Networks of Sensors

Onderwater, M.

2016

document version
Publisher's PDF, also known as Version of record

Link to publication in VU Research Portal

citation for published version (APA)
Onderwater, M. (2016). *Networks of Sensors: Operation and Control*.

Part II

Markov Decision Processes

5

On the Control of a Queueing System with Aging State Information

In previous chapters we considered various aspects of sensor networks and the data produced by these networks. The following chapters deal with *Markov Decision Processes* (MDPs), a popular modelling framework for scenarios involving sequential decision making under uncertainty. For the current chapter we investigate control of a queueing system in which a component of the state space is subject to aging. The controller can choose to forward incoming queries to the system (where it needs time for processing), or respond with a previously generated response (incurring a penalty for not providing a fresh value). Hence, the controller faces a trade-off between data freshness and response times.

A similar trade-off occurs in the context of wireless sensor networks. Consider, e.g., a scenario where sensor nodes periodically report measurements to a centralized middleware component, which then saves these reports into, e.g., a database. When an application needs a measurement from a sensor, it sends a query to the middleware component. The component (having the role of controller) can then decide to either fetch a new measurement from the network, or to return a previously generated measurement from the database. Obtaining fresh measurements from the wireless sensor network is time-consuming due to the wireless transmissions across the network. On the other hand, the latest value in the database might be too old for practical purposes. The middleware component thus faces a trade-off that is similar to the one discussed in this chapter. In practice, middleware components typically use a simple threshold policy, where the network is used when the age of the last reported value exceeds some specified threshold.

For the current chapter we consider a scenario involving a more general queueing system, as an illustration of how a simple threshold policy can be improved by taking the load of the network into account. We model the system as a Markov Decision Process that turns out to be complex, then simplify it, and construct a control policy. This policy has near-optimal performance and achieves lower costs than both a threshold policy and a myopic policy.

This chapter is based on the results presented in [6] and [2].

5.1 Introduction

We illustrate the system in Figure 5.1, where a controller *Ctrl* handles incoming queries that require a response. The controller uses a policy to determine whether a query receives a response with fresh data, or with aged data. In the first case, the query is forwarded to a queue Q_1 where the query is eventually serviced. In the second case, the query is immediately answered with a known, aged, response that is stored in, e.g., a database (DB). The DB is regularly refreshed by reports from a queue Q_2. For modelling purposes we assume that both queries and report requests arrive according to a homogeneous Poisson process with rate λ_1 and λ_2, respectively. Also, we assume that the processing time in the queues is exponentially distributed with parameter μ_1 (for queries) and μ_2 (for report requests).

An example of the interaction between queries, reports, and the age of the latest value in the DB is shown in Figure 5.2. At time 1 a job is completed at the server of Q_2 (resulting in a report) and sets the age to 0. This age then increases linearly until the next report is generated at time 4. Meanwhile, at time 1.5 a query arrives at the controller, at which moment the age of the latest value in the database is 0.5. Then at time 3 the second query arrives, which sees the most recent value in the database at age 2. Query 3 arrives after the second report is generated, at which point the value in the database has age 1. Report 3 at time 6 refreshes the database again and sets the age to 0. Note that the graph does not show which decisions the controller takes on arrival of a query.

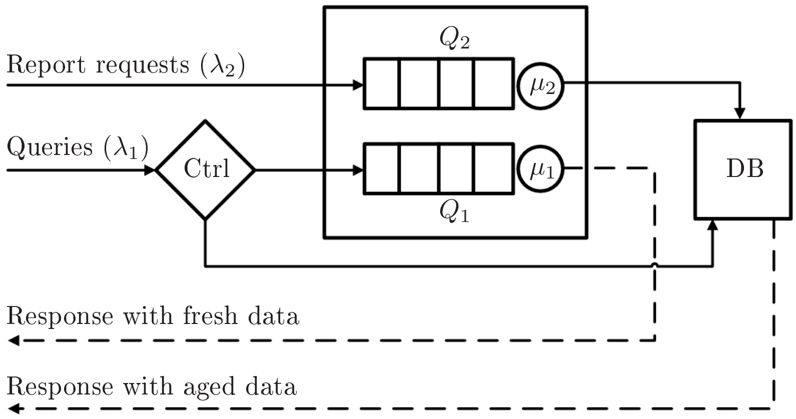

FIGURE 5.1: The controller (Ctrl) assigns incoming queries to either Q_1 in the queueing system, or to the DB. In the first case, the query gets a fresh response, but has to wait some time before it is generated. In the second situation, the systems returns a previously generated (and thus aged) response immediately. The DB is regularly refreshed with fresh values (reports) from Q_2.

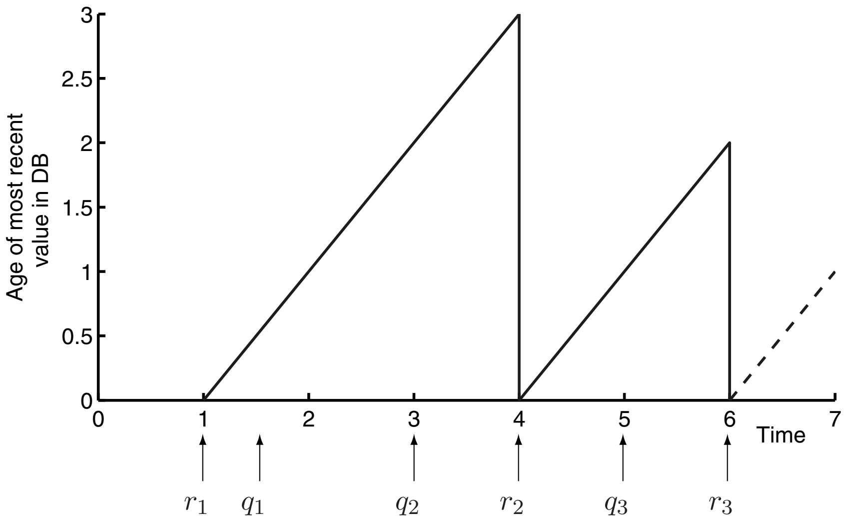

FIGURE 5.2: Interaction between queries $q_1, \ldots, q_3$, reports $r_1, \ldots, r_3$, and the age of the latest value in the database. In the graph, three reports arrive (at times 1, 4, and 6) that reset the age to 0. In between reports, the age increases linearly with time. Upon a query arrival, the controller sees the latest value in the database at a certain age and uses that age to take its decisions. For instance, query q_2 arrives at time 3, at which moment the most recent value in the database has age 2.

The scenario described above is characterized by three distinctive elements: (1) a queueing system, (2) a database that is periodically refreshed from the queueing system, and (3) the controller assigning queries to either of the two other elements. Despite a thorough literature review, we did not find any research with the same combination of elements (apart from [2], where we investigate the same scenario using a different model). Caching scenarios, such as the web server example mentioned before, are related, but seem to be not used together with a queueing system. From a queueing theoretic approach, the papers [41] and [108] are somewhat similar to our situation. They deal with several servers for which aged information about the loads is available and, as in our approach, this aged information is periodically updated by the queues via reports. Their system, however, does not contain a database, but has multiple queues that can serve the incoming jobs. The controller decides which of the queues to use based on the aged load information, and thus addresses a problem different from ours.

Addressing the trade-off between data freshness and response times is traditionally done using a threshold policy. When the age of the database value exceeds a certain given threshold, fresh data is retrieved, and otherwise the latest database value is used. Although such policies are commonly used, there is room for improvement by setting a dynamic threshold: in cases where the information retrieval is time-consuming (as it is in wireless sensor networks), using a database value that is slightly above the threshold value might be acceptable. Motivated by this, our aim in the current chapter is to demonstrate for our example scenario that it is possible to find a policy that performs better than a simple threshold policy. We formulate our control problem as a three-dimensional MDP, which turns out to be complex and hard to solve analytically. Then, we provide a clever strategy for reducing the dimensionality of the model, and construct an approximate model that captures the system dynamics in a simpler way, allowing for an analytical solution. After deriving this solution, we apply one-step policy improvement to obtain an improved policy. We numerically compare this policy to the optimal policy, as well as to a myopic policy and to a traditional age-threshold policy. The improved policy achieves near-optimal performance, and has lower costs than the myopic and the age-threshold policy.

The structure of this chapter is as follows. Section 5.2 introduces the MDP used to model the scenario above, and Section 5.3 illustrates the three steps of our approach to finding a near-optimal control policy. Then, Section 5.4 presents the first of these steps, detailing how the approximate model is constructed. The second step, finding a solution to the approximate model, is in Section 5.5. Section 5.6 contains the third and final step, describing the derivation of our near-optimal control policy. Numerical experiments with this policy are presented in Section 5.7, as well as a closer look at the optimal policy. We finish with conclusions and future research directions in Section 5.8.

5.2 Model formulation

The trade-off we discuss in this chapter is between data freshness and query response times. Here, we assume that the query response time is proportional to the current workload of the system, i.e., the number of queries plus the number of report requests in the system. The decision (to use either the DB, or the queueing system) thus depends on the number of queries in the system, on the number of report requests, and on the age of the most recent value in the DB. In order to analyze decision policies, we formulate the scenario as an MDP. The state space is $\mathcal{X} = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$, where $(i, j, N) \in \mathcal{X}$ denotes a system containing i queries and j report requests, and where the latest report refreshed the DB N time units ago. The controller can choose actions a from $\mathcal{A} = \{Q_1, \text{DB}\}$, where Q_1 indicates forwarding of the query to Q_1 (see Figure 5.1). The cost function $c(i, j, N; a)$ incorporates the costs of each action available to the controller:

$$c(i,j,N;a) = \begin{cases} \gamma_1(i+1) + \gamma_2 j + \gamma_3, & \text{if } a = Q_1, \\ (N-T)^+, & \text{if } a = \text{DB}. \end{cases} \tag{5.1}$$

Here, $\gamma_1(i+1)+\gamma_2 j+\gamma_3$ is a weighted sum (with weights $\gamma_1, \gamma_2, \gamma_3 \in \mathbb{R}$) of the number of queries and report requests in the system, reflecting the workload of the system after assigning a new query to it. The term $(N-T)^+$ in the cost function is a penalty for returning a stale value from the DB instead of a fresh value. The parameter T indicates a threshold below which the latest value in the DB is recent enough to answer the query. Note that we took $\gamma_1(i+1)$ rather than $\gamma_1 i$ in the cost function, because we include the query that is about to be assigned to Q_1 when that action is chosen. Additionally, the resulting expression for the improved policy closely resembles a simple myopic policy, enabling us to interpret the difference between the two policies

The state space, the action set, the transition rates, and the cost function define the MDP. More explicitly, the optimality equation of the MDP can be formulated as follows:

$$\begin{aligned} g + V(i,j,N) = {} & \lambda_2 V(i,j+1,N+1) \\ & + \mu_1 V(i-1,j,N+1)\mathbb{1}_{\{i>0\}} + \mu_2 V(i,j-1,0)\mathbb{1}_{\{j>0\}} \\ & + (1-\lambda_1-\lambda_2-\mu_1\mathbb{1}_{\{i>0\}}-\mu_2\mathbb{1}_{\{j>0\}})V(i,j,N+1) \\ & + \lambda_1 \min\bigl\{\gamma_1(i+1)+\gamma_2 j+\gamma_3+V(i+1,j,N+1); \\ & \qquad (N-T)^+ + V(i,j,N+1)\bigr\}, \end{aligned} \tag{5.2}$$

with $V(i,j,N)$ the relative value function and g the time-average costs. The uniformization term is formed by the third line, where we assumed that parameters $\lambda_1, \lambda_2, \mu_1, \mu_2$ are normalized such that $\lambda_1+\lambda_2+\mu_1+\mu_2 = 1$. Hence, we can regard these parameters as transition probabilities and Eq. (5.2) as a discrete-time model. Also note that N measures the number of uniformized time steps since the generation of the last report, and not "real" time. Finally, we assume the stability conditions $\rho_1 := \lambda_1/\mu_1 < 1$ and $\rho_2 := \lambda_2/\mu_2 < 1$ hold.

5.3 Obtaining the near-optimal policy

Ideally, we would like to solve the optimality equation (5.2) analytically and obtain an expression for the relative value function (and, consequently, for the optimal policy). However, the optimality equation has several complicating aspects that prevent us from doing so:

- It contains the decision capturing the trade-off faced by the controller, which involves evaluation of a minimization term.
- In this minimum the inhomogeneous terms $\gamma_1(i+1)+\gamma_2 j+\gamma_3$ and $(N-T)^+$ add to the complexity of the model.
- The state space variables interact with each other, i.e., in Eq. (5.2), $V(i,j,N)$ depends on 'neighbors' $V(i,j+1,N+1)$, $V(i-1,j,N+1)$, $V(i,j,N+1)$, and $V(i+1,j,N+1)$, so it is challenging to find a solution by decomposing the state space.
- A complex relation between j and N exists due to the term $\mu_2 V(i,j-1,0)$.

In order to circumvent these complexities, we take three steps in the upcoming sections. We derive an approximate model to the original problem (step I), which we can solve analytically for a specific policy (step II). This solution is then converted into a near-optimal control policy for the original problem (step III). In more detail, the three steps are:

I We start in Section 5.4 with a modification of the optimality equation (5.2), obtained by removing the N-dimension, resulting in an MDP for an approximation to $V(i,j,N)$ (denoted by $\widetilde{V}(i,j)$).

II In Section 5.5 we choose a policy for this new MDP and solve it analytically, yielding a solution $\widetilde{V}^{\alpha}(i,j)$. Here, α is the parameter of a Bernoulli routing policy.

III Finally, in Section 5.6, we apply *one-step policy improvement* by inspecting the minimum in Eq. (5.2), substituting $\widetilde{V}^{\alpha}(i,j)$ for $V(i,j,N)$. This results in an improved policy, denoted by π'.

Eq. (5.3) schematically reflects the steps and the notation used.

$$V(i,j,N) \xrightarrow{I} \widetilde{V}(i,j) \xrightarrow{II} \widetilde{V}^{\alpha}(i,j) \xrightarrow{III} \pi'. \tag{5.3}$$

5.4 Step I: model approximation

Looking at Eq. (5.2), we see that N is in the state space to accommodate the penalty term $(N-T)^+$. Therefore, if we replace the $(N-T)^+$ by a suitable constant C, the N can be removed from the state space. Introducing the constant C in Eq. (5.2) yields

$$\begin{aligned}\tilde{g} + \widetilde{V}(i,j) = {}& \lambda_2 \widetilde{V}(i,j+1) \\ & + \mu_1 \widetilde{V}(i-1,j)\mathbb{1}_{\{i>0\}} + \mu_2 \widetilde{V}(i,j-1)\mathbb{1}_{\{j>0\}} \\ & + (1-\lambda_1-\lambda_2-\mu_1\mathbb{1}_{\{i>0\}}-\mu_2\mathbb{1}_{\{j>0\}})\widetilde{V}(i,j) \\ & + \lambda_1 \min\left\{\gamma_1(i+1)+\gamma_2 j+\gamma_3+\widetilde{V}(i+1,j); C+\widetilde{V}(i,j)\right\}.\end{aligned} \tag{5.4}$$

As it turns out, the constant C does not affect our near-optimal policy, so assigning a value to it is not strictly necessary (in Section 5.6.1 the term $(N-T)^+$ is reintroduced). However, the idea of reducing the state space in this manner might be applicable to other MDPs, so for completeness we shortly illustrate how C can be determined for Eq. (5.2). To this end, we inspect this MDP for the policy that always uses the DB to answer queries. Replacing the minimum in Eq. (5.2) by this policy yields the equation

$$\begin{aligned}g^{DB} + V^{DB}(j,N) = {}& \lambda_2 V^{DB}(j+1,N+1) + \mu_2 V^{DB}(j-1,0)\mathbb{1}_{\{j>0\}} \\ & + (1-\lambda_1-\lambda_2-\mu_2\mathbb{1}_{\{j>0\}})V^{DB}(j,N+1) \\ & + \lambda_1\left((N-T)^+ + V^{DB}(j,N+1)\right),\end{aligned} \tag{5.5}$$

where variable i is removed from the notation because it no longer influences the relative value function. In [2, Appendix B], we show that $g^{DB} = \lambda_1 \frac{(1-\lambda_2)^{T+1}}{\lambda_2}$. Note that if we replace $(N-T)^+$ by constant C in Eq. (5.5), then we would have $g^{DB} = \lambda_1 C$. A suitable choice for C is one that leaves the time-average costs intact, i.e., $C = \frac{(1-\lambda_2)^{T+1}}{\lambda_2}$.

5.5 Step II: near-optimal control policies

The next step is to fix a policy for the MDP in Eq. (5.4) so that we can obtain an analytic expression for the corresponding relative value function. For this policy we choose the *Bernoulli policy*, which randomly assigns incoming queries to either Q_1 (with probability $\alpha \in [0,1]$) or to the DB (with probability $1-\alpha$). Replacing the minimum in Eq. (5.4) by the Bernoulli policy yields the difference equation

$$\begin{aligned}\tilde{g}^{\alpha} + \widetilde{V}^{\alpha}(i,j) = {} & \lambda_2 \widetilde{V}^{\alpha}(i,j+1) \\ & + \mu_1 \widetilde{V}^{\alpha}(i-1,j)\mathbb{1}_{\{i>0\}} + \mu_2 \widetilde{V}^{\alpha}(i,j-1)\mathbb{1}_{\{j>0\}} \\ & + (1-\lambda_1-\lambda_2-\mu_1\mathbb{1}_{\{i>0\}}-\mu_2\mathbb{1}_{\{j>0\}})\widetilde{V}^{\alpha}(i,j) \\ & + \lambda_1\alpha\left[\gamma_1(i+1)+\gamma_2 j+\gamma_3+\widetilde{V}^{\alpha}(i+1,j)\right] \\ & + \lambda_1(1-\alpha)\left[C+\widetilde{V}^{\alpha}(i,j)\right].\end{aligned} \tag{5.6}$$

Note how the application of the Bernoulli policy decouples the queueing system from the DB. In the remainder of this section we derive an expression for the relative value function $\widetilde{V}^{\alpha}(i,j)$ by solving Eq. (5.6). This result is summarized in the following theorem:

THEOREM 5.5.1. *The solution to Eq. (5.6) is given by*

$$\widetilde{V}^{\alpha}(i,j) = \frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha}\frac{i(i+1)}{2} + \frac{\gamma_2\lambda_1\alpha}{\mu_2-\lambda_2}\frac{j(j+1)}{2},$$

and

$$\tilde{g}^{\alpha} = \lambda_1(1-\alpha)C + \lambda_1\alpha\left(\frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha} + \frac{\gamma_2\lambda_2}{\mu_2-\lambda_2} + \gamma_1 + \gamma_3\right).$$

Substitution of these expressions for $\widetilde{V}^{\alpha}(i,j)$ and $\tilde{g}^{\alpha}$ into Eq. (5.6) shows that these indeed form a solution. In the following subsections we derive the expressions in Theorem 5.5.1 by solving Eq. (5.6). First, we tackle the inhomogeneous terms $\gamma_1(i+1) + \gamma_2 j + \gamma_3$ and C by considering an equation for $\Delta_1 \widetilde{V}^\alpha(i,j) = \widetilde{V}^\alpha(i+1,j) - \widetilde{V}^\alpha(i,j)$. This removes the inhomogeneous term C and transforms the other term to γ_1. Then we look at $\Delta_1^2 \widetilde{V}^\alpha(i,j) = \Delta_1 \widetilde{V}^\alpha(i+1,j) - \Delta_1 \widetilde{V}^\alpha(i,j)$, which eliminates the remaining inhomogeneous term γ_1. We solve this equation, and then retrace our steps from $\Delta_1^2 \widetilde{V}^\alpha(i,j)$ to $\Delta_1 \widetilde{V}^\alpha(i,j)$ to $\widetilde{V}^\alpha(i,j)$.

During the derivation we encounter an issue concerning uniqueness of solutions to the Poisson equation for $\Delta_1^2 \widetilde{V}^\alpha(i,j)$. There, we postulate a form for a solution and must show that this solution is unique. Showing uniqueness is not trivial and involves several technical arguments that result in additional restrictions on the form of $\Delta_1^2 \widetilde{V}^\alpha(i,j)$. This important part of the derivation is placed in Section 5.9.

5.5.1 Solving the difference equation for $\Delta_1^2 \widetilde{V}^\alpha(i,j)$

The behavior of the difference equation on the interior of the state space differs from the behavior on the boundaries $\{i = 0\}$ and $\{j = 0\}$. Therefore, we first study the difference equation for the interior $\{i, j > 0\}$ of the state space. We define $\Delta_1 \widetilde{V}^\alpha(i,j) := \widetilde{V}^\alpha(i+1,j) - \widetilde{V}^\alpha(i,j)$, and for $i > 0$ and $j > 0$ it holds that

$$\begin{aligned} \Delta_1 \widetilde{V}^\alpha(i,j) = \lambda_1 \alpha \Big[\gamma_1 + \Delta_1 \widetilde{V}^\alpha(i+1,j)\Big] &+ \lambda_1(1-\alpha)\Delta_1 \widetilde{V}^\alpha(i,j) \\ &+ \lambda_2 \Delta_1 \widetilde{V}^\alpha(i,j+1) \\ &+ \mu_1 \Delta_1 \widetilde{V}^\alpha(i-1,j) + \mu_2 \Delta_1 \widetilde{V}^\alpha(i,j-1) \\ &+ (1-\lambda_1-\lambda_2-\mu_1-\mu_2)\Delta_1 \widetilde{V}^\alpha(i,j). \end{aligned} \tag{5.7}$$

Now, define $\Delta_1^2 \widetilde{V}^\alpha(i,j) = \Delta_1 \widetilde{V}^\alpha(i+1,j) - \Delta_1 \widetilde{V}^\alpha(i,j)$, for which we have

$$\begin{aligned} \Delta_1^2 \widetilde{V}^\alpha(i,j) = \lambda_1 \alpha \Delta_1^2 \widetilde{V}^\alpha(i+1,j) &+ \lambda_1(1-\alpha)\Delta_1^2 \widetilde{V}^\alpha(i,j) \\ &+ \lambda_2 \Delta_1^2 \widetilde{V}^\alpha(i,j+1) \\ &+ \mu_1 \Delta_1^2 \widetilde{V}^\alpha(i-1,j) + \mu_2 \Delta_1^2 \widetilde{V}^\alpha(i,j-1) \\ &+ (1-\lambda_1-\lambda_2-\mu_1-\mu_2)\Delta_1^2 \widetilde{V}^\alpha(i,j). \end{aligned}$$

We suggestively write this as

$$\begin{aligned} (\lambda_1\alpha + \mu_1)\Delta_1^2 \widetilde{V}^\alpha(i,j) + (\lambda_2 + \mu_2)\Delta_1^2 \widetilde{V}^\alpha(i,j) = \\ \lambda_1 \alpha \Delta_1^2 \widetilde{V}^\alpha(i+1,j) + \mu_1 \Delta_1^2 \widetilde{V}^\alpha(i-1,j) \\ + \lambda_2 \Delta_1^2 \widetilde{V}^\alpha(i,j+1) + \mu_2 \Delta_1^2 \widetilde{V}^\alpha(i,j-1). \end{aligned} \tag{5.8}$$

The notation suggests that the solution to this equation might be split up in a part that only depends on i and a part that only depends on j. That is, a solution might be given by $\Delta_1^2\widetilde{V}^\alpha(i,j) = \widetilde{V}_1^\alpha(i) + \widetilde{V}_2^\alpha(j)$ with $\widetilde{V}_1^\alpha(i)$ and $\widetilde{V}_2^\alpha(j)$ satisfying

$$\begin{cases} (\lambda_1\alpha + \mu_1)\widetilde{V}_1^\alpha(i) = \lambda_1\alpha\widetilde{V}_1^\alpha(i+1) + \mu_1\widetilde{V}_1^\alpha(i-1), \\ (\lambda_2 + \mu_2)\widetilde{V}_2^\alpha(j) = \lambda_2\widetilde{V}_2^\alpha(j+1) + \mu_2\widetilde{V}_2^\alpha(j-1). \end{cases} \tag{5.9}$$

These equations are simple homogeneous difference equations of which the solutions are given by

$$\begin{cases} \widetilde{V}_1^\alpha(i) = \dfrac{\mu_1\widetilde{V}_1^\alpha(0) - \lambda_1\alpha\widetilde{V}_1^\alpha(1)}{\mu_1 - \lambda_1\alpha} + \dfrac{\lambda_1\alpha\big(\widetilde{V}_1^\alpha(1) - \widetilde{V}_1^\alpha(0)\big)\left(\frac{\mu_1}{\lambda_1\alpha}\right)^i}{\mu_1 - \lambda_1\alpha}, \\ \widetilde{V}_2^\alpha(j) = \dfrac{\mu_2\widetilde{V}_2^\alpha(0) - \lambda_2\widetilde{V}_2^\alpha(1)}{\mu_2 - \lambda_2} + \dfrac{\lambda_2\big(\widetilde{V}_2^\alpha(1) - \widetilde{V}_2^\alpha(0)\big)\left(\frac{\mu_2}{\lambda_2}\right)^j}{\mu_2 - \lambda_2}. \end{cases} \tag{5.10}$$

Note that with these expressions for $\widetilde{V}_1\alpha(i)$ and $\widetilde{V}_2\alpha(j)$, $\Delta_1^2\widetilde{V}^\alpha(i,j)$ is **a** solution to Eq. (5.8). It is, however, not immediately obvious that this is also **the** solution. We return to this issue in Section 5.9.

The values for $\widetilde{V}_1^\alpha(0), \widetilde{V}_1^\alpha(1), \widetilde{V}_2^\alpha(0), \widetilde{V}_2^\alpha(1)$ still need to be determined to make the solution consistent at the boundaries. For this purpose, consider the boundary $\{j = 0\}$ of the state space, where $\Delta_1\widetilde{V}^\alpha(i,0)$ becomes (for $i > 0$)

$$\begin{aligned} \Delta_1\widetilde{V}^\alpha(i,0) = {} & \lambda_1\alpha\Big[\gamma_1 + \Delta_1\widetilde{V}^\alpha(i+1,0)\Big] + \lambda_1(1-\alpha)\Delta_1\widetilde{V}^\alpha(i,0) \\ & + \lambda_2\Delta_1\widetilde{V}^\alpha(i,1) + \mu_1\Delta_1\widetilde{V}^\alpha(i-1,0) \\ & + (1 - \lambda_1 - \lambda_2 - \mu_1)\Delta_1\widetilde{V}^\alpha(i,0). \end{aligned} \tag{5.11}$$

Similarly, for $\Delta_1^2\widetilde{V}^\alpha(i,0)$ we have that

$$\begin{aligned} \Delta_1^2\widetilde{V}^\alpha(i,0) = {} & \lambda_1\alpha\Delta_1^2\widetilde{V}^\alpha(i+1,0) + \lambda_1(1-\alpha)\Delta_1^2\widetilde{V}^\alpha(i,0) \\ & + \lambda_2\Delta_1^2\widetilde{V}^\alpha(i,1) + \mu_1\Delta_1^2\widetilde{V}^\alpha(i-1,0) \\ & + (1 - \lambda_1 - \lambda_2 - \mu_1)\Delta_1^2\widetilde{V}^\alpha(i,0). \end{aligned}$$

Again, we can suggestively write this as

$$\begin{aligned} (\lambda_1\alpha + \mu_1)\Delta_1^2\widetilde{V}^\alpha(i,0) + \lambda_2\Delta_1^2\widetilde{V}^\alpha(i,0) = {} & \\ \lambda_1\alpha\Delta_1^2\widetilde{V}^\alpha(i+1,0) + \mu_1\Delta_1^2\widetilde{V}^\alpha(i-1,0) & \\ + \lambda_2\Delta_1^2\widetilde{V}^\alpha(i,1), & \end{aligned}$$

leading to the following system of equations

$$\begin{cases} (\lambda_1\alpha + \mu_1)\widetilde{V}_1^\alpha(i) = \lambda_1\alpha\widetilde{V}_1^\alpha(i+1) + \mu_1\widetilde{V}_1^\alpha(i-1), \\ \qquad\qquad \lambda_2\widetilde{V}_2^\alpha(0) = \lambda_2\widetilde{V}_2^\alpha(1). \end{cases} \tag{5.12}$$

From these expressions, we obtain that on the boundary $\{j = 0\}$ of the state space, the MDP behaves exactly the same as the MDP on the interior of the state space. Furthermore, it shows that $\widetilde{V}_2^\alpha(0) = \widetilde{V}_2^\alpha(1)$ and thus that $\widetilde{V}_2^\alpha(j)$ in Eq. (5.10) is a constant: $\widetilde{V}_2^\alpha(j) = c_2$. Without loss of generality, we can set $c_2 = 0$ and determine $\Delta_1^2\widetilde{V}^\alpha(i,j)$ completely from $\widetilde{V}_1^\alpha(i)$. Hence, we have $\Delta_1^2\widetilde{V}^\alpha(i,j) = \widetilde{V}_1^\alpha(i) + \widetilde{V}_2^\alpha(j)$, where $\widetilde{V}_2^\alpha(j) \equiv 0$ and

$$\widetilde{V}_1^\alpha(i) = \frac{\mu_1\widetilde{V}_1^\alpha(0) - \lambda_1\alpha\widetilde{V}_1^\alpha(1)}{\mu_1 - \lambda_1\alpha} + \frac{\lambda_1\alpha\big(\widetilde{V}_1^\alpha(1) - \widetilde{V}_1^\alpha(0)\big)\left(\frac{\mu_1}{\lambda_1\alpha}\right)^i}{\mu_1 - \lambda_1\alpha}.$$

5.5.2 Analyzing $\Delta_1\widetilde{V}^\alpha(i,j+1) - \Delta_1\widetilde{V}^\alpha(i,j)$

For the derivation of an expression for $\widetilde{V}^\alpha(i,j)$ (which we do in the next sections), we require an intermediate result about $\Delta_1\widetilde{V}^\alpha(i,j+1) - \Delta_1\widetilde{V}^\alpha(i,j)$. With notation

$$\Delta_2\Delta_1\widetilde{V}^\alpha(i,j) := \Delta_1\widetilde{V}^\alpha(i,j+1) - \Delta_1\widetilde{V}^\alpha(i,j),$$

we prove the following lemma in this section:

Lemma 5.5.2. *The relative value function $\widetilde{V}^\alpha(i,j)$ satisfies*

$$\Delta_2\Delta_1\widetilde{V}^\alpha(i,j) = 0.$$

In words, Lemma 5.5.2 states that first differencing $\widetilde{V}^\alpha(i,j)$ in i, followed by differencing the result in j, equals 0.

Proof. We start again for the interior $\{i,j > 0\}$ of the state space, where we have the following relation for $i > 0$ and $j > 0$:

$$\begin{aligned} \Delta_1\widetilde{V}^\alpha(i,j) = {} & \lambda_1\alpha\Big[\gamma_1 + \Delta_1\widetilde{V}^\alpha(i+1,j)\Big] + \lambda_1(1-\alpha)\Delta_1\widetilde{V}^\alpha(i,j) \\ & + \lambda_2\Delta_1\widetilde{V}^\alpha(i,j+1) \\ & + \mu_1\Delta_1\widetilde{V}^\alpha(i-1,j) + \mu_2\Delta_1\widetilde{V}^\alpha(i,j-1) \\ & + (1-\lambda_1-\lambda_2-\mu_1-\mu_2)\Delta_1\widetilde{V}^\alpha(i,j). \end{aligned}$$

We find for $\Delta_2\Delta_1\widetilde{V}^\alpha(i,j)$

$$\begin{aligned}\Delta_2\Delta_1\widetilde{V}^\alpha(i,j) &= \lambda_1\alpha\Delta_2\Delta_1\widetilde{V}^\alpha(i+1,j) + \lambda_1(1-\alpha)\Delta_2\Delta_1\widetilde{V}^\alpha(i,j)\\ &\quad + \lambda_2\Delta_2\Delta_1\widetilde{V}^\alpha(i,j+1)\\ &\quad + \mu_1\Delta_2\Delta_1\widetilde{V}^\alpha(i-1,j) + \mu_2\Delta_2\Delta_1\widetilde{V}^\alpha(i,j-1)\\ &\quad + (1-\lambda_1-\lambda_2-\mu_1-\mu_2)\Delta_2\Delta_1\widetilde{V}^\alpha(i,j).\end{aligned} \tag{5.13}$$

By similar line of reasoning as before, we derive that $\Delta_2\Delta_1\widetilde{V}^\alpha(i,j) = \bar{V}_1(i) + \bar{V}_2(j)$, with

$$\begin{aligned}\bar{V}_1(i) &= \frac{\mu_1\bar{V}_1(0) - \lambda_1\alpha\bar{V}_1(1)}{\mu_1-\lambda_1\alpha} + \frac{\lambda_1\alpha\big(\bar{V}_1(1)-\bar{V}_1(0)\big)\left(\frac{\mu_1}{\lambda_1\alpha}\right)^i}{\mu_1-\lambda_1\alpha},\\ \bar{V}_2(j) &= \frac{\mu_2\bar{V}_2(0) - \lambda_2\bar{V}_2(1)}{\mu_2-\lambda_2} + \frac{\lambda_2\big(\bar{V}_2(1)-\bar{V}_2(0)\big)\left(\frac{\mu_2}{\lambda_2}\right)^j}{\mu_2-\lambda_2},\end{aligned} \tag{5.14}$$

where the values for $\bar{V}_1(0), \bar{V}_1(1), \bar{V}_2(0), \bar{V}_2(1)$ are determined from $\Delta_2\Delta_1\widetilde{V}^\alpha(i,0)$ and $\Delta_2\Delta_1\widetilde{V}^\alpha(0,j)$. We start with the former by inspecting the term $\Delta_1\widetilde{V}^\alpha(i,1)$. From Eq. (5.7) we have that

$$\begin{aligned}\Delta_1\widetilde{V}^\alpha(i,1) &= \lambda_1\alpha\Big[\gamma_1 + \Delta_1\widetilde{V}^\alpha(i+1,1)\Big] + \lambda_1(1-\alpha)\Delta_1\widetilde{V}^\alpha(i,1)\\ &\quad + \lambda_2\Delta_1\widetilde{V}^\alpha(i,2)\\ &\quad + \mu_1\Delta_1\widetilde{V}^\alpha(i-1,1) + \mu_2\Delta_1\widetilde{V}^\alpha(i,0)\\ &\quad + (1-\lambda_1-\lambda_2-\mu_1-\mu_2)\Delta_1\widetilde{V}^\alpha(i,1).\end{aligned}$$

The term $\Delta_1\widetilde{V}^\alpha(i,0)$ can be obtained from Eq. (5.11), which we repeat here for convenience:

$$\begin{aligned}\Delta_1\widetilde{V}^\alpha(i,0) &= \lambda_1\alpha\Big[\gamma_1 + \Delta_1\widetilde{V}^\alpha(i+1,0)\Big] + \lambda_1(1-\alpha)\Delta_1\widetilde{V}^\alpha(i,0)\\ &\quad + \lambda_2\Delta_1\widetilde{V}^\alpha(i,1)\\ &\quad + \mu_1\Delta_1\widetilde{V}^\alpha(i-1,0)\\ &\quad + (1-\lambda_1-\lambda_2-\mu_1)\Delta_1\widetilde{V}^\alpha(i,0).\end{aligned}$$

Consequently,

$$\begin{aligned}\Delta_2\Delta_1\widetilde{V}^\alpha(i,0) &= \lambda_1\alpha\Delta_2\Delta_1\widetilde{V}^\alpha(i+1,0) + \lambda_1(1-\alpha)\Delta_2\Delta_1\widetilde{V}^\alpha(i,0)\\ &\quad + \lambda_2\Delta_2\Delta_1\widetilde{V}^\alpha(i,1)\\ &\quad + \mu_1\Delta_2\Delta_1\widetilde{V}^\alpha(i-1,0) - \mu_2\Delta_2\Delta_1\widetilde{V}^\alpha(i,0)\\ &\quad + (1-\lambda_1-\lambda_2-\mu_1)\Delta_2\Delta_1\widetilde{V}^\alpha(i,0),\end{aligned}$$

which reduces to

$$(\lambda_2+\mu_2)\Delta_2\Delta_1\widetilde{V}^\alpha(i,0)+(\lambda_1\alpha+\mu_1)\Delta_2\Delta_1\widetilde{V}^\alpha(i,0)=\\ \lambda_2\Delta_2\Delta_1\widetilde{V}^\alpha(i,1)+\lambda_1\alpha\Delta_2\Delta_1\widetilde{V}^\alpha(i+1,0)+\mu_1\Delta_2\Delta_1\widetilde{V}^\alpha(i-1,0).$$

Again we propose a solution of the type $\Delta_2\Delta_1\widetilde{V}^\alpha(i,j)=\bar{V}_1(i)+\bar{V}_2(j)$, resulting in

$$\begin{cases} (\lambda_2+\mu_2)\big(\bar{V}_1(i)+\bar{V}_2(0)\big)=\lambda_2\big(\bar{V}_1(i)+\bar{V}_2(1)\big),\\ (\lambda_1\alpha+\mu_1)(\bar{V}_1(i)+\bar{V}_2(0))=\lambda_1\alpha(\bar{V}_1(i+1)+\bar{V}_2(0))\\ \qquad\qquad+\mu_1(\bar{V}_1(i-1)+\bar{V}_2(0)).\end{cases} \tag{5.15}$$

The upper equation translates to

$$\mu_2\bar{V}_1(i)=\lambda_2\bar{V}_2(1)-(\lambda_2+\mu_2)\bar{V}_2(0), \tag{5.16}$$

i.e., $\bar{V}_1(i)$ is constant for $i>0$, which we denote by $\bar{V}_1(i)=\bar{c}_1$. By repeating the arguments above for the boundary $\{i=0\}$ of the state space, we find that $\bar{V}_2(j):=\bar{c}_2$ is constant. As a consequence, Eq. (5.16) reduces to

$$\mu_2\bar{c}_1=\lambda_2\bar{c}_2-(\lambda_2+\mu_2)\bar{c}_2,$$

or

$$\mu_2\bar{c}_1=-\mu_2\bar{c}_2,$$

i.e., $\bar{c}_1=-\bar{c}_2$ and thus $\Delta_2\Delta_1\widetilde{V}^\alpha(i,j)=0$, which concludes the proof.

□

5.5.3 Solving the difference equation for $\Delta_1\widetilde{V}^\alpha(i,j)$

So far, we have found that $\Delta_1^2\widetilde{V}^\alpha(i,j)$ satisfies

$$\Delta_1^2\widetilde{V}^\alpha(i,j)=\frac{\mu_1\widetilde{V}_1^\alpha(0)-\lambda_1\alpha\widetilde{V}_1^\alpha(1)}{\mu_1-\lambda_1\alpha}+\frac{\lambda_1\alpha\big(\widetilde{V}_1^\alpha(1)-\widetilde{V}_1^\alpha(0)\big)\left(\frac{\mu_1}{\lambda_1\alpha}\right)^i}{\mu_1-\lambda_1\alpha}, \tag{5.17}$$

and we proved that $\Delta_2\Delta_1\widetilde{V}^\alpha(i,j)=0$ in Lemma 5.5.2. Note that this implies that $\Delta_1\widetilde{V}^\alpha(i,j)$ is independent of j for all i. We continue the proof of Theorem. 5.5.1 by reverting the differencing in i used to obtain Eq. (5.17).

Recall that $\Delta_1^2 \widetilde{V}^\alpha(i,j) = \Delta_1 \widetilde{V}^\alpha(i+1,j) - \Delta_1 \widetilde{V}^\alpha(i,j)$. By summing over i, and then using the right-hand side of Eq. (5.17), we can get an expression for $\Delta_1 \widetilde{V}^\alpha(i,j)$:

$$\begin{aligned}\Delta_1 \widetilde{V}^\alpha(i,j) &= \Delta_1 \widetilde{V}^\alpha(0,j) + \sum_{k=0}^{i-1} \Delta_1^2 \widetilde{V}^\alpha(k,j) \\ &= \Delta_1 \widetilde{V}^\alpha(0,j) + \frac{\mu_1 \widetilde{V}_1^\alpha(0) - \lambda_1 \alpha \widetilde{V}_1^\alpha(1)}{\mu_1 - \lambda_1 \alpha} i \\ &+ \frac{\lambda_1 \alpha \big(\widetilde{V}_1^\alpha(1) - \widetilde{V}_1^\alpha(0)\big)}{\mu_1 - \lambda_1 \alpha} \cdot \frac{1 - (\frac{\mu_1}{\lambda_1 \alpha})^i}{1 - \frac{\mu_1}{\lambda_1 \alpha}}.\end{aligned} \tag{5.18}$$

Here, $\Delta_1 \widetilde{V}^\alpha(0,j)$ is a constant (by Lemma 5.5.2) which we determine below. Substituting the expression for $\Delta_1 \widetilde{V}^\alpha(i,j)$ from Eq. (5.18) into Eq. (5.7), we find that necessarily

$$\mu_1 \widetilde{V}_1^\alpha(0) - \lambda_1 \alpha \widetilde{V}_1^\alpha(1) = \gamma_1 \lambda_1 \alpha.$$

Solving this for $\widetilde{V}_1^\alpha(1)$ and substituting the result into Eq. (5.17) yields

$$\Delta_1^2 \widetilde{V}^\alpha(i,j) = \frac{\gamma_1 \lambda_1 \alpha}{\mu_1 - \lambda_1 \alpha} + \left[\widetilde{V}_1^\alpha(0) - \frac{\gamma_1 \lambda_1 \alpha}{\mu_1 - \lambda_1 \alpha}\right] \left(\frac{\mu_1}{\lambda_1 \alpha}\right)^i.$$

Hence, $\Delta_1 \widetilde{V}^\alpha(i,j)$ becomes

$$\begin{aligned}\Delta_1 \widetilde{V}^\alpha(i,j) &= \Delta_1 \widetilde{V}^\alpha(0,j) + \frac{\gamma_1 \lambda_1 \alpha}{\mu_1 - \lambda_1 \alpha} i \\ &+ \left[\widetilde{V}_1^\alpha(0) - \frac{\gamma_1 \lambda_1 \alpha}{\mu_1 - \lambda_1 \alpha}\right] \frac{1 - (\frac{\mu_1}{\lambda_1 \alpha})^i}{1 - \frac{\mu_1}{\lambda_1 \alpha}}.\end{aligned} \tag{5.19}$$

Now we turn our attention to determining the (constant) $\Delta_1 \widetilde{V}^\alpha(0,j)$ by inspecting the corresponding difference equation:

$$\begin{aligned}\Delta_1 \widetilde{V}^\alpha(0,j) &= \lambda_1 \alpha \Big[\gamma_1 + \Delta_1 \widetilde{V}^\alpha(1,j)\Big] + \lambda_1(1-\alpha) \Delta_1 \widetilde{V}^\alpha(0,j) \\ &+ \lambda_2 \Delta_1 \widetilde{V}^\alpha(0,j+1) + \mu_2 \Delta_1 \widetilde{V}^\alpha(0,j-1) \\ &+ (1 - \lambda_1 - \lambda_2 - \mu_1 - \mu_2) \Delta_1 \widetilde{V}^\alpha(0,j).\end{aligned}$$

We can rewrite this equation as follows:

$$\begin{aligned}0 = {} & \lambda_1 \alpha [\Delta_1 \widetilde{V}^\alpha(1,j) - \Delta_1 \widetilde{V}^\alpha(0,j)] + \gamma_1 \lambda_1 \alpha \\ & + \lambda_2 [\Delta_1 \widetilde{V}^\alpha(0,j+1) - \Delta_1 \widetilde{V}^\alpha(0,j)] \\ & + \mu_2 [\Delta_1 \widetilde{V}^\alpha(0,j-1) - \Delta_1 \widetilde{V}^\alpha(0,j)] - \mu_1 \Delta_1 \widetilde{V}^\alpha(0,j).\end{aligned}$$

Using Lemma 5.5.2 we find

$$0 = \lambda_1\alpha[\Delta_1\widetilde{V}^\alpha(1,j) - \Delta_1\widetilde{V}^\alpha(0,j)] + \gamma_1\lambda_1\alpha - \mu_1\Delta_1\widetilde{V}^\alpha(0,j).$$

Eq. (5.19) tells us that $\Delta_1\widetilde{V}^\alpha(1,j) = \Delta_1\widetilde{V}^\alpha(0,j) + \widetilde{V}_1^\alpha(0)$, so

$$\Delta_1\widetilde{V}^\alpha(0,j) = \frac{\lambda_1\alpha}{\mu_1}\widetilde{V}_1^\alpha(0) + \frac{\gamma_1\lambda_1\alpha}{\mu_1}.$$

Substitution into Eq. (5.19) yields

$$\begin{aligned}\Delta_1\widetilde{V}^\alpha(i,j) = {} & \frac{\lambda_1\alpha}{\mu_1}\widetilde{V}_1^\alpha(0) + \frac{\gamma_1\lambda_1\alpha}{\mu_1} + \frac{\gamma_1\lambda_1\alpha}{\mu_1 - \lambda_1\alpha}i \\ & + \left[\widetilde{V}_1^\alpha(0) - \frac{\gamma_1\lambda_1\alpha}{\mu_1 - \lambda_1\alpha}\right]\frac{1 - (\frac{\mu_1}{\lambda_1\alpha})^i}{1 - \frac{\mu_1}{\lambda_1\alpha}}.\end{aligned} \tag{5.20}$$

5.5.4 Deriving $\widetilde{V}^\alpha(i,j)$

We derive an expression for $\widetilde{V}^\alpha(i,j)$ using $\Delta_1\widetilde{V}^\alpha(i,j) = \widetilde{V}^\alpha(i+1,j) - \widetilde{V}^\alpha(i,j)$, then summing over i, followed by applying Eq. (5.20):

$$\begin{aligned}\widetilde{V}^\alpha(i,j) &= \widetilde{V}^\alpha(0,j) + \sum_{k=0}^{i-1}\Delta_1\widetilde{V}^\alpha(k,j) \\ &= \widetilde{V}^\alpha(0,j) + i\Big(\frac{\lambda_1\alpha}{\mu_1}\widetilde{V}_1^\alpha(0) + \frac{\gamma_1\lambda_1\alpha}{\mu_1}\Big) \\ &+ \frac{\gamma_1\lambda_1\alpha}{\mu_1 - \lambda_1\alpha}\frac{i(i-1)}{2} \\ &+ \frac{1}{1 - \frac{\mu_1}{\lambda_1\alpha}}\left[\widetilde{V}_1^\alpha(0) - \frac{\gamma_1\lambda_1\alpha}{\mu_1 - \lambda_1\alpha}\right]\left[i - \frac{1 - (\frac{\mu_1}{\lambda_1\alpha})^i}{1 - \frac{\mu_1}{\lambda_1\alpha}}\right].\end{aligned} \tag{5.21}$$

In the derivation so far we have postulated a form of a solution several times (Eqs. (5.9) and (5.12–5.15)), resulting in the expression for $\widetilde{V}^\alpha(i,j)$ in Eq. (5.21). Here, we finally deal with the uniqueness issue. As mentioned earlier, ensuring uniqueness of a solution $\widetilde{V}^\alpha(i,j)$ to Eq. (5.6) is not trivial. Conventional uniqueness proofs rely on bounded cost functions, and the cost function in Eq. (5.1) is unbounded. Addressing this point requires several technical arguments which we, for readability, place in Section 5.9. In short, uniqueness is ensured if $\widetilde{V}^\alpha(i,j)$ does not grow exponentially fast. Therefore, we choose the remaining constant $\widetilde{V}_1^\alpha(0)$ in Eq. (5.21) such that the exponential term $(\frac{\mu_1}{\lambda_1\alpha})^i$ disappears:

$$\widetilde{V}_1^\alpha(0) = \frac{\gamma_1\lambda_1\alpha}{\mu_1 - \lambda_1\alpha}.$$

Substitution into Eq. (5.21) yields

$$\widetilde{V}^{\alpha}(i,j) = \widetilde{V}^{\alpha}(0,j) + \frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha}i + \frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha}\frac{i(i-1)}{2},$$

or

$$\widetilde{V}^{\alpha}(i,j) = \widetilde{V}^{\alpha}(0,j) + \frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha}\frac{i(i+1)}{2}.$$

Repeating the steps in Sections 5.5.1-5.5.4 for differencing in j instead of i gives

$$\widetilde{V}^{\alpha}(i,j) = \widetilde{V}^{\alpha}(i,0) + \frac{\gamma_2\lambda_1\alpha}{\mu_2-\lambda_2}\frac{j(j+1)}{2},$$

so that necessarily

$$\widetilde{V}^{\alpha}(i,j) = \frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha}\frac{i(i+1)}{2} + \frac{\gamma_2\lambda_1\alpha}{\mu_2-\lambda_2}\frac{j(j+1)}{2}. \tag{5.22}$$

Finally, substituting this expression for $\widetilde{V}^{\alpha}(i,j)$ into Eq. (5.6) and solving for $\tilde{g}^{\alpha}$ yields

$$\tilde{g}^{\alpha} = \lambda_1(1-\alpha)C + \lambda_1\alpha\left(\frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha} + \frac{\gamma_2\lambda_2}{\mu_2-\lambda_2} + \gamma_1 + \gamma_3\right). \tag{5.23}$$

This concludes the derivation of the expressions in Theorem 5.5.1.

Remark: The structure of $\widetilde{V}^{\alpha}(i,j)$ in Eq. (5.22) and $\tilde{g}^{\alpha}$ in Eq. (5.23) can be explained intuitively using known results about the $M/M/1$ queue. The Bernoulli policy chooses Q_1 with probability α and the DB with probability $1-\alpha$, thereby decoupling the system in three separate elements: the DB, Q_1, and Q_2. Choosing the DB incurs a penalty C, which results in time-average costs $\lambda_1(1-\alpha)C$. This corresponds to the first term in Eq. (5.23). The alternative choice (assignment to the queueing system) incurs costs $\gamma_1(i+1)+\gamma_2 j+\gamma_3$. Note that the two queues (the first with arrival rate $\lambda_1\alpha$, the second with arrival rate λ_2) are independent and that the i and j terms are summed in the cost function. Consequently, the time-average costs of assignment to the queueing system are just the summed time-average costs of the two $M/M/1$ queues with holding costs γ_1 and γ_2 respectively (and of fixed costs $\gamma_1+\gamma_3$). For a $M/M/1$ queue we know (from, e.g., [28]) that $g = \frac{\rho}{1-\rho}h$, with $\rho = \lambda/\mu$ the system load, λ the arrival rate, μ the service rate, and holding costs h. This explains the $\frac{\gamma_1\lambda_1\alpha}{\mu_1-\lambda_1\alpha} + \frac{\gamma_2\lambda_2}{\mu_2-\lambda_2} + \gamma_1 + \gamma_3$ term (multiplied by $\lambda_1\alpha$) in Eq. (5.23). Also, the relative value function $\widetilde{V}^{\alpha}(i,j)$ in Eq. (5.22) is just the sum of the relative value functions of the two $M/M/1$ queues (multiplied by $\lambda_1\alpha$).

5.6 Step III: one-step policy improvement

5.6.1 Obtaining the improved policy

In the previous section we approximated $V(i,j,N)$ by $\widetilde{V}^\alpha(i,j)$. Next, we apply a technique called *one-step policy improvement*, introduced in [118], by inspecting the minimization term in Eq. (5.2), with $V(i,j,N)$ replaced by $\widetilde{V}^\alpha(i,j)$:

$$\min\left\{\gamma_1(i+1)+\gamma_2 j+\gamma_3+\widetilde{V}^\alpha(i+1,j);(N-T)^+ + \widetilde{V}^\alpha(i,j)\right\}. \quad (5.24)$$

Hence, the improved policy assigns a query to the DB if

$$\gamma_1(i+1)+\gamma_2 j+\gamma_3+\widetilde{V}^\alpha(i+1,j) \geq (N-T)^+ + \widetilde{V}^\alpha(i,j).$$

Substituting Eq. (5.22) and simplifying yields

$$\frac{\gamma_1}{1-\rho_1\alpha}(i+1)+\gamma_2 j+\gamma_3 \geq (N-T)^+. \quad (5.25)$$

Note that this improved policy is independent of the constant C, as mentioned at the beginning of Section 5.5. Also, in the derivation of Eq. (5.25) we see that by choosing $\gamma_1(i+1)$ rather than $\gamma_1 i$ in the cost function, we obtain an expression where the α only occurs in front of the $(i+1)$ term. This allows us to intuitively explain the role of α: it acts as a tuning parameter of the improved policy, determining the influence of the number of queries i in the system on the decisions. For $\alpha=0$ the improved policy is independent of λ_1, but as α gets closer to 1 the number of queries in the system is weighed more heavily in the decision, and the policy becomes more biased towards the DB.

5.6.2 Determining α

The improved policy in Eq. (5.25) specifies a **class** of policies – only after choosing α (originally the parameter of the Bernoulli policy) do we have a concrete policy for which we can, e.g., determine average costs. However, we have no analytical relationship between $V(i,j,N)$ and $\widetilde{V}^\alpha(i,j)$, and thus determining α analytically is not possible. The best analytical option we have is to minimize $\tilde{g}^\alpha$ (of the Bernoulli policy applied to the simplified MDP) w.r.t. α, and use the resulting minimum for the improved policy. Unfortunately, subsequent experiments with value iteration demonstrate unsatisfactory performance of

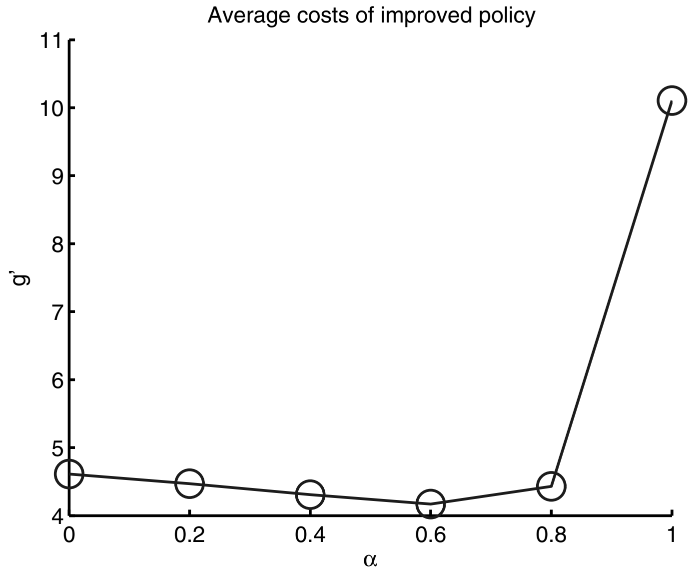

FIGURE 5.3: Average costs g' of the improved policy, for various values of α. The points in the graph are obtained with value iteration, using parameters $\mu_1 = \mu_2 = 0.3, T = 2, \gamma_1 = \gamma_2 = \gamma_3 = 3, \rho_1 = 0.8, \rho_2 = 0.1$. Our fitting approach for determining the minimum $\hat{\alpha}$ yields $\hat{\alpha} = 0.48$.

the resulting improved policy. We observed this behavior for various values for $\lambda_1, \lambda_2, \mu_1, \mu_2$, and T, so the unsatisfactory performance was general. The $\widetilde{V}^\alpha(i,j)$ and $\tilde{g}^\alpha$ do not approximate $V(i,j,N)$ and g from Eq. (5.2) sufficiently well.

Fortunately, a simple numerical approach allows us to compute an α that yields an improved policy with the desired near-optimal performance. To illustrate this procedure, consider Figure 5.3 showing approximations of the average costs g' of the improved policy (obtained with value iteration) as a function of α. The shape resembles a second-degree polynomial, and by carefully fitting such a polynomial to the approximate values, we can approximate $g'(\alpha)$. Then, we use the minimum $\hat{\alpha}$ of the fitted polynomial as input for the improved policy. Note that, due to this procedure, the improved policy is not an analytical policy: each time an improved policy is required, $\hat{\alpha}$ must be computed using the fitting procedure.

This approach for determining $\hat{\alpha}$ requires several approximate values α_i that together capture the shape of $g'(\alpha)$. They should be positioned such that the minimum of the polynomial and that of $g'(\alpha)$ are at approximately the same α-value. Strictly speaking we need only three α-values to fit a second-degree polynomial. However, $g'(\alpha)$ is not truly a second-degree polynomial, and using four values results in a more appropriate fit in cases where $g'(\alpha)$ resembles the polynomial shape less. So how should we position these four points? In the next section we argue that the most interesting scenario from a practical point of view is one where ρ_1 is large. In this scenario, the number of queries i in the system is typically large. Recall that $\hat{\alpha}$ influences the improved policy in Eq. (5.25) via i: as $\hat{\alpha}$ gets closer to 1 the number of queries in the system is weighed more heavily in the decision, and the policy becomes more biased towards the DB. Hence, we should concentrate the fit of the polynomial on the right side of the interval, near $\alpha = 1$. Following this reasoning, we take $\alpha_1 = 0.25, \alpha_2 = 0.6, \alpha_3 = 0.85$, and $\alpha_4 = 0.95$.

The value of each $g'(\alpha_i)$ is obtained by running value iteration. The time needed to execute these four runs of value iteration should be shorter than the time needed to compute the optimal policy, otherwise there is no reason to use the improved policy. To this end, we do value iteration for the $g'(\alpha_i)$ on a much smaller state space than the one used for finding the optimal policy. Suppose that we run value iteration for the optimal policy on the truncated state space $\bar{\mathcal{X}} = [0, K_1] \times [0, K_2] \times [0, K_3]$ (in Section 5.7 we determine K_1, K_2, and K_3 experimentally in such a way that we avoid boundary effects). For the $g'(\alpha_i)$, we use the further truncated state space $\hat{\mathcal{X}} := \big[0, \lfloor \frac{K_1}{4} \rfloor\big] \times \big[0, \lfloor \frac{K_2}{4} \rfloor\big] \times \big[0, \lfloor \frac{K_3}{4} \rfloor\big]$. This effectively reduces the time needed to calculate $\hat{\alpha}$ (and thus also the improved policy) to a mere fraction of the time needed to obtain an optimal policy. The number by which K_1, K_2, and K_3 are divided (4) is determined experimentally to yield both low time-average costs and a short run time for the improved policy. Note that the further reduction of the state space is appropriate, because we do not require numerically accurate approximations of $g'(\alpha_1), \ldots, g'(\alpha_4)$. We only need to capture the general shape illustrated in Figure 5.3.

The complete procedure is as follows:

1. Calculate the bounds of the further truncated state space $\hat{\mathcal{X}}$.
2. For each of the values α_i, evaluate the improved policy using $\hat{\mathcal{X}}$ as state space, and record $g'(\alpha_i)$.
3. Fit a second-degree polynomial through $g'(\alpha_1), \ldots, g'(\alpha_4)$ using least squares.
4. Calculate the minimum of this polynomial, and use the α-value for which this minimum is attained as $\hat{\alpha}$.

In the example in Figure 5.3 this procedure yields $\hat{\alpha} = 0.48$, which agrees well with what the figure suggests. Figure 5.3 is generated with parameters $\mu_1 = \mu_2 = 0.3, T = 2, \gamma_1 = \gamma_2 = \gamma_3 = 3, \rho_1 = 0.8, \rho_2 = 0.1$, i.e., values corresponding to a high load on Q_1 and low load on Q_2. We expect a significant fraction of the queries to be assigned to Q_1, since a low load on Q_2 results in large N and thus using the DB is expensive. This observation is supported by the value $\hat{\alpha} = 0.48$ that our procedure yields for the improved policy. Also, the figure indicates that the sensitivity of the average costs $g'(\alpha)$ to α is minor around the minimum $\hat{\alpha}$.

5.7 Numerical results

In this section we experimentally inspect the performance of the improved policy by numerically comparing it to the optimal policy. Additionally, we compare a traditional age-threshold policy and a myopic policy to the optimal policy, allowing us to assess how the improved policy performs in relation to these other two policies. The three policies that we compare to the optimal policy listed in Eq. (5.26). We see that the age-threshold policy π^t ignores the load on the queueing system, and bases its actions solely on the age N. The myopic policy π^m takes the load of the system into account, by assigning queries to the DB or Q_1 based on the cost function in Eq. (5.1) only, ignoring the relative value function $V(i, j, N)$. In contrast, the improved policy π' is based on an approximation of the relative value function, and thus does include expectations about future query arrivals and report requests in its decisions. These expectations are captured by the parameter $\hat{\alpha}$, which determines how much emphasis the improved policy puts on the number of queries i in the system. Note that for $\hat{\alpha} = 0$ the improved and myopic policy are identical.

$$\begin{aligned}
\pi^t(i,j,N) &= \begin{cases} \text{DB}, & \text{if } N \leq T, \\ Q_1, & \text{otherwise}, \end{cases} \\
\pi^m(i,j,N) &= \begin{cases} \text{DB}, & \text{if } \gamma_1(i+1) + \gamma_2 j + \gamma_3 \geq (N-T)^+, \\ Q_1, & \text{otherwise}, \end{cases} \qquad (5.26) \\
\pi'(i,j,N) &= \begin{cases} \text{DB}, & \text{if } \frac{\gamma_1}{1-\rho_1\hat{\alpha}}(i+1) + \gamma_2 j + \gamma_3 \geq (N-T)^+, \\ Q_1, & \text{otherwise}. \end{cases}
\end{aligned}$$

Looking at our scenario, we expect that as $\rho_2 \to 1$, performance should be quite good, since the DB is refreshed often and thus most queries can be answered from the DB. Additionally, in situations with small ρ_1 the controller has to deal with only a small number of queries, costs are typically low, and the policies should demonstrate good performance. Hence, the most interesting part of the parameter space is where ρ_1 is high and ρ_2 is low (we call this the *critical region*). We structure our numerical analysis accordingly, by first inspecting the performance of the policies for $0 < \rho_1 \leq 0.8, 0 < \rho_2 < 1$, followed by an inspection of the critical region $0.7 < \rho_1 \leq 1, 0 < \rho_2 < 0.2$.

All numerical experiments below are done using the value iteration algorithm [159], and thus require a truncation of the state space $\mathcal{X} = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$ to $\bar{\mathcal{X}} = [0, K_1] \times [0, K_2] \times [0, K_3]$. Choosing the K_i must be done carefully to avoid the influence of boundary effects on the average costs. Tests on the three policies above, and on the optimal policy, suggests that a truncation to $\bar{\mathcal{X}} = [0, 200] \times [0, 200] \times [0, 200]$ is sufficient for $0 < \rho_1 \leq 0.8, 0 < \rho_2 < 1$. Increasing ρ_1 beyond 0.8 quickly adds boundary effects and requires a larger truncated state space: $\bar{\mathcal{X}} = [0, 300] \times [0, 300] \times [0, 300]$. Also, for value iteration we set the convergence criterion such that the procedure stops when the difference of the spans of two consecutive approximations is smaller than 0.001. Finally, we choose the parameters of the cost function in Eq. (5.1). We set $T = 2, \gamma_1 = \gamma_2 = \gamma_3 = 3$ and keep these fixed during all experiments.

In the following sections we numerically investigate the performance of our improved policy. First, we compare the three policies listed above to the optimal policy in Sections 5.7.1 (for the non-critical region) and 5.7.2 (for the critical region). Then in Section 5.7.3 we look at the time needed to calculate $\hat{\alpha}$, and thus the improved policy. Section 5.7.4 introduces a special random policy, where the controller flips a (fair) coin to decide which of the two actions to take. A large number of such policies are then compared to the three policies described above. Finally, in Section 5.7.5 we take a closer look at the optimal policy and its structure.

5.7.1 Analysis of region $0 < \rho_1 \leq 0.8, 0 < \rho_2 < 1$

In Figures 5.4 – 5.6 we inspect the performance of the three policies as compared to the optimal policy. We fix $\mu_1 = \mu_2 = 0.3$ and vary ρ_1 and ρ_2. The figures contain the difference in average costs with the optimal policy (in %), where the load ρ_2 on Q_2 is varied on the horizontal axis, and the load ρ_1 of Q_1 is reflected by the various lines. Figure 5.4 demonstrates that the simple

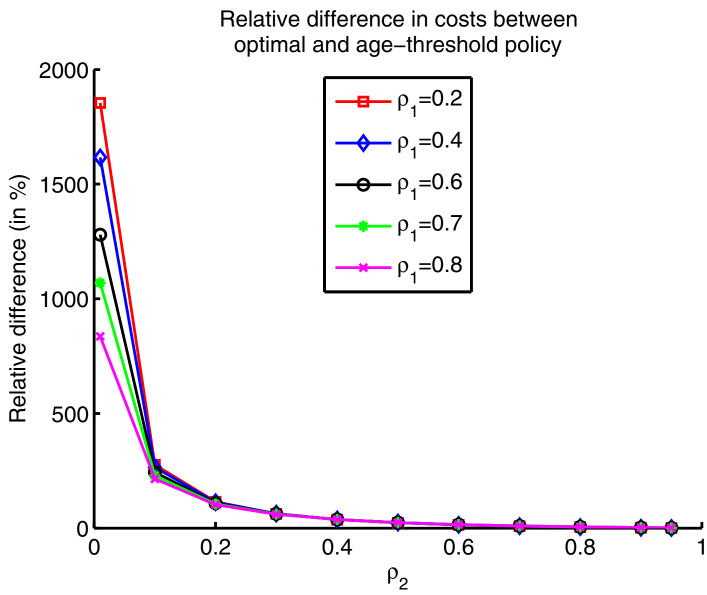

FIGURE 5.4: Relative difference in average costs of the age-threshold policy compared to the optimal policy.

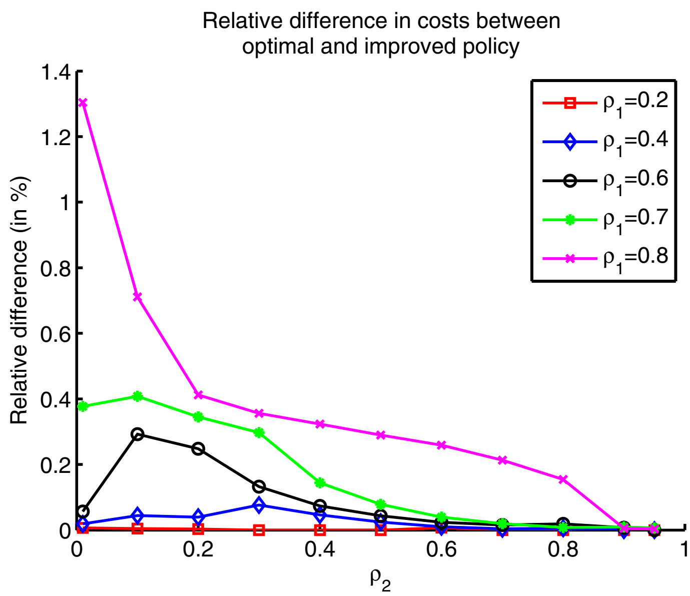

FIGURE 5.5: Relative difference in average costs of the improved policy compared to the optimal policy.

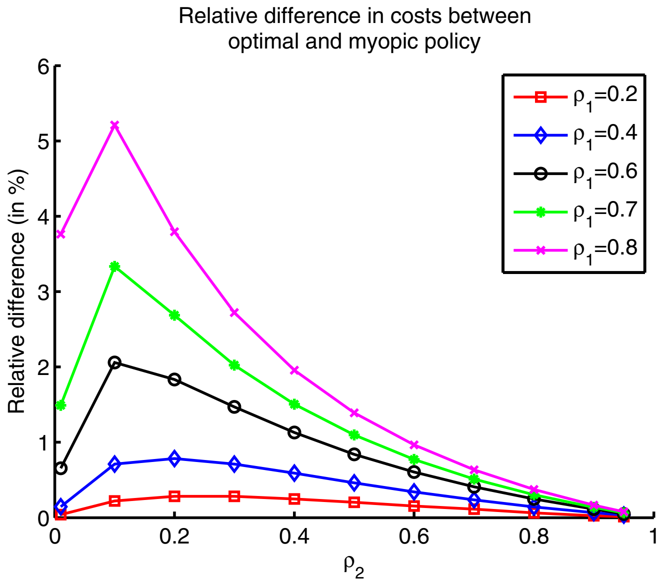

FIGURE 5.6: Relative difference in average costs of the myopic policy compared to the optimal policy.

age-threshold policy differs from optimality by as much as 2,000%. In contrast, the improved and myopic policies in Figures 5.5 and 5.6 are able to stay within 1.3% and 5.5% of optimality, respectively. Clearly, these policies perform significantly better than the age-threshold policy, so including the load of the queue system in the decision by the controller certainly is beneficial. Further inspection of Figures 5.4 – 5.6 reveals that the performance of the three policies degrades when ρ_1 and ρ_2 reach the critical region. We take a detailed look at this region in the next section.

5.7.2 Analysis of the critical region $0.7 < \rho_1 \leq 1, 0 < \rho_2 < 0.2$

We continue with a closer look at the critical region, i.e., the left-hand side of Figures 5.4 – 5.6, by repeating the corresponding numerical experiments for different values of ρ_1 and ρ_2 (again with $\mu_1 = \mu_2 = 0.3$) . The results are in Figures 5.7 – 5.9. As in the previous section, performance of the age-threshold policy is quite bad, with differences of up to 1,500%. Comparing Figures 5.8 to 5.9 clearly demonstrate that the improved policy has better overall performance than the myopic policy, with differences from optimality of at most 7% and 17%, respectively. The benefits of including the approximation to the relative value function in the improved policy are evident here.

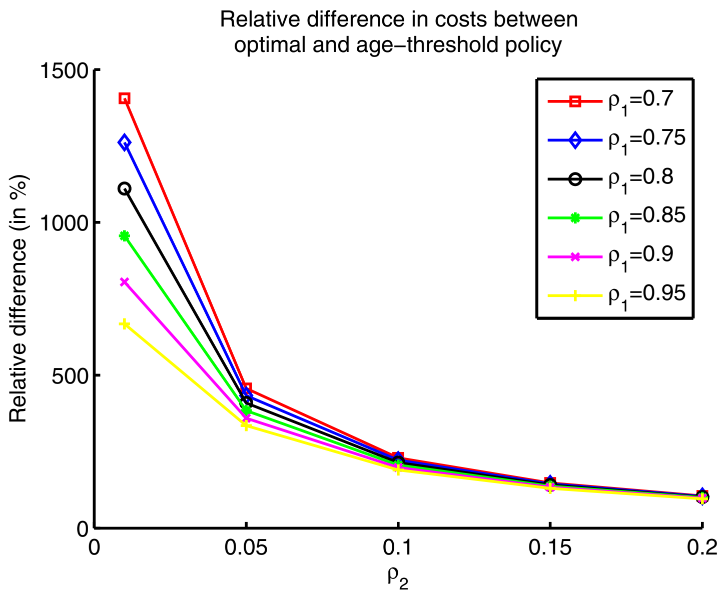

FIGURE 5.7: Again, the relative difference in average costs of the age-threshold policy compared to the optimal policy, but now inside the critical region.

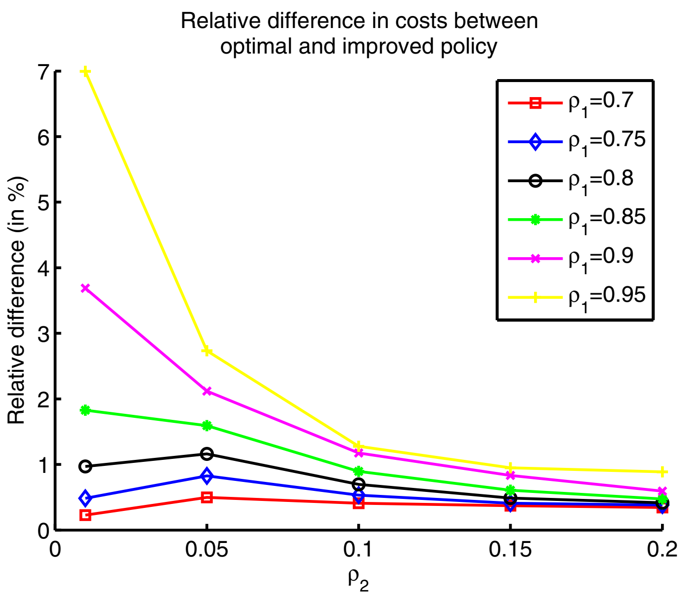

FIGURE 5.8: Again, the relative difference in average costs of the improved policy compared to the optimal policy, but now inside the critical region.

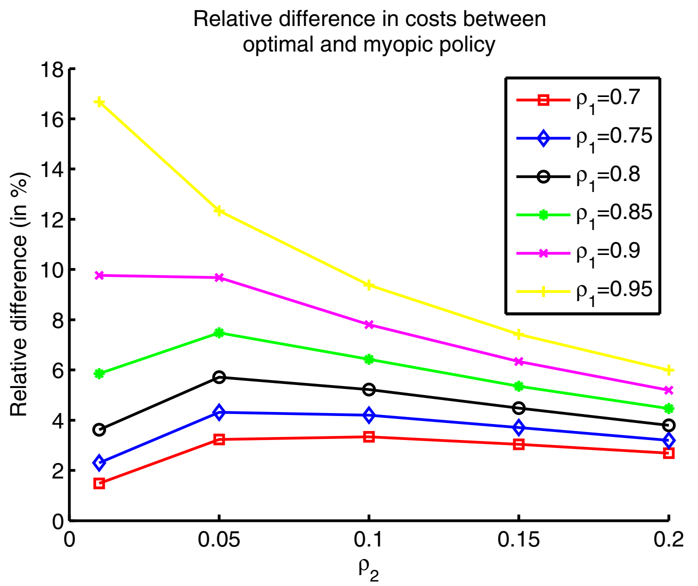

FIGURE 5.9: Again, the relative difference in average costs of the myopic policy compared to the optimal policy, but now inside the critical region.

Finally, Figures 5.8 and 5.9 demonstrate that the relative differences are not monotone. The left-most points (at $\rho_2 = 0.01$) seem to be closer to optimality than the points at $\rho_2 = 0.05$. Further experiments suggest that this is not caused by boundary effects. Also, the differences cannot be explained by the stopping criterion of value iteration, because the differences are too large. Since the observed feature is present in both figures, it seems likely that the optimal policy causes it, and thus that this behavior is a feature of the system. We return to this topic later in Section 5.7.5 when we talk about the optimal policy.

5.7.3 Computational complexity

As described in Section 5.6.2, the improved policy requires four short runs of the value iteration algorithm to determine the parameter $\hat{\alpha}$. The total duration of these runs should be less than the time required to find the optimal policy. Table 5.1 shows the time needed to find $\hat{\alpha}$ for the improved policy, divided by the time required to determine the optimal policy. As parameter values we use the same scenario as in Section 5.7.2, i.e., $\mu_1 = \mu_2 = 0.3$. The two tables clearly demonstrate that determining the improved policy is much faster than finding the optimal policy.

$\rho_1 \rightarrow$	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**
$\rho_2 = 0.01$	0.0122	0.0061	0.0065	0.0059	0.0054	0.0054
$\rho_2 = 0.05$	0.0048	0.0069	0.0068	0.0076	0.0070	0.0069
$\rho_2 = 0.10$	0.0060	0.0058	0.0070	0.0067	0.0078	0.0077
$\rho_2 = 0.15$	0.0056	0.0054	0.0067	0.0063	0.0075	0.0061
$\rho_2 = 0.20$	0.0064	0.0063	0.0061	0.0071	0.0070	0.0068

TABLE 5.1: The run time for determining $\hat{\alpha}$ for the improved policy divided by the run time needed to obtain the optimal policy for various values of ρ_1 (columns) and ρ_2 (rows).

5.7.4 Model complexity

To get a feel for the complexity of the model in Eq. (5.2), we plot a so-called *Ordered Performance Curve* (OPC) [69]. Each point in this plot shows the average costs of a policy that we generate randomly: at each state (i, j, N) we choose action $a = \{Q_1\}$ with probability 0.5, or $a = \{DB\}$ otherwise. By repeating this procedure, we create 2,500 such policies, evaluate them, and plot their average costs in Figure 5.10. Additionally, this figure shows the average costs of the optimal policy and (in our case) of the improved, the age-threshold, and myopic policies. The parameters are $\mu_1 = \mu_2 = 0.3, \rho_1 = 0.8, \rho_2 = 0.1$, based on the critical region in the parameter space. The random policies all perform better than the age-threshold policy, and once again confirm that there is room for improvement on such a traditional policy.

Since the markers of the optimal, improved, and myopic policies are indistinguishable in Figure 5.10, the fifteen best policies are plotted again in Figure 5.11. The steep slope on the left of both figures illustrates that none of the randomly selected policies is able to closely match the performance of the optimal policy. Hence, the plot demonstrates that the performance of the improved policy is not easily replicated by a random policy, and that including the load of the system in the decision policy is meaningful.

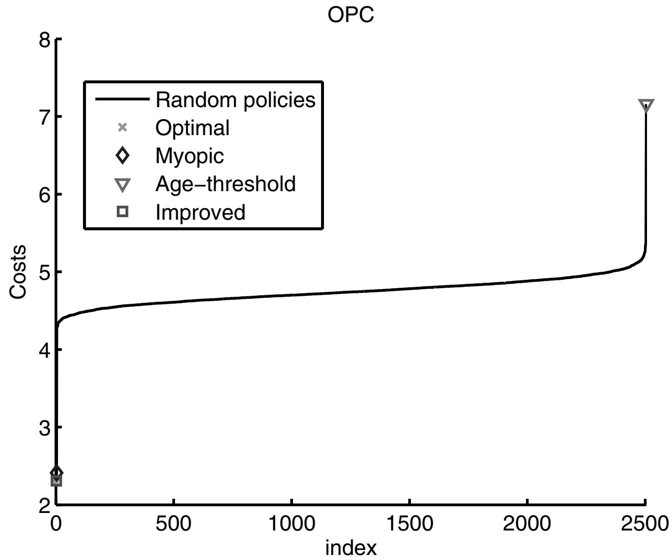

Figure 5.10: Ordered Performance Curve - costs of 2,500 randomly selected policies, as well as the optimal, improved, myopic, and age-threshold policies. The age-threshold policy clearly performs badly.

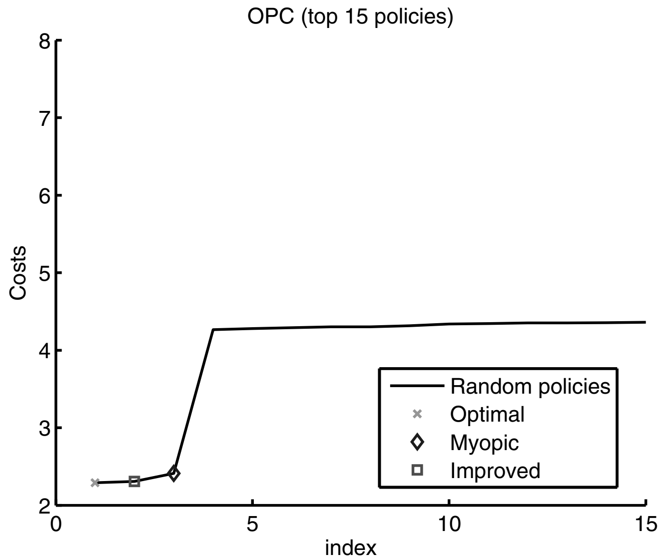

Figure 5.11: The same opc as in Figure 5.10, but now only for the fifteen best policies. The improved and myopic policies are both close to the optimal policy, and perform significantly better than the best random policy.

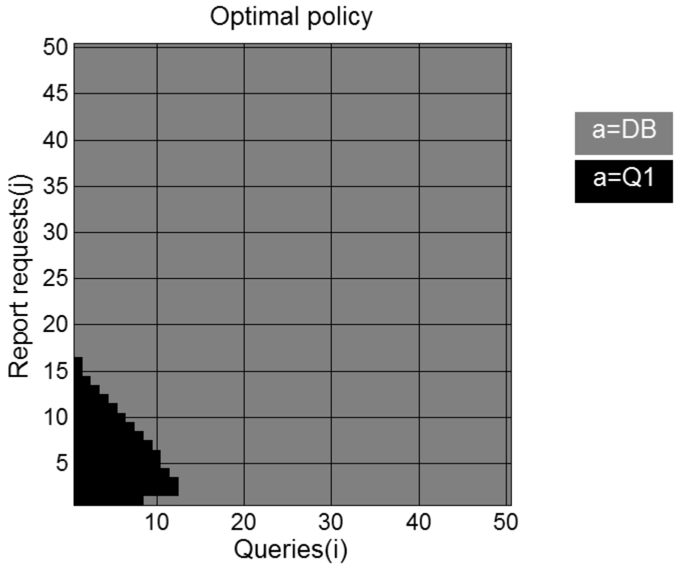

FIGURE 5.12: The optimal policy for $N = 55$, with gray indicating that action $a =$ DB is taken, and black that $a = Q_1$ is taken.

5.7.5 The optimal policy

Next, we inspect the optimal policy in Figures 5.12 and 5.13, again using parameters $\mu_1 = \mu_2 = 0.3, \rho_1 = 0.8, \rho_2 = 0.1$ from the critical region. The first shows a cross-section of the optimal policy at $N = 55$, the second at $N = 120$. Here, for each grid point (i, j) the color gray indicates that action $a =$ DB is taken and black that $a = Q_1$. The figures suggest that (away from the boundaries) the optimal policy is a hyperplane in three-dimensional space, i.e., a switching policy. This observation is supported by intuitions about the problem scenario: once Q_1 reaches a certain load, the controller switches to using the DB, and continue to do so as the load increases. Hence, an optimal policy with a switching structure is in line with our expectations. We were unable to verify this structure mathematically, but we expect that a proof is feasible. The conjecture below formalizes the claim:

CONJECTURE 5.7.1 (Asymptotic switching policy). *The optimal policy for the* MDP *in Eq. (5.2) is a switching curve for N sufficiently large.*

Looking at Figures 5.12 and 5.13, we see that the optimal policy is cropped near the boundary $\{j = 0\}$ of the state space. This effect is caused by the interaction between the number of report requests j and the costs $(N-T)^+$ for

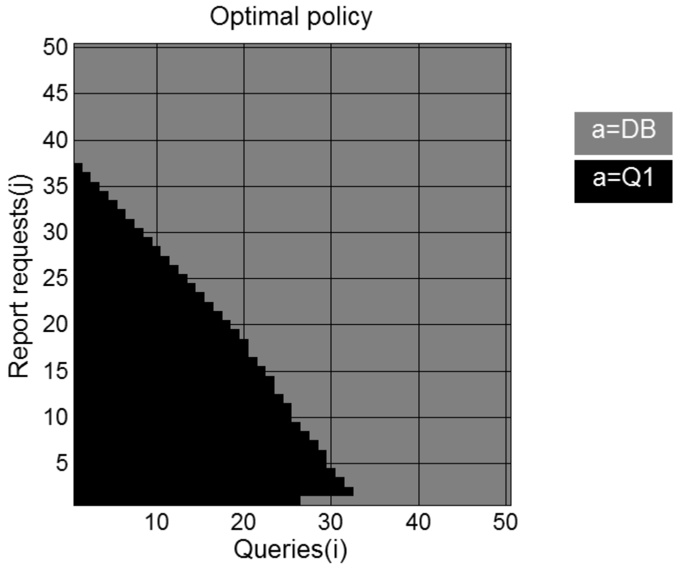

FIGURE 5.13: The optimal policy for $N = 120$, with gray indicating that action $a =$ DB is taken, and black that $a = Q_1$ is taken.

DB assignments. They are connected via N using the term $\mu_2 V(i, j-1, 0)\mathbb{1}_{\{j>0\}}$ in Eq. (5.2), which drops out at the boundary $\{j = 0\}$ of the state space. Consequently, on the boundary the connection between j and N is severed, and changes the structure of the MDP and the optimal policy significantly. This also explains the observation in Section 5.7.2 that the performance of the improved and myopic policies changes for $\rho_2 \approx 0$.

Still, in situations where the boundary $\{j = 0\}$ of the state space is not reached frequently, we expect switching policies to perform well since the boundary effect is relatively small. This is supported by the results on our improved policy and the myopic policy (both are switching policies) in the previous sections.

5.8 Conclusion

In this chapter we investigated the trade-off between data freshness and query response times. We formulated this trade-off as a Markov Decision Process with a three-dimensional state space. The resulting model contained several complication aspects, preventing a derivation of an analytical expression for the optimal policy. Instead, we introduced a three-step approach to finding an approximate policy with near-optimal performance. The first step demonstrated how the original three-dimensional model can be approximated by a simpler two-dimensional model that still captures the important dynamics. Then, in the second step, we described how this simpler model can be solved analytically, using differencing techniques to deal with the inhomogeneous terms. In step three we applied one-step policy improvement to construct our approximate policy. Finally, we numerically demonstrated that this improved policy has near-optimal performance, and significantly outperforms both an age-threshold policy and a myopic policy. Moreover, the experiments reveal that there is room for improvement on the traditional age-threshold policy, which is commonly used in practice. Future research directions include making the improved policy analytic by removing the dependence on short runs of value iteration for determining the parameter α (see also the remark at the end of Section 6.7), and proving the conjecture that the optimal policy is asymptotically a switching curve.

5.9 Appendix: uniqueness

In Section 5.5 we solved the two-dimensional difference equation (5.6), known in MDP literature as the *Poisson equation*. For this equation we have only one boundary condition $V(0,0) = 0$, which is not enough to completely determine the solution. Consequently, after solving the difference equation the constant $\widetilde{V}_1^\alpha(0)$ is yet to be determined in Eq. (5.21).

In order to investigate uniqueness we repeat arguments from Chapter 2 and 4 of [27]. First, note that Eq. (5.6) induces a *Markov cost chain* with transition matrix P, state space $\mathcal{X} = \mathbb{N}_0 \times \mathbb{N}_0$, and cost function

$$c(i,j) = \lambda_1\alpha\left[\gamma_1(i+1) + \gamma_2 j + \gamma_3\right] + \lambda_1(1-\alpha)C.$$

Denote with $\mathbb{B}(\mathcal{X})$ the Banach space of bounded real-valued functions u on $\mathcal{X}$ with the supremum norm, i.e., the norm $||\cdot||$ defined by

$$||u|| = \sup_{(i,j)\in\mathcal{X}} |u(i,j)|\,.$$

Conventional uniqueness proofs for Markov cost chains rely on bounded cost functions contained in $\mathbb{B}(\mathcal{X})$. However, our cost function $c(i,j)$ is unbounded and thus not contained in $\mathbb{B}(\mathcal{X})$. A remedy to this situation is to consider suitable larger Banach spaces instead of $\mathbb{B}(\mathcal{X})$. In order to construct such a space, consider a *weight function* $w : \mathcal{X} \to [1, \infty)$. The *w-norm* is then defined by

$$||u||_w = \sup_{(i,j)\in\mathcal{X}} \frac{|u(i,j)|}{w(i,j)}.$$

A function u is said to be w-bounded if $||u||_w < \infty$, and the space of all w-bounded functions is denoted by $\mathbb{B}_w(\mathcal{X})$. We also define the matrix norm related to $||\cdot||_w$ as $||A||_w = \sup\{||Au||_w : ||u||_w \leq 1\}$. This norm can be rewritten in the following equivalent form (see Eq. (7.2.8) in [67])

$$||A||_w = \sup_{x\in\mathcal{X}} \sum_{y\in\mathcal{X}} \frac{|A_{xy}|\, w(y)}{w(x)}.$$

Finally, we introduce the taboo transition matrix ${}_MP$ as

$${}_MP_{xy} = \begin{cases} P_{xy}, & y \neq M, \\ 0, & y \in M, \end{cases}$$

with $x, y \in \mathcal{X}$ and in our case $M = (0, 0)$. We now state a property and adapted theorem from [27] on uniqueness of solutions of Eq. (5.6).

PROPERTY 5.9.1 (page 19 of [27]). *A Markov chain is called w-geometrically recurrent with respect to M [w-GR(M)] if there exists an* $\epsilon > 0$ *such that* $||{}_MP||_w \leq 1 - \epsilon$.

THEOREM 5.9.2 (Lemma 2.1 combined with Theorem 2.10 of [27]). *Suppose that the Markov chain induced by a policy* π *is unichain, stable, aperiodic, and w-GR(M). Let both* (g, V) *and* (g', V') *be solutions to the Poisson equation. Then* $g = g'$ *and the relative value functions* V *and* V' *differ by only a constant.*

In our case, the Bernoulli policy does indeed induce a Markov chain that is unichain, stable, and aperiodic. The key to ensuring uniqueness is choosing a suitable weight function w such that Property 5.9.1 is satisfied. Section 3.4 of [146] shows that a suitable weight function is of the form

$$w(i,j) = K \prod_{k=1}^{i} (1 + m_k) \prod_{l=1}^{j} (1 + n_l),$$

where $\{m_k\}$, $\{m_l\}$, and K are constants. Unfortunately, the expressions involved are cumbersome and not easy to state explicitly, making it difficult for us to illustrate the construction of the weight function. In the remainder of this section we make an additional assumption that allows us to find a weight function that is explicit. This assumption is only made to facilitate explicitness, and readers interested in the case without the assumption are referred to [146].

Following Section 4.1 of [27], we assume that $\rho_1\alpha + \rho_2 < 1$. The non-zero entries in the transition matrix are given by

$$\begin{aligned}
&P_{(i,j)(i+1,j)} = \lambda_1\alpha,\\
&P_{(i,j)(i,j+1)} = \lambda_2,\\
&P_{(i,j)(i-1,j)} = \mu_1 \mathbb{1}_{\{i>0\}},\\
&P_{(i,j)(i,j-1)} = \mu_2 \mathbb{1}_{\{j>0\}},\\
&P_{(i,j)(i,j)} = 1 - P_{(i,j)(i+1,j)} - P_{(i,j)(i,j+1)} - P_{(i,j)(i-1,j)} - P_{(i,j)(i,j-1)}.
\end{aligned}$$

Set $w(i,j) = (1+k_1)^i(1+k_2)^j$ for some constants k_1 and k_2. Now consider

$$||_M P||_w = \sum_{(i',j')\neq(0,0)} \frac{P_{(i,j)(i',j')}w(i',j')}{w(i,j)},$$

which is given by

$$\begin{cases}
\lambda_1\alpha(1+k_1) + \lambda_2(1+k_2), & (i,j) = (0,0),\\
\lambda_1\alpha k_1 + \lambda_2 k_2 + 1 - \mu_1, & (i,j) = (1,0),\\
\lambda_1\alpha k_1 + \lambda_2 k_2 + 1 - \mu_2, & (i,j) = (0,1),\\
\lambda_1\alpha k_1 + \lambda_2 k_2 + 1 - \dfrac{\mu_1 k_1}{1+k_1}, & i > 1, j = 0,\\
\lambda_1\alpha k_1 + \lambda_2 k_2 + 1 - \dfrac{\mu_2 k_2}{1+k_2}, & i = 0, j > 1,\\
\lambda_1\alpha k_1 + \lambda_2 k_2 + 1 - \dfrac{\mu_1 k_1}{1+k_1} - \dfrac{\mu_2 k_2}{1+k_2}, & i > 0, j > 0.
\end{cases}$$

We need to choose k_1 and k_2 such that all expressions are strictly less than 1. Observe that if the fourth and fifth expression are less than 1, then all others are also satisfied. Hence, we can restrict our attention to the system

$$\begin{aligned}
f_1(k_1,k_2) &= 1 + \lambda_1\alpha k_1 + \lambda_2 k_2 - \frac{\mu_1 k_1}{1+k_1},\\
f_2(k_1,k_2) &= 1 + \lambda_1\alpha k_1 + \lambda_2 k_2 - \frac{\mu_2 k_2}{1+k_2},
\end{aligned}$$

with the assumptions $\lambda_1\alpha + \lambda_2 + \mu_1 + \mu_2 < 1$ and $\rho_1 + \rho_2 < 1$.

Observe that $f_1(0,0) = f_1\big((\mu_1 - \lambda_1\alpha)/(\lambda_1\alpha), 0\big) = 1$. Thus, the points $(0,0)$ and $\big((\mu_1 - \lambda_1\alpha)/(\lambda_1\alpha), 0\big)$ lie on the curve $f_1(k_1,k_2) = 1$. Furthermore, k_2

satisfies $k_2 = \mu_1/\lambda_2 - \mu_1/\big(\lambda_2(1+k_1)\big) - \lambda_1\alpha/\lambda_2$. Note that this function has a maximum value at $k_1 = \sqrt{\mu_1/(\lambda_1\alpha)} - 1$. Hence, this description determines the form of f_1; the curve $f_1(k_1, k_2) = 1$ starts in $(0, 0)$ and increases to an extreme point, and then decreases to the k_1-axis again. The curve f_2 has a similar form, but with the role of the k_1-axis interchanged with the k_2-axis.

The curves determine an area of points (k_1, k_2) such that f_1 and f_2 are strictly less than one if the partial derivative to k_1 at $(0, 0)$ of the curve $f_1(k_1, k_2) = 1$ is greater than the partial derivative to k_2 of the curve $f_2(k_1, k_2) = 1$ at $(0, 0)$. These partial derivatives are given by $(\mu_1 - \lambda_1\alpha)/\lambda_2$ and $\lambda_1\alpha/(\mu_2 - \lambda_2)$, respectively. Since $\rho_1\alpha + \rho_2 < 1$, we have $\lambda_1\alpha\mu_2 + \lambda_2\mu_1 < \mu_1\mu_2$. Adding $\lambda_1\alpha\lambda_2$ to both sides gives $\lambda_1\alpha\lambda_2 < \mu_1\mu_2 - \lambda_1\alpha\mu_2 - \lambda_2\mu_1 + \lambda_1\alpha\lambda_2 = (\mu_1 - \lambda_1\alpha)(\mu_2 - \lambda_2)$. Hence, the relation $\lambda_1\alpha/(\mu_2 - \lambda_2) < (\mu_1 - \lambda_1\alpha)/\lambda_2$ holds. Thus, indeed the partial derivative to k_1 at $(0, 0)$ of the curve $f_1(k_1, k_2) = 1$ is greater than the partial derivative to k_2 of the curve $f_2(k_1, k_2) = 1$ at $(0, 0)$, and there is an area of pairs (k_1, k_2) such that the Markov chain is w-GR(M). For these points it holds that $(1 + k_n) < 1/\rho_n$ for $n = 1, 2$. Observe that any sphere with radius $\epsilon > 0$ around $(0, 0)$ has a non-empty intersection with this area. Hence, the cost function cannot contain terms in i and/or j that grow exponentially fast to infinity, and neither can the relative value function. Consequently, we need to choose $\widetilde{V}_1^{\alpha}(0)$ in Eq. (5.21) such that the exponential term $\big(\frac{\mu_1}{\lambda_1\alpha}\big)^i$ disappears.