-5.82)	**94.76**,(3.16)	**96.62**,(5.19)	**96.99**,(6.69)
50	**86.52**,(-13.48)	**95.18**,(1.76)	**96.31**,(6.67)	**96.73**,(7.45)
55	90.74,(0.00)	**93.59**,(2.73)	**96.93**,(6.12)	**96.06**,(8.85)
60	**91.69**,(-8.31)	93.49,(0.00)	**95.24**,(5.35)	**95.66**,(9.56)

Table 4.4: The percentage of optimal profit reached by Thompson sampling for the wide bid distribution.

	λ			
I	5	10	15	20
20	**100.95**,(0.95)	**97.82**,(2.11)	**98.22**,(1.73)	**98.20**,(3.56)
25	97.37,(0.00)	**97.89**,(1.18)	**99.14**,(3.35)	**97.85**,(4.69)
30	96.08,(0.00)	**98.26**,(1.97)	**97.35**,(5.11)	**97.88**,(5.27)
35	95.41,(0.00)	**97.17**,(2.75)	**97.11**,(4.14)	**97.68**,(5.79)
40	94.11,(0.00)	**96.74**,(1.53)	**96.29**,(6.22)	**96.01**,(4.26)
45	94.45,(0.00)	**94.50**,(2.9)	**96.66**,(5.23)	**95.94**,(5.64)
50	**87.32**,(-12.68)	**95.28**,(1.86)	**96,.55**,(6.91)	**96.82**,(7.54)
55	85.16,(0.00)	**94.09**,(3.23)	**96.43**,(5.62)	**96.68**,(9.47)
60	**94.59**,(-5.41)	**95.48**,(3.11)	**95.49**,(5.6)	**96.15**,(10.05)

Recall from the first column of Table 4.1 that the profit made by the no learning policy is not significantly different from the clairvoyant optimal policy. This is intuitive because the mean α/β of the initial Gamma prior matches the true value of $\lambda = 5$. The profits in the first columns of Tables 4.2-4.4 are much more "noisy" — sometimes statistically worse than the no learning profit and sometimes better. This is perhaps to be expected because the

estimates of the mean demand as computed by the learning algorithms can oscillate around the true λ depending on the observed stochastic demands.

The last three columns from Table 4.1 suggest that the profit of the no learning policy is statistically worse than the clairvoyant optimal profit. This is intuitive since the true λ does not match the initial Gamma prior mean. In each of these last three columns of Table 4.1 the no learning policy roughly seems to make decreasing relative profits as the initial inventory I increases. This is perhaps because the temporal accumulation of error induced by the mean mismatch increases with increasing initial inventory. This error accumulation phenomenon is also observed in each of the last three columns of Tables 4.2, 4.3, and 4.4, where the percentage (first number in each column) of the optimal clairvoyant profit reached by the learning algorithms seems to decrease with increasing inventory. The last three columns in Tables 4.2, 4.3, and 4.4 show that the learning algorithms are able to improve (third number in every cell) upon the profit of the no learning policy. The range of this improvement seems to increase as the true λ drifts away from the mean of the initial Gamma prior. This value of active learning as a function of the mean mismatch also seems to increase with increasing inventory. Tables 4.2-4.4 suggest that there is no significant difference among the profits made by the three learning algorithms.

In Tables 4.1-4.4, the true bid distribution was similar to a centered Weibull. We also tried right- and left-skewed Weibull bids (not reported in this chapter for brevity), and were able to draw qualitatively similar conclusions.

Tables 4.5-4.7 report results of simulations similar to the previous tables, but this time with the narrow bid distribution. For this narrow bid distribution, we only tried semi-stochastic CEC and Thompson sampling. This decision was made based on our observation (from simulations and tables for the wide bid distribution) that the knowledge gradient algorithm was computationally slower but did not provide a statistically significant improvement in profit.

The profit reached by the no learning policy is generally smaller in Table 4.5 than in Table 4.1, for each column. This is because the narrow bid distribution is more different

from the flat Dirichlet prior that in the wide one. Even for the first column of Table 4.5, although the true λ matches the Gamma prior mean, the profit of the no learning policy can be statistically worse than the clairvoyant optimal profit. The learning algorithms are still noisy for the same reason as in Table 4.1. Consistent with these points, the value of active learning over no learning in the last three columns of Tables 4.6-4.7 seems to be generally larger than the corresponding tables for the wide bid distribution. Finally, other qualitative observations from the tables for the wide bid distribution also hold for the narrow bid distribution.

In summary, we conclude that active learning may provide higher profits as compared to not learning, especially when the initial priors are off. Among the three learning algorithms we implemented, semi-stochastic CEC and Thompson sampling were computationally faster and yet produced profits comparable to knowledge gradient.

Table 4.5: The percentage of optimal profit reached with no learning for the narrow bid distribution.

	λ			
I	5	10	15	20
20	**90.66**	**88.88**	**86.64**	**88.83**
25	**93.13**	**88.65**	**85.47**	**83.90**
30	**90.93**	**87.10**	**80.76**	**83.17**
35	98.13	**85.82**	**82.94**	**81.09**
40	**90.9**	**85.59**	**79.15**	**80.34**
45	94.57	**85.13**	**77.71**	**78.37**
50	**86.09**	**84.38**	**77.24**	**75.04**
55	94.57	**82.82**	**74.86**	**75.80**
60	91.45	**84.52**	**73.95**	**74.87**

Table 4.6: The percentage of optimal profit reached by semi-stochastic CEC for the narrow bid distribution.

	λ			
I	5	10	15	20
20	90.52,(0.00)	**93.58**,(4.69)	**93.44**,(6.8)	**94.29**,(5.45)
25	**92.02**,(-1.11)	**91.74**,(3.09)	**92.29**,(6.82)	**92.96**,(9.06)
30	90.21,(0.00)	**90.68**,(3.58)	**91.35**,(10.59)	**92.48**,(9.30)
35	**94.8**,(-5.2)	**90.70**,(4.88)	**90.92**,(7.98)	**91.45**,(10.36)
40	**88.35**,(-2.55)	**91.57**,(5.97)	**89.85**,(10.7)	**91.1**,(10.8)
45	**91.22**,(-8.78)	**90.14**,(5.01)	**89.12**,(11.41)	**90.75**,(12.38)
50	**80.42**,(-5.67)	**88.56**,(4.18)	**89.40**,(12.16)	**88.98**,(13.94)
55	94.64,(0.00)	**85.93**,(3.10)	**87.76**,(12.9)	**88.98**,(13.18)
60	**83.12**,(-16.88)	**86.91**,(2.39)	**87.55**,(13.6)	**87.71**,(12.84)

Table 4.7: The percentage of optimal profit reached by Thompson sampling for the narrow bid distribution.

	λ			
I	5	10	15	20
20	90.78,(0.00)	**93.89**,(5)	**93.67**,(7.03)	**94.31**,(5.48)
25	92.38,(0.00)	91.61,(2.96)	**92.56**,(7.09)	**92.91**,(9.01)
30	90.31,(0.00)	**90**,(2.9)	**91.24**,(10.48)	**92.89**,(9.72)
35	**94.93**,(-5.07)	**90.73**,(4.91)	**91.81**,(8.87)	**92.37**,(11.28)
40	**87.10**,(-3.8)	**90.19**,(4.59)	**90.25**,(11.1)	**89.76**,(9.41)
45	92.62,(0.00)	**89.95**,(4.82)	**89.64**,(11.93)	**90.58**,(12.21)
50	**81.09**,(-5)	**89.11**,(4.73)	**89.63**,(12.39)	**89.38**,(14.34)
55	93.51,(0.00)	**86.64**,(3.81)	**87.46**,(12.6)	**88.49**,(12.69)
60	**83.29**,(-16.71)	**87.32**,(2.8)	**88.45**,(14.5)	**87.36**,(12.49)

Chapter 5

SUMMARY AND FUTURE WORK

In this dissertation, MDP models for four classes of dynamic resource allocation problems were presented. The models in the first three chapters had the weakly coupled form. Chapter 1 demonstrated that a large class of advance scheduling problems can be modeled as weakly coupled MDPs. This enabled the application of a Lagrangian relaxation approach with affine value function approximation and constraint generation to approximately solve these problems. The numerical experiments showed that this approach outperformed a myopic heuristic by a statistically significant margin. There are two directions for future research that can extend the problem statement and model discussed in Chapter 1. One would be a combined advance and allocation scheduling model that books customers for future days, and also reactively adapts to stochastic events that occur during a day. The other direction can investigate patient preference.

Chapter 2 presented a weakly coupled MDP model for a dynamic resource-constrained project scheduling problem (DRCPSP). A simulation-based policy iteration method was implemented. The results suggested that this is a viable method for solving DRCPSPs. Future research could consider an extension where task-durations are stochastic.

Chapter 3 provided an approximate dynamic programing approach for weakly coupled MDPs with imperfect information. One possible future work is to study percentile optimization as it is discussed in [27] for weakly coupled MDP with unknown reward or transition probabilities. The theory and algorithm discussed in [27] can be applied to weakly coupled MDPs with uncertainty in reward or transition probabilities, and the hope would be to exploit the weakly coupled structure for computational benefits.

In Chapter 4, an MDP model with imperfect information was formulated for a business-

analytics problem. Approximation techniques similar to the ones in Chapter 3 are presented and implemented for this multi-unit sequential-auction design problem with unknown parameters of the demand and bid value distributions.

Moreover, it would be interesting to apply other approaches to handle parametric uncertainties in general MDPs for different applications. Examples include the robust optimization approach from [56, 81], or the data-driven method from [57]. One such application would be the optimal design of clinical trials. Clinical trials consist of dynamic decision making, and parameter uncertainty is a common issue. Mathematical modeling and specifically MDPs could be used in clinical trials, but as it is mentioned by Villar et al. [109], there is a gap in the literature in terms of applying theoretical models to real-world clinical trials.

BIBLIOGRAPHY

[1] D Aberdeen, S Thiebaux, and L Zhang. Decision-theoretic military operations planning. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling*, Whistler, British Columbia, Canada, 2004.

[2] M Abramowitz and I A Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover, New York, New York, USA, 1972.

[3] D Adelman and A J Mersereau. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research*, 56(3):712–727, 2008.

[4] Vishal Ahuja and John R Birge. Response-adaptive designs for clinical trials: Simultaneous learning from multiple patients. *European Journal of Operational Research*, 248(2):619–633, 2016.

[5] E Altman. Applications of markov decision processes in communication networks. In E Feinberg and A Shwartz, editors, *Handbook of Markov Decision Processes: Methods and Applications*, chapter 16, pages 489–536. Springer, 2002.

[6] V F Araman and R Caldentey. Dynamic pricing for nonperishable products with demand learning. *Operations Research*, 57(5):1169–1188, 2009.

[7] K J Arrow, T E Harris, and J Marschak. Optimal inventory policy. *Econometrica*, 19(3):250–272, 1951.

[8] D Astaraky and J Patrick. A simulation based approximate dynamic programming approach to multi-class, multi-resource surgical scheduling. *European Journal of Operational Research*, 245(1):309–319, 2015.

[9] Y Aviv and A Pazgal. Dynamic pricing of short life-cycle products through active learning. *Working paper available at* ***http://apps.olin.wustl.edu/faculty/aviv/papers/Pricing_MS_Dec_2005.pdf***, 2005.

[10] Y Aviv and A Pazgal. A partially observed markov decision process for dynamic pricing. *Management Science*, 51(9):1400–1416, 2005.

[11] K S Azoury. Bayes solution to dynamic inventory models under unknown demand distribution. *Management Science*, 31(9):1150–1160, 1985.

[12] M A Begen and M Queyranne. Appointment scheduling with discrete random durations. *Mathematics of Operations Research*, 36(2):240–257, 2011.

[13] D P Bertsekas. *Dynamic Programming and Optimal Control*, volume 1 and 2. Athena Scientific, Nashua, NH, USA, 2007.

[14] D Bertsimas and J Niño-Mora. Conservation laws, extended polymatroids and multi-armed bandit problems; a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

[15] J Blazewicz, J K Lenstra, and A H G Rinooy Kan. Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5(1):11–24, 1983.

[16] F Caro and J Gallien. Dynamic assortment with demand learning for seasonal consumer goods. *Management Science*, 53(2):276–292, 2007.

[17] R G Casey, M R Qinlan, R Flynn, R Granger, T E D McDermott, and J. A. Thornhill. Urology outpatient non-attendance: Are we wasting our time? *Irish J Med Sci*, 176(4):305–308, 2007.

[18] T Cayirli and E Veral. Outpatient scheduling in health care: A review of literature. *Production and Operations Management*, 12:519–549, 2003.

[19] T Cayirli, K K Yang, and S A Quek. A universal appointment rule in the presence of no-shows and walk-ins. *Production and Operations Management*, 21(4):682–697, 2012.

[20] S Chakraborty, K Muthuraman, and M Lawley. Sequential clinical scheduling with patient no-shows and general service time distributions. *IIE Transactions*, 42:354–366, 2010.

[21] X Chen, A Ghate, and A K Tripathi. Dynamic lot-sizing in sequential online retail auctions. *European Journal of Operational Research*, 215(1):257–267, 2011.

[22] J Choi, M J Realff, and J H Lee. Dynamic programming in a heuristically confined state space: a stochastic resource-constrained project scheduling application. *Computers & Chemical Engineering*, 28(6):1039–1058, 2004.

[23] J Choi, M J Realff, and J H Lee. A q-learning-based method applied to stochastic resource constrained project scheduling with new project arrivals. *International Journal of Robust and Nonlinear Control*, 17(13):1214–1231, 2007.

[24] P C Chu and J E Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(3):63–86, 1998.

[25] L Corfield, A Schizas, A Williams, and A Noorani. Non-attendance at the colorectal clinic: A prospective audit. *Ann R Coll Surg Engl*, 90(5):377–380, 2008.

[26] J Daggy, M Lawley, D Willis, D Thayer, C Suelzer, P-C DeLaurentis, A Turkan, S Chakraborty, and L Sands. Using no-show modeling to improve clinic performance. *Health Informatics*, 16:246–259, 2010.

[27] E Delage and S Mannor. Percentile optimization for markov decision processes with parameter uncertainty. *Operations research*, 58(1):203–213, 2010.

[28] R A Deyo and T S Inui. Dropouts and broken appointments: A literature review and agenda for future research. *Med Care*, 11(1146-1157), 18.

[29] D A Dolgov and E H Durfee. Optimal resource allocation and policy formulation in loosely-coupled markov decision processes. In *ICAPS*, pages 315–324, 2004.

[30] A Erdelyi and H Topaloglu. Approximate dynamic programming for dynamic capacity allocation with multiple priority levels. *IIE Transactions*, 43:129–142, 2010.

[31] A Erdelyi and H Topaloglu. A dynamic programming decomposition method for making overbooking decisions over an airline network. *Informs Journal of Computing*, 22:443–456, 2010.

[32] S A Erdogan and B T Denton. Surgery planning and scheduling: A literature review. In J J Cochran, editor, *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley and Sons, 2011.

[33] V F Farias and B Van Roy. Dynamic pricing with a prior on market response. *Operations Research*, 58(1):16–29, 2010.

[34] J B Feldman, N Liu, H Topaloglu, and S Ziya. Appointment scheduling under patient preference and no-show behavior. `http://www.columbia.edu/~nl2320/doc/paper_FINAL.pdf`, 2014.

[35] P I Frazier, W B Powell, and S Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[36] G Gallego and G Van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science*, 40(8):999–1020, 1994.

[37] A Ghate. Optimal minimum bids and inventory scrapping in sequential, single-unit, vickrey auctions with demand learning. *European Journal of Operational Research*, 245(2):555–570, 2015.

[38] J Gittins. A dynamic allocation index for the sequential design of experiments. *Progress in statistics*, pages 241–266, 1974.

[39] Y Gocgun. *Approximate dynamic programming for dynamic stochastic resource allocation with applications to healthcare*. PhD thesis, University of Washington, Seattle, WA, USA, 2010.

[40] Y Gocgun and A Ghate. A lagrangian approach to dynamic resource allocation. In B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yucesan, editors, *Proceedings of the Winter Simulation Conference*, pages 3330–3338, Baltimore, 2010.

[41] Y Gocgun and A Ghate. Lagrangian relaxation and constraint generation for allocation and advance scheduling. *Computers & Operations Research*, 39(10):2323–2336, 2012.

[42] Y Gocgun and M L Puterman. Dynamic scheduling with due dates and time windows: an application to chemotherapy patient appointment booking. *Health Care Manag Sci*, 2013.

[43] L V Green and S Savin. Reducing delays for medical appointments: A queueing approach. *Operations Research*, 56(6):1526–1538, 2008.

[44] D Gupta and B Denton. Appointment scheduling in health care: Challenges and opportunities. *IIE Transactions*, 40:800–819, 2008.

[45] A Y Ha. Inventory rationing in a make-to-stock production system with several demand classes and lost sales. *Management Science*, 43(8):1093–1103, 1997.

[46] A Y Ha. Optimal dynamic scheduling policy for a make-to-stock production system. *Operations Research*, 45(1):42–53, 1997.

[47] W Hamilton, A Round, and D J Sharp. Patient, hospital, and general practitioner characteristics associated with nonattendance: a cohort study. *Br J Gen Pract*, 52:317–319, 2002.

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

[49] R Hassin and S Mendel. Scheduling arrivals to queues: a single sever model with no-shows. *Management Science*, 54(3):565–572, 2008.

[50] J Hawkins. *A Lagrangian decomposition approach to weakly coupled dynamic optimization problems and its applications*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2003.

[51] W L Herring and J W Herrmann. A stochastic dynamic program for the single-day surgery scheduling problem. *IIE Transactions on Healthcare Systems Engineering*, 1:213–225, 2011.

[52] W Herroelen, B De Reyck, and E Demeulemeester. Resource-constrained project scheduling: A survey of recent developments. *Computers & Operations Research*, 25(4):279–302, 1998.

[53] A L Hixon, R W Chapman, and J Nuovo. Failure to keep clinic appointments: Implications for residency education and productivity. *Fam Med*, 31(9):627–630, 1999.

[54] Y Huang and P Zuniga. Dynamic overbooking scheduling system to improve patient access. *Journal of operational research society*, 63:810–820, 2012.

[55] D L Iglehart. The dynamic inventory problem with unknown demand distribution. *Management Science*, 10(3):429–440, 1964.

[56] G N Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[57] H Jiang and U V Shanbhag. Data-driven schemes for resolving misspecified mdps:

asymptotics and error analysis. In *2015 Winter Simulation Conference (WSC)*, pages 3801–3812. IEEE, 2015.

[58] G C Kaandorp and G Koole. Optimal outpatient appointment scheduling. *Healthcare Management Science*, 10:217–229, 2007.

[59] D Kalathil, N Nayyar, and R Jain. Decentralized learning for multiplayer multiarmed bandits. *IEEE Transactions on Information Theory*, 60(4):2331–2345, 2014.

[60] S Kim and R E Giachetti. A stochastic mathematical appointment overbooking model for healthcare providers to improve profits. *IEEE transactions*, 36(6):1211–1219, 2006.

[61] K J Klassen and R Yoogalingam. Improving performance in out-patient appointment services with a simulation optimization approach. *Production and Operations Management*, 18(4):447–458, 2009.

[62] R Kopach, P-C DeLaurentis, M Lawley, K Muthuraman, L Ozsen, R Rardin, H Wan, P Intrevado, X Qu, and D Willis. Effects of clinical characteristics on successful open-access scheduling. *Health Care Management Science*, 10:111–124, 2007.

[63] V Krishnamurthy. *Partially Observed Markov Decision Processes*. Cambridge University Press, 2016.

[64] P R Kumar. A survey of some results in stochastic adaptive control. *SIAM Journal on Control and Optimization*, 23(3):329–380, 1985.

[65] L R LaGanga and S R Lawrence. Appointment overbooking in health care clinics to improve patient service and clinic performance. *Production and Operations Management*, 21:874–888, 2012.

[66] M A Lariviere and E L Porteus. Stalking information: Bayesian inventory management with unobserved lost sales. *Management Science*, 4(3):346–363, 1999.

[67] V J Lee, A Earnest, M I Chen, and B Krishan. Prediction of failed attendances in a multi-specialty outpatient center using electronic databases. *BMC Health Serv Res*, 2005.

[68] J YT Leung. *Handbook of scheduling: algorithms, models, and performance analysis*. CRC Press, 2004.

[69] J Li, D E Blumenfeld, N Huang, and J M Alden. Throughput analysis of production systems: recent advances and future topics. *International Journal of Production Research*, 47(14):3823–3851, 2009.

[70] H Liu, K Liu, and Q Zhao. Learning in a changing world: Restless multiarmed bandit with unknown dynamics. *IEEE Transactions on Information Theory*, 59(3):1902–1916, 2013.

[71] N Liu and S Ziya. Panel size and overbooking decisions for appointment-based services under patient no-shows. *Production and Operations Management*, 0:1–15, 2014.

[72] N Liu, S Ziya, and V G Kulkarni. Dynamic scheduling of outpatient appointments under patient no-shows and cancellation. *Manufacturing and Service Operations Management*, 12(2):347–364, 2010.

[73] W S Lovejoy. Myopic policies for some inventory models with uncertain demand distributions. *Management Science*, 36(6):724–738, 1990.

[74] J Luo, V G Kulkarni, and S Ziya. Appointment scheduling under patient no-shows and service interruptions. *Manufacturing and service operations management*, 14:670–684, 2012.

[75] J Magerlein and J Martin. Surgical demand scheduling: A review. *Health Services Research*, 13:418–433, 1978.

[76] P Melchiors. *Dynamic and Stochastic Multi-Project Planning*, volume 673 of *Lecture Notes in Economics and Mathematical Systems*. Springer, Switzerland, 2015.

[77] R Meshram, D Manjunath, and A Gopalan. On the whittle index for restless multi-armed hidden markov bandits. *arXiv preprint arXiv:1603.04739*, 2016.

[78] M Murray and C Tantau. Redefining open access to primary care. *Managed Care Quarterly*, 7:45–51, 1999.

[79] K Muthuraman and M Lawley. A stochastic overbooking model for outpatient clinical scheduling with no-shows. *IIE Transactions*, 40:820–837, 2008.

[80] D M Negoescu, K Bimpikis, M L Brandeau, and D A Iancu. Dynamic learning of patient response types: An application to treating chronic diseases. *Management Science*, 2017.

[81] A Nilim and L El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[82] J Patrick. A markov decision model for determining optimal outpatient scheduling. *Health Care Management Science*, 15:91–102, 2012.

[83] J Patrick, M L Puterman, and M Queyranne. Dynamic multi-priority patient scheduling for a diagnostic resource. *Operations Research*, 56(6):1507–1525, 2008.

[84] J H Patterson. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. *Management Science*, 30(7):854–867, 1984.

[85] A Pavitsos and E G Kyriakidis. Markov decision models for the optimal maintenance of a production unit with an upstream buffer. *Computers & Operations Research*, 36(6):1993 – 2006, 2009.

[86] M L Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer Science & Business Media, New York, NY, USA, 2012.

[87] E Pinker, A Seidmann, and Y Vakrat. Managing online auctions: current business and research issues. *Management Science*, 49(11):1457–1484, 2003.

[88] E Pinker, A Seidmann, and Y Vakrat. Using bid data for the management of sequential, multi-unit, online auctions with uniformly distributed bidder valuations. *European Journal of Operational Research*, 202:574–583, 2010.

[89] W B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality of dimensionality*. John Wiley and Sons, Hoboken, New Jersey, USA, 2007.

[90] A A B Pritsker, L J Watters, and P M Wolfe. Multiproject scheduling with limited resources: A zero-one programming approach. *Management Science*, 16(1):93–108, 1968.

[91] M Puterman. *Markov Decision Processes*. John Wiley and Sons, New Jersey, 1994.

[92] L Robinson and R Chen. Traditional and open-access appointment scheduling policies: the effects of patient no-shows. *Manufacturing and Service Operations Management*, 12:330–346, 2009.

[93] K W Ross and D Tsang. The stochastic knapsack problem. *IEEE Transactions on Communications*, 37(7):740–747, 1989.

[94] S M Ross. *Introduction to stochastic dynamic programming*. Academic Press, New York, 1983.

[95] S M Ross. *Stochastic Processes*. Wiley, New York, New York, USA, 1996.

[96] C T Rust, N H Gallups, W S Clark, D S Jones, and W D Wilcox. Patient appointment failures in pediatric resident continuity clinics. *Arch Pediatr Adolesc Med*, 149(6):693–695, 1995.

[97] I O Ryzhov, W B Powell, and P I Frazier. The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195, 2012.

[98] G Van Ryzin and G Vulcano. Optimal auctioning and ordering in an infinite horizon inventory-pricing system. *Operations Research*, 52(3):347–367, 2004.

[99] A Saure, J Patrick, S Tyldesley, and M L Puterman. Dynamic multi-appointment patient scheduling for radiation therapy. *European Journal of Operational Research*, 223:573–548, 2012.

[100] H Scarf. Bayes solutions of the statistical inventory problems. *Annals of Mathematical Statistics*, 30(2):490–508, 1959.

[101] M Strens. A bayesian framework for reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

[102] K T Talluri and G J Van Ryzin. *The Theory and Practice of Revenue Management*. Springer, New York, 2005.

[103] W R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

[104] T Tolio. *Design of Flexible Production Systems – Methodologies and Tools*. Springer, Berlin, Germany, 2009.

[105] A K Tripathi, S K Nair, and G Karuga. Optimal lot sizing policies for sequential online auctions. *IEEE Transactions of Knowledge and Data Engineering*, 21(4):554–567, 2009.

[106] V-A Truong. Optimal advanced scheduling with expediting. `http://www.columbia.edu/~vt2196/AdvanceSchedulingMSRevision2.pdf`, 2014.

[107] S Vakili, K Liu, and Q Zhao. Deterministic sequencing of exploration and exploitation for multi-armed bandit problems. *IEEE Journal of Selected Topics in Signal Processing*, 7(5):759–767, 2013.

[108] D Vengerov. A reinforcement learning approach to dynamic resource allocation. *Engineering Applications of Artificial Intelligence*, 20(3):383–390, 2007.

[109] Sofía S Villar, Jack Bowden, and James Wason. Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 30(2):199, 2015.

[110] G Vulcano, G Van Ryzin, and C Maglaras. Optimal dynamic auctions for revenue management. *Management Science*, 48(11):1388–1407, 2002.

[111] J Walraevens, B Steyaert, and H Bruneel. Performance analysis of a single-server ATM queue with a priority scheduling. *Computers & Operations Research*, 30(12):1807–1829, 2003.

[112] J Warden. 4.5 million oupatients miss appointments. *Br Med J*, 310:1158, 1995.

[113] C Watkins and P Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[114] P Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, 25(A):287–298, 1988.

[115] C Zacharias and M Pinedo. Appointment scheduling with no-shows and overbooking. *Production and Operations Management*, 23(5):788–801, 2014.

[116] B Zeng, A Turkcan, J Lin, and M Lawley. Clinic scheduling models with overbooking for patients with heterogeneous no-show probabilities. *Annals of Operations Research*, 178:121–144, 2010.