REINFORCEMENT LEARNING AND STOCHASTIC OPTIMIZATION

A unified framework for sequential decisions

Warren B. Powell

August 22, 2021

WILEY-INTERSCIENCE

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data:

Optimization Under Uncertainty: A unified framework
Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

PART VI - MULTIAGENT SYSTEMS

Part VI of our book consists of a single chapter on multiagent systems, but this chapter opens up an entirely new line of thinking. This chapter builds entirely on our universal framework, since each agent can be modeled using the same framework we have developed earlier in the book. Decisions made by each agent will draw on the same classes of policies.

We begin by revisiting basic learning problems, but now these are presented using a two-agent model: an environment agent, and a controlling agent. We contrast the resulting model to the approach used by a substantial and mature literature known as "partially observable Markov decision processes" (or POMDPs). We will show that using our approach produces models that are more practical and scalable than those developed in the POMDP literature. We also feel that our approach fixes a fundamental error made in the POMDP literature regarding knowledge of the transition function.

We then transition to systems with multiple controlling agents, where we use different policies to achieve different behaviors. We also introduce the idea that we can model different levels of beliefs about other agents, which spans beliefs about what another agent knows, to beliefs about how they behave. This is a modeling choice rather than a comparison of algorithms to solve a specific model. Multiagent systems open up an entirely new approach for modeling and controlling complex systems.

There is an extensive literature on multiagent systems. The vast majority of applications use fairly trivial policies. At the other extreme are very sophisticated modeling papers that use the POMDP modeling framework and work to achieve some form of optimality as a system. These approaches tend to be limited to relatively small systems.

Our approach is to model each agent using an extended version of our universal modeling framework that accommodates a new dimension, which is communication. We make

decisions using our four classes of policies to find the best policy that can be computed in reasonable time, given the data available to the controlling agent.

We note that our universal modeling framework provides for belief states alongside deterministically known physical and informational parameters and quantities. We have illustrated many applications where belief states were not present. In multiagent systems, belief states are always present since there is always unknown information about other agents. Of course, we may choose to ignore belief state variables, but this will be an explicit modeling choice. However, the ability to learn from decisions will almost always be present, if we wish to take advantage of what we learn.

CHAPTER 20

MULTIAGENT MODELING AND LEARNING

There is a host of problems that are best approached as multiagent systems, where the use of multiple agents allows us to capture a division of knowledge. The simplest example is any learning problem where there is a truth (which we can model as known only to an "environment agent") that needs to be learned by an agent that is making decisions (which we will call a "controlling agent"). However, this is just the beginning of the variety of systems that can be captured by exploiting the concept of multiple agents.

In this chapter, we are going to introduce the fundamental elements of a multiagent model, motivated by applications of increasing complexity. We start with an overview of multiagent systems, where we summarize the dimensions of a multiagent system, outline how to generalize our modeling framework to the multiagent environment, and then cover the area of communication which arises purely because of the presence of multiple agents.

The remainder of the chapter is divided between two-agent systems, and systems with multiple (possibly many) agents.

We begin by showing how to model pure learning problems as two-agent systems, with an environment agent that contains the ground truth, and a controlling agent that has to learn the environment to make decisions. We use the setting of mitigating the flu in a population, and develop a spectrum of models which we then use to illustrate the use of different classes of policies. We contrast our modeling strategy for pure learning problems with an established field known as partially observable Markov decision processes (or POMDPs) where learning problems are modeled and solved as a single system.

We then introduce a two-agent version of the newsvendor problem to illustrate two agents that are ostensibly cooperative, but with different objectives. This application provides a nice setting for learning the behavior of other controlling agents.

The second half of the chapter moves to multiagent systems, beginning with a classical system involving hundreds of independent agents that represent thermostats for apartments in a large building. The chapter closes with a cooperative system involving different hospitals managing, and sharing, blood supplies.

Multiagent systems are a rich problem class, and a single chapter will not be able to cover all the dimensions of multiagent systems. Our goal, rather, is to illustrate how to apply our universal policy, and to illustrate how the four classes of policies can be applied in this context.

20.1 OVERVIEW OF MULTIAGENT SYSTEMS

We begin by describing the dimensions of a multiagent system, followed by a presentation of how to model communication, which is the modeling element not present in our original (single-agent) framework. We then describe how to model a multiagent system, and discuss controlling architectures.

20.1.1 Dimensions of a multiagent system

There are a number of dimensions to multiagent systems. A sample includes:

1) The agents - We start with the list of agents and their capabilities.

2) Learning - This includes:

- Learning the environment.
- Learning about other agents, which includes
  - Learning what they know.
  - Learning about their behavior (specifically, how they make decisions).

3) Communication - Communication consists the following:

- We have to describe which agents can send information to, and receive information from, other agents.
- The speed and capacity of communication between each pair of agents.
- The accuracy of the communication between each pair of agents. Accuracy may be a result of technology (communication errors) or a choice (one agent providing biased information to another).

4) Coordination - This describes any mechanisms used to coordinate the behavior of multiple agents to achieve a common goal.

5) Reward structure - How agents interact depends on how they are rewarded. Competition (or cooperation) is a matter of degree, and may range from zero-sum games (e.g. competition for an open resource), to agents who might be on the same team but with different objectives. Agents with the same reward structure should learn cooperative behaviors.

6) Resources - Agents often have to manage resources. A common example is energy, but an agent may be distributing vaccines, medical supplies, ammunition, food, water, parts, ... We would need to specify:

- Which resources does an agent need to manage?
- How much of the resource the agent can store?
- How much is consumed by the agent itself?
- What are the exogenous demands that have to be satisfied (and how are these learned)?
- How is it replenished (does it return to a home base, refilling stations, or can it be visited by replenishment agents)?

While individual agents can have a wide range of capabilities in the context of an application domain, the capabilities that relate directly to the control of a multiagent system include:

- The environment agent - This agent cannot make any decisions, or perform any learning (that is, anything that implies intelligence). This is the agent that would know the truth about the environment, which includes unknown parameters that we are trying to learn, or which performs the modeling of physical systems that are being observed by other agents. Controlling agents are, however, able to change the environment.

- Controlling agents - These are agents that make decisions that act on other agents, or the ground truth agent (acting as the environment). Controlling agents may communicate information to other controlling and/or learning agents. Controlling agents may also change the environment, or the state (physical, informational and/or belief) of other controlling agents.

- Learning agents - These agents do not make any decisions, but can observe and perform learning (about the ground truth and/or other controlling agents), and communicate beliefs to other agents.

- Replenishment agents - These are agents with the ability to replenish resources. They may be stationary or mobile, where a mobile replenishment agent might be replenished by a stationary replenishment agent. These agents can perform learning and make decisions, which make them similar to controlling agents, but with a narrow set of activities.

There can be many types of agents who make decisions, communicate information and/or perform learning. For example, we might list:

- Single stationary device - Examples include robotic arms used in manufacturing, field cameras, as well as machinery such as a heating and air conditioning system for a building.

- Single mobile device - These can include land robots, flying drones, and underwater vehicles. In time this will also include driverless vehicles.

- Fleets of devices - We may have a group of robots, drones and underwater vehicles, and in time may include fleets of autonomous electric vehicles.

- Individual people in the field - This could be a medical technician making decisions about how to test and/or treat people for disease, a policeman working a neighborhood, or a soldier acting alone. It could even be individuals in a population making decisions about whether protect themselves from exposure to a virus or to get vaccinated.

- Teams of people in the field - This might be a group of people under the control of a single person, as might arise in military operations, medical personnel responding to the outbreak of disease.

- Individual people managing a set of resources for a company - This could be someone assigning locomotives to trains for a specific train yard (or region of the country), a manager making manufacturing and inventory decisions for a single supplier in a supply chain.

- Senior managers making decisions that guide lower level managers - For example, a senior manager might set productivity targets that are used to evaluate field managers who actually assign people to tasks. The term "senior manager" can apply to any decision-maker within a company making decisions about budgets, pricing of products, marketing.

We note that devices and people can both be agents, but they are very different types of agents, since devices will struggle to develop skills that we take for granted in any human. People can develop much more complex behaviors than devices, which introduces a much more complex learning challenge.

20.1.2 Communication

Communication is a characteristic of multiagent systems that does not exist in any form with our basic single agent model. There are a number of dimensions to modeling communication between agents. These include:

- Communication architecture - We have to decide who can communicate what to whom. It will generally not be the case in more complex systems that any agent can (or would) send everything in their state vector S_{tq} to every other agent. We may have coordinating agents that communicate with everyone, make decisions and then send these decisions (in some form) to other agents. Above we introduced the sets $\mathcal{Q}_q^+$ and $\mathcal{Q}_q^-$ that capture the sets of agents that agent q can act on, or those which can act on q, with any type of decision. We can capture just the information architecture using

  $$\begin{aligned} \mathcal{I}_q^+ &= \text{the agents to whom agent } q \text{ can send information to,} \\ \mathcal{I}_q^- &= \text{the agents that can send information to agent } q. \end{aligned}$$

- Active observations - We may choose to observe the environment or other agents, by running a test (e.g. to see who is infected with the flu) or with sensors such as radar. We may observe the location of an agent, resources under the agent's control, and decisions the agent may make. For example, a navy ship has to make the decision whether to turn on its radar to observe another ship, which simultaneously reveals the location of the ship sending the radar.

- Receiving information - If information is sent from q' to q, the agent q has to update their own beliefs, which has to reflect the confidence that agent q has in the information coming from agent q'. Assume that we have just obtained from agent q' updated information $\widehat{W}_{q'qi}$ about the i^{th} element of the state variable S_{qi} for agent q. Let

  $$\begin{aligned} \beta_{qi} &= \text{the precision (one over the variance) of our belief in the estimate } S_{qi}, \\ \beta^W_{q'q} &= \text{the precision in the information flowing from } q' \text{ to } q \text{ (we could make this depend on } i \text{ as well)}, \\ \delta_{q'q} &= \text{the bias introduced by agent } q' \text{ when sending information to agent } q. \end{aligned}$$

  If we receive the information $\widehat{W}_{q'qi}$, we update our estimate S_{qi} and its precision β_{qi} using the formulas we first introduced in section 3.4.1:

  $$S_{qi} \leftarrow \frac{\beta_{qi}S_{qi} + \beta^W_{q'q}\widehat{W}_{q'qi}}{\beta_{qi} + \beta^W_{q'q}}, \tag{20.1}$$

  $$\beta_{qi} \leftarrow \beta_{qi} + \beta^W_{q'q}. \tag{20.2}$$

  Note that our precision $\beta_{q'q}$ depends on both the sending and receiving agents, which means it depends on relationships, rather than just the reliability of either agent.

- Sending information - We have to model the act of sending information in S_{tq} to another agent q'. The information may be sent accurately, or with some combination of noise and bias.

- Signal distortion - We have to capture the presence of noise which reflects the difference between what is being sent, and what is received. Signal distortion can arise in two ways:

  - Passive distortion - This is a byproduct of technology (the communication channel may introduce noise) and environment (weather or exposure to magnetic fields can introduce noise).
  - Active distortion - This is where the sending agent intentionally distorts the information being sent. There are different types of active distortion:
    - Active noise - This is where the sending agent adds a zero-mean noise term to hide the true mean.
    - Active bias - The sending agent may intentionally bias a signal for any of a variety of reasons.

20.1.3 Modeling a multiagent system

To model our system, we begin by using our standard vocabulary from our universal framework and simply add the index q (which we use to index agents). However, we start with the architecture that describes how they communicate and interact, which is a new element to our modeling framework that only arises with multiagent systems.

The agent architecture:
We begin by describing the set of agents which we model using:

$$\begin{aligned}
\mathcal{Q} &= \text{The set of agents,} \\
\mathcal{Q}_q^+ &= \text{the set of agents } q' \text{ that agent } q \text{ can affect with a decision } x_{tqq'}, \\
&\quad \text{where these decisions can represent sending information, money, or} \\
&\quad \text{physical resources.}
\end{aligned}$$

We will use our conventional notation for capturing constraints on the flow of physical resources. The flow of information, on the other hand, is described by bandwidth constraints (say, bits per second), as well as the reliability of the information. For now, since information is used to update our beliefs B_{tq}, we are going to introduce the vector $\zeta_{qq'}$:

$$\begin{aligned}
\zeta_{qq'i} &= \text{the vector of parameters governing the speed, capacity and reliability of the} \\
&\quad \text{information about data element } i \text{ that is communicated from } q \text{ to any agent} \\
&\quad q' \in \mathcal{Q}_q^+, \\
&= (\beta_{qq'i}, \delta_{qq'i}, \eta_{qq'i}), \text{ where:} \\
\beta_{qq'i} &= \text{the precision (inverse of the variance) of the reliability of the information} \\
&\quad \text{sent from } q \text{ to } q' \text{ in terms of its accuracy of describing some data element } i, \\
\delta_{qq'i} &= \text{the bias in the information sent about data element } i \text{ sent from } q \text{ to } q', \\
\eta_{qq'i} &= \text{the energy required to send information about data element } i \text{ from } q' \text{ to } q.
\end{aligned}$$

Communication is an entirely new dimension to our modeling framework, since this does not arise at all in the context of a single agent system.

State variables:
As with single agent systems, each agent has a state variable with the same three classes of information:

$$\begin{aligned}
R_{tq} &= \text{The state of resources controlled by agent } q \text{ at time } t. \\
I_{tq} &= \text{Any other information known to agent } q \text{ at time } t. \\
B_{tq} &= \text{The beliefs of agent } q \text{ about anything known to any other agent} \\
&\quad \text{(and therefore not known to agent } q\text{). This covers parameters in the} \\
&\quad \text{ground truth, anything known by any other agent (for example, the} \\
&\quad \text{resources that an agent } q' \text{ might be controlling), and finally, beliefs} \\
&\quad \text{about how other agents make decisions.}
\end{aligned}$$

Belief states are the richest and most challenging dimension of multiagent systems, especially when there is more than one controlling agent, as would occur in competitive games.

Decision variables:
Since communication and learning is such a fundamental component of multiagent systems, we are going to introduce new decision variables specifically for the purpose of capturing information sent from an agent q to another agent q':

$$\begin{aligned}
z_{tqq'i} &= \text{Information about data element } i \text{ sent from agent } q \text{ at time } t \text{ to agent} \\
&\quad q' \text{ arriving at time } t+1, \\
z_{tqi} &= (z_{tqq'i})_{q' \in \mathcal{Q}_q^+}.
\end{aligned}$$

The reliability, bias and energy required to send information are described by $\zeta_{qq'i}$.

In addition, we are still going to have our traditional "x" variables that will capture the movement of physical resources, and will still continue to control (or influence) observations that we make, as we have done previously:

$$\begin{aligned} x_{tq} &= \text{Decisions (scalar or vector) of decisions made by agent } q\text{, which might include:} \\ x_{tqq'} &= \text{Decisions made by agent } q \text{ that act on agent } q'. \end{aligned}$$

Whereas we would require that $x_t \in \mathcal{X}_t$ for our single-agent notation, we now use $x_{tq} \in \mathcal{X}_{tq}$ which might simply require that x_{tq} be drawn from a finite set (say, of drugs, or choice of people), or it can represent flow conservation constraints that we can write using matrix notation as

$$\begin{aligned} A_{tq}x_{tq} &= R_{tq}, \\ x_{tq} &\leq u_{tq}, \\ x_{tq} &\geq 0. \end{aligned}$$

Decisions are made by the policy $X^\pi_q(S_{tq})$.

If $\mathcal{X}_t$ is our feasible region on our decisions x_{tq}, how do we represent constraints on the flow of information? The simple reality is: there are no constraints on the flow of information. Normally, we would expect that $z_{tqq'}$ would contain at least a noisy estimate of one or more of the variables in our state S_{tq} (remember that S_{tq} is our state of knowledge), but this is not necessary. Agents can transmit information that is simply incorrect, and this happens in adversarial settings where an agent may wish to misinform an adversary. Of course, we see this happening all the time on social media!

A new dimension in the formulation of decisions that is introduced in the study of multiagent systems is the decision to communicate. We have already seen decisions to observe in chapter 7, where we choose x^n and then observe a function $\hat{F}^n = F(x^n, W^{n+1})$. However, now we have the ability to choose from what we know (in our state S_t at time t) and then communicate it with some level of precision to another agent.

Exogenous information variables:
Exogenous information to an agent may come from outside of our system, or from another agent:

$$\begin{aligned} W_{tq} &= \text{Exogenous information arriving to agent } q \text{ from any exogenous source, which may include:} \\ W_{tqq'} &= \text{Information arriving to agent } q \text{ from agent } q'. \end{aligned}$$

The information in $W_{tqq'}$ will typically be a byproduct of a decision $x_{tqq'}$. It is important to be clear about the timing of when the information from a decision $x_{tqq'}$ made at time t arrives to agent q'. We are going to assume that it arrives at time $t+1$.

Transition function:
This is simply

$$S_{t+1,q} = S^M_q(S_{tq}, x_{tq}, W_{t+1,q}),$$

which we have used throughout the book. Now, we have a transition function for each agent q.

Objective function:
We assume that each agent has a performance metric that we write as

$$C_q(S_{tq}, x_{tq}) \quad = \quad \text{the contribution earned by agent } q \text{ from being in state } S_{tq} \text{ and making decision } x_{tq},$$

where $x_{tq} = X_q^\pi(S_{tq})$.

We then face the problem of optimizing across policies. This is not as straightforward as it is with single agents. Two possible optimization mechanisms include:

- Each agent optimizes their own policy - While this fits in the framework we have covered in the rest of the book, we still encounter the reality that all the agents are presumably searching for policies at the same time. Since the policy of an agent q' can influence the exogenous information $W_{t+1,q'q}$ that arrives to agent q, it means that the exogenous information processes are evolving as part of the policy search process (whether it is online or offline).

- A single optimizing process (not necessarily an agent) may be managing the process of searching over all the policies in search of a globally optimizing set of policies. Keep in mind that the policy used by each agent q can only depend on what agent q knows, which is captured by their state variable S_{tq}.

Since there are so many mechanisms for searching for policies, we are simply going to introduce the problem of searching for policies. Keep in mind that while we may, of course, be searching across policy classes, we think it is more common that the class of policy will have been chosen for each agent, leaving the optimization to consist of any tunable parameters.

There is a tendency in the literature on multiagent systems to work with a "system state" $S_t = (S_{tq})_{q \in \mathcal{Q}}$. We take the position that this is meaningless, since no agent ever sees all this information, including a central controlling agent, which is not allowed to see the states S_{tq} of the individual agents (although there can be information sharing). We would approach the modeling of each agent as its own system, with the understanding that a challenge of any intelligent agent is to develop models that help the agent to forecast the exogenous information process W_{tq}. Of course, this depends on the policy being used by the agent, and the anticipated behaviors of other agents.

20.1.4 Controlling architectures

Now that we have a model, it is worth discussing designing policies. In our single agent problems up to now, we assume that we have a single performance metric that we are optimizing. Multiagent systems are more complex. We still assume that each agent has a single objective, but it has the ability to influence the behavior of other agents, which means an agent can change the environment it works in to improve its own performance. The structure of a system influences how each agent behaves.

Given the diversity of types of agents, it should not be surprising that there is a wide range of multiagent systems. For this reason, we list a few examples of systems that illustrate interesting multiagent settings:

- Learning systems (controlling agent and environment agent) - A controlling agent can learn an environment, but can also modify the environment to help its own performance. We illustrate this in section 20.2 using the context of mitigating the spread of the flu in a population.

- Two-agent adversarial systems (games) - These often describe zero-sum games, but may include semi-cooperative settings where two agents interact with different objectives (but not necessarily completely opposite). We illustrate this using a semi-cooperative game we call the two-agent newsvendor problem (see section 20.4).
- Oligopolistic systems - These often arise in markets where there is a small number of players (say, three to five) which are more complex than two-agent systems, but small enough that the effect of individual players on a market can be discerned.
- Multiple independent agents - This describes systems where each agent is behaving completely independently of every other agent, but using a policy that has been chosen to achieve a system-wide goal. We illustrate this in section 20.5 using a collection of thermostats in a building.
- Multiple cooperating agents - This can describe teams of people working together, or groups of suppliers who make up the supply chain for a product. We use the setting of coordinating the blood supplies managed by different hospitals in section 20.6.
- Hierarchical systems - These would describe central (or centralized) managers controlling field agents, or even layers of managers controlling the next lower layer, as arises in the military or large companies. In this context, agents typically have to balance their local performance (which can include the performance of the agents below them) with following the guidance of higher level agents (ideally these are aligned, but as we all know...).

20.2 A LEARNING PROBLEM - FLU MITIGATION

We are going to use the problem of protecting a population against the flu as an illustrative example of a learning problem. It will start as a learning problem with an unknown but controllable parameter, which is the prevalence of the flu in the population. We will use this to illustrate different classes of policies, after which we will propose several extensions.

20.2.1 A static model

Let μ be the prevalence of the flu in the population (that is, the fraction of the population that has come down with the flu). In a static problem where we have an unknown parameter μ, we make observations using

$$W_{t+1} = \mu + \varepsilon_{t+1}, \tag{20.3}$$

where the noise $\varepsilon_{t+1} \sim N(0, \sigma_W^2)$ is what keeps us from observing μ perfectly.

We express our belief about μ by assuming that $\mu \sim N(\bar{\mu}_t, \bar{\sigma}_t^2)$. Since we fix the assumption of normality, we express our belief about μ as $B_t = (\bar{\mu}_t, \bar{\sigma}_t^2)$. We are again going to express uncertainty using $\beta_t = 1/\bar{\sigma}_t^2$ which is the precision of our estimate of μ, and $\beta^W = 1/\sigma_W^2$ is the precision of our observation noise ε_{t+1}.

We need to estimate the number of people with the disease by running tests, which produces the noisy estimate W_{t+1}. We represent the decision to run a test by the decision variable x_t^{obs} where

$$x_t^{obs} = \begin{cases} 1 & \text{if we observe the process and obtain } W_{t+1}, \\ 0 & \text{if no observation is made.} \end{cases}$$

If $x_t^{obs} = 1$, then we observe W_{t+1} which we can use to update our belief about μ using

$$\bar{\mu}_{t+1} = \frac{\beta_t \bar{\mu}_t + \beta^W W_{t+1}}{\beta_t + \beta^W}, \tag{20.4}$$

$$\beta_{t+1} = \beta_t + \beta^W. \tag{20.5}$$

If $x_t^{obs} = 0$, then $\bar{\mu}_{t+1} = \bar{\mu}_t$ and $\beta_{t+1} = \beta_t$.

For this problem our state variable is our belief about μ, which we write

$$S_t = B_t = (\bar{\mu}_t, \beta_t).$$

If this was our problem, it would be an instance of a one-armed bandit. We might assess a cost for making an observation, along with a cost of uncertainty. For example, assume we have the following costs:

$$\begin{aligned} c^{obs} &= \text{The cost of sampling the population to estimate the number of people infected with the flu,} \\ C^{unc}(S_t) &= \text{the cost of uncertainty,} \\ &= c^{unc}\bar{\sigma}_t, \\ C(S_t, x_t) &= c^{obs}x_t^{obs} + C^{unc}(S_t). \end{aligned}$$

Using this information, we can put this model in our canonical framework as follows:

State variables - $S_t = (\bar{\mu}_t, \beta_t)$.

Decision variables - $x_t = x_t^{obs}$ determined by our policy $X^{obs}(S_t)$ (to be determined later).

Exogenous information - W_{t+1} which is our noisy estimate of how many people have the flu from equation (20.3) (and we only obtain this if $x^{obs} = 1$).

Transition function - Equations (20.4) and (20.5).

Objective function - We would write our objective as

$$\max_{\pi} \mathbb{E}\left\{\sum_{t=0}^{T} C(S_t, x_t)|S_0\right\}. \tag{20.6}$$

We now need a policy $X^{obs}(S_t)$ to determine x_t^{obs}. We can use any of the four classes of policies described in section 7.3 or chapter 11. We sketch examples of policies in section 20.2.5 below.

20.2.2 Variations of our flu model

We are going to present a series of variations of our flu model to bring out different modeling issues. We refer to our first model above as "model 1"; the remaining models are

2) A time-varying model.

3) A time-varying model with drift.

4) A dynamic model with a controllable truth.

5) A flu model with a resource constraint and exogenous state.

6) A spatial model.

These variations are designed to bring out the modeling issues that arise when we have an evolving truth (with known dynamics), an evolving truth with unknown dynamics (the drift), an unknown truth that we can control (or influence), followed by problems that introduce the dimension of having a known and controllable physical state.

2) A time-varying model - If the true prevalence of the flu is evolving exogenously (as we would expect in this application), then we would write the true parameter as depending on time, μ_t, which might evolve according to

$$\mu_{t+1} = \max\{0, \mu_t + \varepsilon^{\mu}_{t+1}\}, \tag{20.7}$$

where $\varepsilon^{\mu}_{t+1} \sim N(0, \sigma^{\mu,2})$ describes how our truth is evolving. If the truth evolves with zero mean and known variance $\sigma^{\varepsilon,2}$, our belief state is the same as it was with a static truth (that is, $S_t = (\bar{\mu}_t, \beta_t)$). What does change is the transition function which now has to reflect both the noise of an observation ε_{t+1} as well as the uncertainty in the evolution of the truth, captured by ε^{μ}_{t+1}.

Remark: When μ was a constant, we did not have a problem referring to it as a parameter, whereas the state of our system is the belief which evolves over time (state variables should only include information that changes over time). When μ is changing over time, in which case we write it as μ_t, then it is more natural to think of the value of μ_t as the state of the system, but not observable to the controller. For this reason, many authors would refer to μ_t as a *hidden state*. However, we still have the belief about μ_t, which creates some confusion: What is the state variable? We are going to resolve this confusion below.

3) A time-varying model with drift - Now assume that

$$\varepsilon^{\mu}_{t+1} \sim N(\delta, \sigma^{\varepsilon,2}).$$

If $\delta \neq 0$, then it means that μ_t is drifting higher or lower (for the moment, we are going to assume that δ is a constant). We do not know δ, so we would assign a belief such as

$$\delta \sim N(\bar{\delta}_t, \bar{\sigma}^{\delta,2}_t).$$

Again let the precision be given by $\beta^{\delta}_t = 1/\bar{\sigma}^{\delta,2}_t$.

We might update our estimate of our belief about δ using

$$\hat{\delta}_{t+1} = W_{t+1} - W_t.$$

Now we can update our estimate of the mean and variance of our belief about δ using

$$\bar{\delta}_{t+1} = \frac{\beta^{\delta}_t \bar{\delta}_t + \beta^W \hat{\delta}_{t+1}}{\beta^{\delta}_t + \beta^W}, \tag{20.8}$$

$$\beta^{\delta}_{t+1} = \beta^{\delta}_t + \beta^W. \tag{20.9}$$

In this case, our state variable becomes

$$S_t = B_t = \big((\bar{\mu}_t, \beta_t), (\bar{\delta}_t, \beta^{\delta}_t)\big).$$

Here, we are modeling only the belief about μ_t, while μ_t itself is just a dynamically varying parameter. This changes in the next example.

4) A dynamic model with a controllable truth - Now consider what happens when our decisions might actually change the truth μ_t. Let

x_t^{vac} = the number of vaccination shots we administer in the region.

We assume that the vaccination shots reduce the presence of the disease by θ^{vac} for each vaccinated patient, which is x_t^{vac}. We are going to assume that the decision made at time t is not implemented until time $t+1$. This gives us the following equation for the truth

$$\mu_{t+1} = \max\{0, \mu_t - \theta^{vac} x_{t-1}^{vac} + \varepsilon_{t+1}^{\mu}\}. \tag{20.10}$$

We express our belief about the presence of the disease by assuming that it is Gaussian where $\mu_t \sim N(\bar{\mu}_t, \sigma_t^2)$. Again letting the precision be $\beta_t = 1/\sigma_t^2$, our belief state is $B_t = (\bar{\mu}_t, \beta_t)$, with transition equations similar to those given in equations (7.26) and (7.27) but adjusted by our belief about what our decision is doing. If we make an observation (that is, if $x_t^{obs} = 1$), then

$$\bar{\mu}_{t+1} = \frac{\beta_t(\bar{\mu}_t - \theta^{vac} x_{t-1}^{vac}) + \beta^W W_{t+1}}{\beta_t + \beta^W}, \tag{20.11}$$

$$\beta_{t+1} = \beta_t + \beta^W. \tag{20.12}$$

If $x_t^{obs} = 0$, then $\bar{\mu}_{t+1} = \bar{\mu}_t - \theta^{vac} x_{t-1}^{vac}$, and $\beta_{t+1} = \beta_t$.

This setting introduces a modeling challenge: Is the state μ_t? Or is it the belief $(\bar{\mu}_t, \beta_t)$? When μ_t was static or evolved exogenously, it seemed clear that the state was our belief about μ_t. However, now that we can control μ_t, it seems more natural to view μ_t as the state. This problem is an instance of a partially observable Markov decision problem. Later we are going to review how the POMDP community models these problems, and offer a different approach.

This problem has an unobservable state that is controllable. The next two problems will introduce the dimension of combining both observable and unobservable states that are both controllable.

5) A flu model with a resource constrained and exogenous state - Now imagine that we have a limited number of vaccinations that we can administer. Let R_0 be the number of vaccinations we have available. Our vaccinations x_t^{vac} have to be drawn from this inventory. We might also introduce a decision x_t^{inv} to add to our inventory (at a cost). This means our inventory evolves according to

$$R_{t+1} = R_t + x_{t-1}^{inv} - x_{t-1}^{vac},$$

where we require $x_{t-1}^{vac} \leq R_t$. We still have our decision of whether to observe the environment x_t^{obs}, so our decision variables are

$$x_t = (x_t^{inv}, x_t^{vac}, x_t^{obs}).$$

While we are at it, we might as well include information about the weather such as temperature I_t^{temp} and humidity I_t^{hum} which can contribute to the spread of the flu. We would model these in our "other information" variable

$$I_t = (I_t^{temp}, I_t^{hum}).$$

Our state variable becomes

$$S_t = \big(R_t, (I_t^{temp}, I_t^{hum}), (\bar{\mu}_t, \beta_t)\big). \tag{20.13}$$

We now have a combination of a controllable physical state R_t that we can observe perfectly, exogenous environmental information $I_t = (I_t^{temp}, I_t^{hum})$, and the belief state $B_t = (\bar{\mu}_t, \beta_t)$ which captures our distribution of belief about the controllable state μ_t that we cannot observe.

Note how quickly our solvable two-dimensional problem just became a much larger five-dimensional problem. This is a big issue if we are trying to use Bellman's equation, but only if we are using a lookup table representation of the value function (otherwise, we do not care).

6) A spatial model - Imagine that we have to allocate our supply of flu vaccines over a set of regions $\mathcal{I}$. For this problem, we have a truth μ_{ti} and belief $(\bar{\mu}_{ti}, \beta_{ti})$ for each region $i \in \mathcal{I}$. Next assume that x_{ti}^{vac} is the number of vaccines allocated to region i, which is subject to the constraint

$$\sum_{i \in \mathcal{I}} x_{ti}^{vac} \leq R_t. \tag{20.14}$$

Our inventory R_t now evolves according to

$$R_{t+1} = R_t + x_t^{inv} - \sum_{i \in \mathcal{I}} x_{ti}^{vac}.$$

The coupling constraint (20.14) prevents us from solving for each region independently. This produces the state variable

$$S_t = \big(R_t, (\bar{\mu}_{ti}, \beta_{ti})_{i \in \mathcal{I}}\big). \tag{20.15}$$

What we have done with this extension is to create a state variable that is potentially very high dimensional, since spatial problems may easily range from hundreds to thousands of regions.

20.2.3 Two-agent learning models

There are two perspectives that we can take in any learning problem: one from the perspective of the environment, and one from the perspective of the controller that makes decisions:

The environment perspective - The environment (sometimes called the "ground truth") knows μ_t, but cannot make any decisions (nor does it do any learning).

The controller perspective - The controller makes decisions that affect the environment, but is not able to see μ_t. Instead, the controller only has access to the belief about μ_t.

The model of the environment agent is given in figure 20.1. The model of the controlling agent is given in figure 20.2.

It is best to think of the two perspectives as agents, each working in their own world. There is the "environment agent" which does not make decisions, and the "controlling

State variables $S_t^{env} = \big(\mu_t, \delta\big)$ (we include the drift δ, even if it is not changing).

Decision variables There are no decisions.

Exogenous information $W_{t+1}^{env} = \varepsilon_{t+1}^{\mu}$.

Transition function $S^{env} = S^{M,env}(S_t^{env},, W_{t+1}^{env})$, which includes equation (20.7) describing the evolution of μ_t.

Objective function Since there are no decisions, we do not have an objective function.

Figure 20.1 The canonical model of the environment.

State variables $S_t^{cont} = \big((\bar{\mu}_t, \beta_t), (\bar{\delta}_t, \beta_t^{\delta})\big)$.

Decision variables $x_t = (x_t^{vac}, x_t^{obs})$.

Exogenous information W_{t+1}^{cont} which is our noisy estimate of how many people have the flu (and we only obtain this if $x_t^{obs} = 1$).

Transition function $S_{t+1}^{cont} = S^{M,cont}(S_t^{cont}, x_t, W_{t+1}^{cont})$, which consist of equations (20.11) and (20.12).

Objective function We can write this in different ways. Assuming we are implementing this in a field situation, we want to optimize cumulative reward. Let:

c^{obs} = The unit cost of sampling the population to estimate the number of people infected with the flu.

$C^{vac}(\bar{\mu}_t)$ = The cost we assess when we think that the number of infected people is $\bar{\mu}_t$.

Now let $C^{cont}(S_t, x_t) = c^{obs}x_t^{obs} + C^{vac}(\bar{\mu}_t)$ be the cost at time t when we are in state S_t and make decision x_t (note that x_t^{vac} impacts S_{t+1}). Finally, we want to optimize

$$\max_{\pi} \mathbb{E}\left\{\sum_{t=0}^{T} C^{cont}(S_t, X_t^{\pi}(S_t))|S_0\right\}. \tag{20.16}$$

Figure 20.2 The canonical model of the controlling agent.

agent" which makes decisions, and performs learning about the environment that cannot be observed (such as μ_t). Once we have identified our two agents, we need to define what is known by each agent. This begins with who knows what about parameters such as μ_t, but it does not stop there.

Table 20.1 shows the environmental state and controlling state variables for each of the variations of our flu problem that we presented in section 20.2.2. A few observations are useful:

- The two-agent perspective means we have two systems. The environment agent is a simple system with no decisions, but with access to μ_t and the dynamics of how vaccinations affect μ_t. The state of the system for the environment agent is S_t^{env} which includes μ_t. The state for the system for the controlling agent, S_t^{cont}, is the belief about μ_t, along with any other information known to the controlling agent such as R_t. The two systems are completely distinct, beyond the ability to communicate.

Model	S_t^{env}	S_t^{cont}	Description
1)	(μ_t)	$(\bar{\mu}_t, \beta_t)$	Static, unknown truth
2)	$\big((\mu_t), (I_t^{temp}, I_t^{hum})\big)$	$\big(R_t, (I_t^{temp}, I_t^{hum}), (\bar{\mu}_t, \beta_t)\big)$	Resource constrained with exogenous information
3)	$\big(\mu_t, \delta\big)$	$\big((\bar{\mu}_t, \beta_t), (\bar{\delta}_t, \beta_t^{\delta})\big)$	Dynamic model with uncertain drift
4)	$\big(\mu_t, x_{t-1}^{vac}, \theta^{vac}\big)$	$\big(\bar{\mu}_t, \beta_t\big)$	Dynamic model with a controllable truth
5)	$\big((\mu_{ti})_{i\in\mathcal{I}}, x_{t-1}^{vac}, \theta^{vac}\big)$	$\big(R_t, (\bar{\mu}_t, \beta_t)\big)$	Resource constrained model
6)	$\big((\mu_{ti})_{i\in\mathcal{I}}, x_{t-1}^{vac}, \theta^{vac}\big)$	$\big(R_t, (\bar{\mu}_{ti}, \beta_{ti})_{i\in\mathcal{I}}\big)$	Spatially distributed model

Table 20.1 Environmental state variables and controller state variables for different models.

- In model 2, we model the temperature I_t^{temp} and humidity I_t^{hum} as state variables for both the environment, which presumably would control changes to these variables, and the controlling agent, since we have assumed that the controlling agent is able to observe these perfectly. We could, of course, insist that the controller can only observe these through imperfect instruments, in which case they would be handled in the same way we handle μ_t.

- Normally a state variable S_t should only include information that changes over time (otherwise the information would go in the initial state S_0). For this presentation, we included information such as the drift δ (model 3) and the effect of vaccinations on the prevalence of the flu θ^{vac} (model 4) in the environmental state variables to indicate information known to the environment but not to the controlling agent.

- In model 4, we include the decision x_{t-1}^{vac} in the state variable for the environment. We assume that the controlling agent makes the decision to vaccinate x_{t-1}^{vac} at time $t-1$ which is then communicated to the environment (which is how it gets added to S_t^{env}) and is then implemented during time period t. The information arrives to the environment through the exogenous information variable W_t^{env}.

- For models 5 and 6, we see how quickly we can go from two or three dimensions, to hundreds or thousands of dimensions. The spatially distributed model cannot be solved using standard discrete representations of state spaces, but approximate dynamic programming has been used for very high-dimensional resource allocation problems (see chapter 18).

In addition to modeling what each agent knows, we have to model communication. This will become an important issue when we model multiple controlling agents which we address in section 20.5. For our problem with a single controlling agent and a passive environment, there are only two types of communication: 1) the ability of the controlling agent to observe the environment (with noise) and 2) the communication of the decision x_t^{vac} to the environment.

It is not hard to see that *any* learning problem can (and we claim should) be presented using this "two-agent" perspective.

20.2.4 Transition functions for two-agent model

Our two-agent model has focused on what each agent knows (the state variable), but there is another dimension that deserves a closer look, which is the transition function itself. Assume that the true model describing the evolution of μ_t (known only to the environment) is

$$\begin{aligned}\mu_{t+1} &= \theta_0^\mu \mu_t + \theta_{24}^\mu \mu_{t-24} + (\theta_0^{temp} U_t + \theta_1^{temp} U_{t-1} + \theta_2^{temp} U_{t-2}) \\ &\quad -(\theta_1^{vac} x_{t-1}^{vac} + \theta_2^{vac} (x_{t-1}^{vac})^2) + \varepsilon_{t+1}^\mu. \end{aligned} \tag{20.17}$$

where

$$U_t = \left(\max\{0, I_t^{temp} - I^{threshold}\} \right)^2$$

where $I^{threshold}$ is a threshold temperature (say, 25 degrees F) below which colds and sneezing begins to spread the flu. The inclusion of temperature over the current and two previous time periods captures lag in the onset of the flu due to cold temperatures.

For certain classes of policies, the controlling agent needs to develop its own model of the evolution of the flu. The controlling agent would not know the true dynamics in equation (20.17) and might instead use the following time-series model for the observed number of flu cases W_t:

$$W_{t+1} = \theta_0^W W_t + \theta_1^W W_{t-1} + \theta_2^W W_{t-2} - \theta^{vac} x_{t-1}^{vac} + \varepsilon_{t+1}^W. \tag{20.18}$$

Our model in equation (20.18) is a reasonable time-series model for the sequence of observations $(W_1, \ldots, W_t)$ to predict W_{t+1}. There are, however, several errors in this model:

- The controlling agent is using observations W_t, W_{t-1} and W_{t-2} while the environment uses μ_t, which is not observable to the controller.
- The controlling agent did not realize there was a 24-hour lag in the development of the flu.
- The controlling agent is ignoring the effect of temperature.
- The controlling agent is not properly capturing the effect of vaccinations on infections.

Just the same, ignorance is bliss and our controlling agent moves forward with their best effort at modeling the evolution of the flu. Assume that the time-series model (20.18) is a reasonable fit of the data. We suspect that a careful examination of the errors (they should be independent and identically distributed) might fail a proper statistical test, but it is also possible that we cannot reject the hypothesis that the errors do satisfy the appropriate conditions. This does not mean that the model is correct; it just means that we do not have the data to reject it.

Now imagine that a graduate student is writing a simulator for the flu model, and assume that there is only one person writing the code (which is typically what happens in practice). Our erstwhile graduate student will create the true transition equation (20.17). When she goes to create the transition model used by the controller, she would create the best approximation possible given the information she was allowed to use, but she would know immediately that there are a number of errors in her approximation. This would allow her to declare that this model is "non-Markovian," but it is only because she is using her knowledge of the true model.

We offer the observation that almost all statistical models (such as equation (20.18)) are just approximations, which means given enough data, we would be able to argue that there is some violation (typically that the errors are independent and identically distributed). Someone developing this model might insist, without any data, that the approximate transition function (20.18) is "non-Markovian" by simply comparing the model to the "true" model (20.17), which is unknown to the controller (but known to the modeler). In effect, the modeler is cheating by using her knowledge of the true model, which simply would not happen in practice (this is a true story).

20.2.5 Designing policies for the flu problem

Once we formulate our models of each agent, we need to design policies for the controlling agent. The creation of effective, high quality policies can be major projects. What we want to do is to sketch examples of each of the four classes of policies to help reinforce why it is important to understand all four classes.

Policy function approximations - Policy function approximations are analytical functions that map states to actions. Of the four classes of policies, this is the only class that does not involve an imbedded optimization problem.

For our flu problem, it is common to use the structure of the problem to identify simple functions for making decisions. For example, we might use the following rule for determining whether to make an observation of the environment:

$$X^{pfa-obs}(S_t|\theta^{obs}) = \begin{cases} 1 & \bar{\sigma}_t/\bar{\mu}_t \geq \theta^{obs}, \\ 0 & \text{otherwise.} \end{cases} \tag{20.19}$$

The policy captures the intuition that we want to make an observation when the level of uncertainty (captured by the standard deviation of our estimate of the true prevalence), relative to the mean, is over some number. The parameter θ^{obs} has to be tuned, which we do using the objective function (20.6). A nice feature of the tunable parameter is that it is unitless.

To determine x_t^{vac}, we might set μ^{vac} as a target infection level, and then vaccinate at a level that we believe (or hope?) that we get down to the target. To do this, first compute

$$\zeta_t(\theta^{vac}) = \frac{1}{\theta^{vac}} \max\{0, \big(\bar{\mu}_t - \mu^{vac}\big)\}.$$

We can view ζ_t as the distance to our goal μ^{vac}. This calculation ignores the uncertainty in our estimate $\bar{\mu}_t$, so instead we might want to use

$$\zeta_t(\theta^{\zeta}) = \max\{0, \big(\bar{\mu}_t + \theta^{\zeta}\bar{\sigma}_t - \mu^{vac}\big)\}.$$

This policy is saying that μ_t *might* be as large as $\bar{\mu}_t + \theta^{\zeta}\bar{\sigma}_t$, where θ^{ζ} is a tunable parameter. Now our policy for x^{vac} would be be

$$X^{pfa-vac}(S_t|\theta^{vac}, \theta^{\zeta}) = \frac{1}{\theta^{vac}}\zeta_t(\theta^{\zeta}). \tag{20.20}$$

Using our policy $X^{obs}(S_t)$, we can write our policy for $x_t = (x_t^{vac}, x_t^{objs})$ as

$$X^{PFA}(S_t|\theta) = \big(X^{pfa-vac}(S_t|\theta^{vac}, \theta^{\zeta}), X^{pfa-obs}(S_t|\theta^{obs})\big),$$

where $\theta = (\theta^{vac}, \theta^{obs}, \theta^{\zeta})$. This policy would have to be tuned in the objective function (20.16). This policy could then be compared to that obtained by approximating Bellman's equation.

An alternative approach for designing a policy function approximation is to assume that it is represented by a linear model

$$X^{PFA}(S_t|\theta) = \sum_{f\in\mathcal{F}} \theta_f \phi_f(S_t).$$

Parametric functions are easy to estimate, but they require that we have some intuition into the structure of the policy. An alternative is to use a neural network, where θ is the weights on the links in the graph of the neural network. It is important to keep in mind that neural networks tend to be very high dimensional (θ may have thousands, even hundreds of thousands, of dimensions), and they may not replicate obvious properties. Either way, we would tune θ using the objective function in (20.16).

Cost function approximations - We are going to illustrate CFAs using the spatially distributed flu vaccination problem, where we assume we are allowed to observe just one region $x \in \mathcal{I}$ at a time (we have just one inspection team). Assume that we can only treat one region at a time, where we are always going to treat the region that has the highest estimated prevalence of the flu.

We do not know μ_{tx}, but at time t assume that we have an estimate $\bar{\mu}_{tx}$ for the prevalence of the flu in region $x \in \mathcal{I}$, where we assume that $\mu_x \sim N(\bar{\mu}_{tx}, \bar{\sigma}^2_{tx})$. We use this belief to decide which region to vaccinate, which we describe using the policy

$$X^{vac}(S_t) = \arg\max_{x\in\mathcal{I}} \bar{\mu}_{tx}.$$

We then have to decide which region to observe. We can approach this problem as a multiarmed bandit problem, where we have to decide which region ("arm" in bandit lingo) to observe. The most popular class of policies for learning problems in the computer science community is known as upper confidence bounding for multiarmed bandit problems. A class of UCB policy is interval estimation which would choose the region x that solves

$$X^{obs-IE}(S_t|\theta^{IE}) = \arg\max_{x\in\mathcal{I}} \left(\bar{\mu}_{tx} + \theta^{IE}\bar{\sigma}_{tx}\right) \tag{20.21}$$

where $\bar{\sigma}_{tx}$ is the standard deviation of the estimate $\bar{\mu}_{tx}$.

The policy $X^{obs-IE}(S_t|\theta^{IE})$ is a form of parametric cost function approximation; it requires solving an imbedded optimization problem, and there is no explicit effort to approximate the impact of a decision now on the future. It is easy to compute, but θ^{IE} has to be tuned. To do this, we need an objective function. Note that we are going to tune the policy in a simulator, which means we have access to μ_{tx} for all $x \in \mathcal{I}$.

Let $x^{obs}_t = X^{obs-IE}(S_t|\theta^{IE})$ be the region we choose to observe given what we know in S_t. This gives us the observation

$$W_{t+1,x^{obs}_t} = \mu_{t,x^{obs}_t} + \varepsilon_{t+1},$$

where $\varepsilon \sim N(0, \sigma^2_W)$. We would then use this observation to update the estimates $\bar{\mu}_{t,x^{obs}_t}$ using the Bayesian updating equations for the mean (7.26) and precision (7.27) (see equations (20.11) and (20.12) above).

It is important to remember that the true prevalence μ_{tx} is changing over time as a result of our policy of observation and vaccination, so we are going to refer to it as $\bar{\mu}^{\pi}_{tx}(\theta^{IE})$, where the observation policy is parameterized by θ^{IE}.

We are learning in the field, which means we want to minimize the prevalence of the flu over all regions, over time. Since we are using a simulator to evaluate policies, we would evaluate our performance using the true level of flu prevalence, given by

$$F^{\pi}(\theta^{IE}) = \mathbb{E}_{S_0}\left\{\sum_{t=0}^{T}\sum_{x\in\mathcal{I}}\bar{\mu}^{\pi}_{tx}(\theta^{IE})|S^0\right\}. \tag{20.22}$$

We then need to tune our policy by solving

$$\min_{\theta^{IE}} F^{\pi}(\theta^{IE}).$$

Policies based on value functions - Any sequential decision problem with a properly defined state variable can be solved using Bellman's equation:

$$V_t(S_t) = \max_x \big(C(S_t,x_t) + \mathbb{E}\{V_{t+1}(S_{t+1})|S_t,x_t\}\big),$$

which gives us the policy

$$X^{VFA}(S_t) = \arg\max_x \big(C(S_t,x_t) + \mathbb{E}\{V_{t+1}(S_{t+1})|S_t,x_t\}\big).$$

In practice we cannot compute $V_t(S_t)$, so we resort to methods that approximate the value function (see chapters 15, 16 and 17).

The use of approximate value functions has been recognized for a wide range of dynamic programming and stochastic control problems. However, it has been largely overlooked for problems with a belief state, with the notable exception of the literature on Gittins indices (see section 7.6), which reduces high-dimensional belief states (the beliefs across an entire set of arms) down to one dynamic program per arm.

In principle, approximate dynamic programming can be applied to even high-dimensional problems, including those with belief states, by replacing the value function $V_t(S_t)$ with a statistical model such as

$$V_t(S_t) \approx \overline{V}_t(S_t|\theta) = \sum_{f\in\mathcal{F}}\theta_f\phi_f(S_t),$$

where $(\phi_f(S_t))_{f\in\mathcal{F}}$ is a set of features. Alternatively, we might approximate $\overline{V}_t(S_t)$ using a neural network.

We note that we might write our policy as

$$X^{VFA}(S_t|\theta) = \arg\max_x \left(C(S_t,x) + \sum_{f\in\mathcal{F}}\theta_f\phi_f(S_t,x)\right), \tag{20.23}$$

where $(\phi_f(S_t,x_t)_{f\in\mathcal{F}}$ is a set of features involving both S_t and x_t. For example, we might design something like

$$X^{VFA}(S_t|\theta) = \arg\max_x \left(C(S_t,x) + \left(\theta_{t0} + \theta_{t1}\bar{\mu}_t + \theta_{t2}\bar{\mu}_t^2 + \theta_{t3}\bar{\sigma}_t + \theta_{t4}\beta_t\bar{\sigma}_t\right)\right).$$

There are a variety of strategies for fitting θ that have been developed, as we reviewed in chapters 3 and 16.

Direct lookahead policy - Direct lookahead policies involve solving an approximate lookahead model that we previously gave in equation (11.24), but repeat it here for convenience:

$$X_t^{DLA}(S_t) = \arg\max_{x_t}\left(C(S_t,x_t) + \tilde{E}\left\{\max_{\tilde{\pi}}\tilde{E}\left\{\sum_{t'=t+1}^{T} C(\tilde{S}_{tt'},\tilde{X}^{\tilde{\pi}}(\tilde{S}_{tt'}))|\tilde{S}_{t,t+1}\right\}|S_t,x_t\right\}\right). \tag{20.24}$$

The problem with direct lookahead policies is that it requires solving a stochastic optimization problem (to solve the original stochastic optimization problem). To make it tractable, we can introduce various approximations. Some that are relevant for our problem setting could be:

1) Use a deterministic approximation. These are effective for pure resource allocation problems (google maps using a deterministic lookahead to find the best path to the destination over a stochastic graph), but seem unlikely to work well for learning problems.

2) Use a parameterized policy for $\tilde{\pi}$. We could use any of the policies suggested above as our lookahead policy. We would then also have to use Monte Carlo sampling to approximate the expectations.

3) We can solve a simplified Markov decision process.

4) We could approximate the lookahead using Monte Carlo tree search.

We are going to illustrate the third approach. We start with model 3 of the flu problem, which requires the state variable

$$S_t^{cont} = \big((\bar{\mu}_t,\beta_t),(\bar{\delta}_t,\beta_t^{\delta})\big).$$

We might be able to do a reasonable job of solving a dynamic program with a two-dimensional state variable (using discretization), but not a four-dimensional state. One approximation strategy is to fix the belief about the drift δ by holding $(\bar{\delta}_t,\beta_t^{\delta})$ constant. This means that we continue to model the true δ with uncertainty, but we ignore the fact that we can continue to learn and update the belief. This means the state variable $\tilde{S}_{tt'}$ in the lookahead model is given by

$$\tilde{S}_{tt'} = (\tilde{\bar{\mu}}_{tt'},\tilde{\beta}_{tt'}).$$

Assuming that we can discretize the two-dimensional state, we can solve the lookahead model using classical backward dynamic programming on this approximate model (we could do this in steady state, or over a finite horizon, which makes more sense). Solving this model will give us exact value functions $\widetilde{V}_{tt'}(\tilde{S}_{tt'})$ for our approximate lookahead model, from which we can then find the decision to make now given by

$$X_t^{\pi}(S_t) = \arg\max_{x}\Big(C(S_t,x) + \mathbb{E}\{\widetilde{V}_{t,t+1}(\tilde{S}_{t,t+1})|S_t\}\Big). \tag{20.25}$$

Then, we implement $x_t = X_t^{\pi}(S_t)$, step forward to $t+1$, observe W_{t+1}, update to state S_{t+1} and repeat the process.

A hybrid policy - We have two types of decisions: whether to observe x_t^{obs}, and how many to vaccinate x_t^{vac}. We can combine them into a single, two-dimensional decision $x_t = (x_t^{obs}, x_t^{vac})$ and then think of enumerating all possible actions. However, we can also use hybrids. For example, we could use the policy function in equation (20.19), but then turn to any of the other four classes of policies for x_t^{vac}. This not only reduces the dimensionality of the problem, but might help if we feel that we have confidence in the function for x_t^{obs} (perhaps a UCB-style policy?) but are less confident designing a function for the more complex x_t^{vac} since it is managing physical resources.

20.3 THE POMDP PERSPECTIVE*

The POMDP community approaches the controllable version of our flu problem by viewing it as a dynamic program with state μ_t and action x_t that controls (or at least influences) this state. Viewed from this perspective, μ_t is *the* state of the system. Any reference to "the state" refers to the environment state μ_t in our two-agent model above. In our resource constrained system, we would add R_t to the state variable giving us $S_t = (R_t, \mu_t)$, but for now we are going to focus on the unconstrained problem.

The community then shifts to the idea of modeling the belief about μ_t, and then introduces the "belief MDP" where the belief is the state (instead of μ_t).

The problem with these two versions of a Markov decision process is that there is not a clear model of who knows what. There is also the confusion of a "state" s (sometimes called the physical state), and our belief $b(s)$ giving the probability that we are in state s, where $b(s)$ is its own state variable!! This issue arises not just in who has access to the value of μ_t, but also information about the transition function. We resolved this confusion above with our two-agent model.

To help us present the POMDP perspective, we are going to make the following assumptions assumptions:

A1) We are solving the problem in steady state.

A2) Our state space is discrete, which means we can write $S_t \in \mathcal{S} = \{s_1, \ldots, s_K\}$. For example, our state may be the blood sugar of a patient (discretized). We are not able to observe the state perfectly (our observation of blood sugar comes with sampling error and the natural variations of blood sugar in the patient).

A3) We act on the (unknown) state with a decision x_t, which might be the choice of diet to control blood sugar (or a choice of medication, including the type of drug and its dosage).

A4) We can compute the one-step transition matrix $p(s'|s,x)$ which is the probability that we transition to $S = s'$ given that we are in state s and take action x. It is important to remember that $p(s'|s,x)$ is computed using

$$p(s'|s,x) \quad = \quad \mathbb{E}_S \mathbb{E}_{W|S} \{ \mathbb{1}_{\{S_{t+1}=s'=S^M(s,x,W)\}} | S_t = s \}.$$

The first expectation $\mathbb{E}_S$ captures our uncertainty about the state S (such as the actual blood sugar), while the second expectation $\mathbb{E}_{W|S}$ captures the noise in the observation of S. Computing the one-step transition matrix $p(s'|s,x)$ means we need to know the transition function $S^M(s,x,W)$, which may not be known (we may not know how a patient responds to a diet or drug). In addition we need to know the probability distribution for W.

Physical (unobservable) system	
$S_t = s$	Physical (unobservable) state s
x_t	Decision (made by the controller) that acts on S_t
W_{t+1}	Exogenous information impacting the physical state S_t
$S^M(s, x, w)$	Transition function for physical state $S_t = s$ given $x_t = x$ and $W_{t+1} = w$
$p(s'\|s, x)$	$Prob[S_{t+1} = s'\|S_t = s, x_t = x]$
Controller system	
$b(s)$	Probability (belief) we are in physical (unobservable) state s
W^{obs}	Noisy observation of physical state $S_t = s$
$\mathcal{W}^{obs}$	Space of outcomes of W^{obs}
$P^{obs}(w\|s)$	Probability of observing $W^{obs} = w$ given $S_t = s$

Table 20.2 Table of notation for POMDPs

Table 20.2 presents the notation we use in our model.

With these assumptions, we can formulate the familiar form of Bellman's equations for discrete states and actions

$$V(s) = \max_x \left(C(s, x) + \sum_{s' \in \mathcal{S}} p(s'|s, x)V(s') \right). \tag{20.26}$$

The problem with solving Bellman's equation (20.26) to determine actions is that the controller determining x is not able to see the state s. The POMDP community addresses this by creating a belief $b(s)$ for each state $s \in \mathcal{S}$. At any point in time, we can only be in one state, which means

$$\sum_{s \in \mathcal{S}} b(s) = 1.$$

The POMDP literature then creates what is known as the *belief MDP* in terms of the belief state vector $b = (b(s_1), \ldots, b(s_K)) = (b_1, \ldots, b_K)$ where b_k is the probability (our belief) that $\mu = s_k$. This is a dynamic program whose state is given by the continuous vector b. We next introduce the transition function for the belief vector b given by

$B^M(b, x, W)$ = the transition function that gives the probability vector $b' = B^M(b, x, W)$ when the current belief vector (the prior) is b, we make decision x and then observe the random variable W, which is a noisy observation of μ_t if we have chosen to make an observation.

The function $B^M(b, x, W)$ returns a vector b' that has an element $b'(s)$ for each physical state s. To be clear, if $b_t(s)$ is our belief that we are in state s, then b_{t+1} is the vector of beliefs that we are in each state s' which is given by

$$b_{t+1} = B^M(b_t, x_t, W_{t+1}).$$

We will write $B^M(b, x, W)(s)$ to refer to element s of the vector returned by $B^M(b, x, W)$.

The derivation of the belief transition function is a moderately difficult exercise in the use of Bayes theorem, which we defer until section 20.8.1. However, we can show that Bellman's equation (20.26) can be written

$$V(b_t) = \max_x \left(C(b_t, x) + \sum_{s \in \mathcal{S}} b_t(s) \sum_{s' \in \mathcal{S}} p(s'|s, x) \sum_{w^{obs} \in \mathcal{W}^{obs}} P^{obs}(w^{obs}|s', x) V(B^M(b_t, x, w^{obs})) \right). \tag{20.27}$$

If equation (20.27) can be solved, then the policy for making decisions for the controller is given by

$$X^*(b_t) = \arg\max_x \left(C(b_t, x) + \sum_{s \in \mathcal{S}} b_t(s) \sum_{s' \in \mathcal{S}} p(s'|s, x) \sum_{w^{obs} \in \mathcal{W}^{obs}} P^{obs}(w^{obs}|s', x) V(B^M(b_t, x, W^{obs})) \right).$$

As if the computations behind these equations were not daunting enough, we need to also realize that we are combining the decisions of the controller with a knowledge of the dynamics (captured by the transition matrix) of the physical system. The one-step transition matrix requires knowledge of both the transition function $S^M(s, x, W)$, which will not always be known to the controller. We are going to illustrate a setting where the transition is not known to the controller below.

A challenge here is that even through we have discretized the unobservable state, the vector b is continuous. However, Bellman's equation using the state b has some nice properties that the research community has exploited. Just the same, it is still limited to problems where the state space $\mathcal{S}$ of the unobservable system is relatively small. Keep in mind that small problems can easily produce state spaces of 10,000, and we could never execute these equations with a state space that large (think of the size of our state space when we have a belief for each of 50 states in the U.S., or 3,000 counties). For this reason, the POMDP community has developed a variety of approximation strategies.

By contrast, our two-agent model avoids the assumption that the controller knows the transition function, and opens the door to using any of the four classes of policies which, as we show above, can be designed to scale to very high dimensional problems. Just as important, it opens the door to simple PFAs and CFAs that are likely to be much easier to explain and implement.

20.4 THE TWO-AGENT NEWSVENDOR PROBLEM

In section 2.3.1 we introduced the newsvendor problem, which is a basic problem that arises in many applications involving the allocation of resources under under uncertainty. In this problem, we allocate a quantity x_t, then observe a demand $\hat{R}_{t+1}$. We are going to use the cost minimizing version, where there is an underage cost c^u for each unit of unsatisfied demand, and an overage cost c^o for each unit of excess inventory (that is still not held until the next time period). This produces a cost function $F(x, D)$ given by

$$F(x, D) = c^u \max\{0, D - x\} + c^o \max\{x - D\}. \tag{20.28}$$

There are many applications where a "field agent" (that we designate "q") perceives a need to provide resources to meet an estimated demand, but has to ask for the resources from a "central agent" (q') who may not fill the entire request. The field agent typically has a higher cost of running out than having too much (that is, $c_q^u > c_q^o$). Imagine running out of food, blood, or ammunition, compared to having excess quantities of those three resources.

Let

$$\begin{aligned}\hat{R}_{t+1} &= \text{actual demand for resources at time } t+1,\\ \hat{R}_t^e &= \text{estimate of the demand } \hat{R}_{t+1} \text{ made at time } t,\\ &= \mathbb{E}\hat{R}_{t+1}.\end{aligned}$$

In other words, $\hat{R}_t^e$ is an unbiased estimate of $\hat{R}_{t+1}$, information that we were not provided in our original newsvendor problem.

We can find an optimal solution to the problem in equation (20.28) if we knew the distribution of $\hat{R}_{t+1}$ given the estimate R_t^e, but let's assume we do not know the distribution. However, we can still say that the optimal solution would be given by

$$X^\pi(S_t|\theta_t) = R_t^e + \theta_t, \tag{20.29}$$

where S_t is what is known by the decision-maker at time t. We might start by assuming that $S_t = R_t^e$, but the state variable will also depend on the process for adaptively updating θ_t.

The real challenge that our field agent faces is that he has to ask for the resources from a "central agent," who has her own objective. This is a twist (that arises often in practice) that we did not face in our original newsvendor problem. Let

$$\begin{aligned}x_{tqq'} &= \text{the request made by field agent } q \text{ to the central agent } q' \text{ for resources},\\ &= X_q^\pi(S_{tq}|\theta_{tq}),\\ x_{tq'q} &= \text{the amount that the central agent } q' \text{ decides to give to the field agent } q,\\ &= X_{q'}^\pi(S_{tq'}|\theta_{tq'}).\end{aligned}$$

This produces a cost function $F_q(x, \hat{R})$ for our field agent q given by

$$F_q(x_{tqq'}, \hat{R}_{t+1}) = c_q^u \max\{0, \hat{R}_{t+1} - x_{tq'q}\} + c_q^o \max\{0, x_{tq'q} - \hat{R}_{t+1}\}, \tag{20.30}$$

where the amount that our central agent provides to the field agent, $x_{tq'q}$, depends on the original request $x_{tqq'}$ made by the field agent to the central agent, although the relationship is something that the field agent has to try to estimate.

To understand how the central agent might make her decision $x_{tq'q}$, we have to formulate her objective function, which is given by

$$F_{q'}(x_{tq'q}, \hat{R}_{t+1}) = c_{q'}^u \max\{0, \hat{R}_{t+1} - x_{tq'q}\} + c_{q'}^o \max\{0, x_{tq'q} - \hat{R}_{t+1}\}. \tag{20.31}$$

The only difference between the objective for agent q and agent q' is that q uses the costs (c_q^u, c_q^o) while agent q' uses $(c_{q'}^u, c_{q'}^o)$. Typically, $c_q^u > c_{q'}^u$ and $c_q^o < c_{q'}^o$, reflecting the likelihood that field agents have a much stronger desire not to run out of resources.

Note that the performance of the central agent still depends on satisfying the unknown demand $\hat{R}_{t+1}$, but it might be reasonable to assume that the underage and overage costs for

the central agent might satisfy $c^u_{q'} = c^o_{q'}$, which means she really wants the field agent to match expected demand as closely as possible, while the field agent wants to guard against being under. This creates a tension (which happens frequently in practice) where the field agent wants higher resource levels than the central agent, who is less sensitive to underage and more sensitive to the cost of the excess inventory. This highlights how two agents, presumably working together (but with different goals) can end up behaving competitively.

Recall that the field agent is given an unbiased estimate R^e_t of $\hat{R}_{t+1}$, but since $c^u_q < c^o_q$, it is to be expected that the optimal value of θ_{tq} in equation (20.29) means that $\theta_{tq} > 0$, which means that the request $x_{tqq'}$ made by agent q to agent q', despite being based on the unbiased estimate R^e_t, is likely to be a biased estimate of $\hat{R}_{t+1}$.

We are going to borrow from the original newsvendor policy in equation (20.29) and propose a policy for agent q of the same form

$$X^\pi_q(S_{tq}|\theta_{tq}) = x_{tqq'} = R^e_t + \theta_{tq}. \tag{20.32}$$

In this case, we now have two reasons to believe that $\theta_{tq} > 0$:

- Since $c^u_q > c^o_q$, we will want to order a quantity greater than our estimate R^e_t.
- For reasons we present below, we are going to expect that our central agent is going to give the field agent *less* than he asks, which means we expect $x_{tq'q} < x_{tqq'}$. The field agent will know this, and use this as a reason to further inflate his request.

Since it is likely that $x_{tqq'}$ is biased upward, it makes sense to propose a policy for the central agent of the form

$$X^\pi_{q'}(S_{tq}|\theta_{tq}) = x_{tqq'} - \theta_{tq'}. \tag{20.33}$$

We have subtracted the correction term $\theta_{tq'}$ so that we can expect $\theta_{tq'} > 0$.

The policy in (20.33) uses only the request from agent q to guide the decision of the central agent. This makes the central agent relatively easy to manipulate. For example, while the central agent will learn that the request $x_{tqq'}$ is inflated and make adjustments (as shown in (20.33)), the field agent could keep inflating $x_{tqq'}$.

A more plausible model would be to assume that the central agent balances the request $x_{tqq'}$ from the field agent against an independent source of knowledge. One idea would be to estimate the bias and variance of the request from the field agent, and the bias and variance of an independent source of knowledge, using the methods described in section 3.5. Let $w_{tqq'}$ and $w_{t\cdot q'}$ be the weights given to each source of information, which are computed by taking the inverse of the variance of each estimate plus the square of the bias, normalized so the two weights sum to one (see section 3.6.3). We then obtain a blended estimate of the need using

$$x^{blend}_{tq'} = w_{tqq'}x_{tqq'} + w_{t\cdot q'}x_{t\cdot q'}. \tag{20.34}$$

This mechanism also encourages a level of truthfulness from the field agent, since significant bias or noise would have the effect of reducing $w_{tqq'}$.

If we adopt the policies (20.32) and (20.33) for agents q and q', we then need to design policies for adjusting θ_{tq} and $\theta_{tq'}$ which we represent using

$$\theta_{t+1,q} = \Theta^\pi_q(S_{tq}),$$
$$\theta_{t+1,q'} = \Theta^\pi_{q'}(S_{tq'}).$$

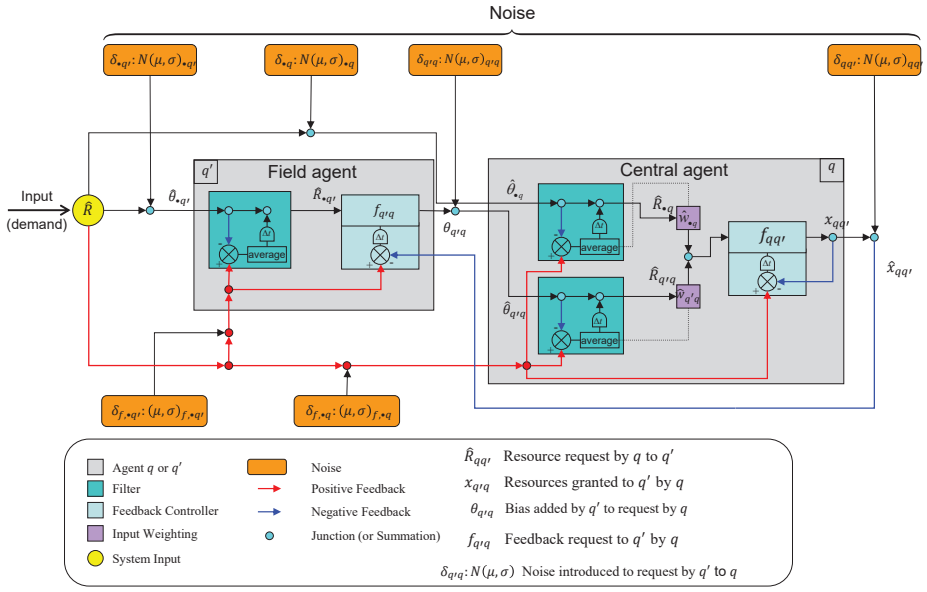


Figure 20.3 Information flow diagram for a two-agent newsvendor problem.

Now we face the challenge of designing the adaptive learning policies $\Theta^\pi_q(S_{tq})$ and $\Theta^\pi_{q'}(S_{tq'})$.

Figure 20.3 depicts the flow of information in a graphical format, similar to that used by engineers to draw circuits. This figure shows where noise might enter the different forms of communication. It shows the field agent making decisions based on the input from the field, and then sending a request to the central agent. It then shows the central agent combining the request from the field agent with its own information about the environment to make its own recommendation.

We need to recognize that there is not a "right" way to design these policies, since this has to do with modeling human behavior. For example, it is possible that the central agent (who has to manage hundreds of field agents) uses a simple rule

$$\Theta^\pi_{q'}(S_{tq'}|\rho_{q'}) = (1-\rho_{q'})\theta_{tq'} + \rho_{q'}(x_{tqq'} - \hat{R}_{t+1}). \tag{20.35}$$

Note that with this policy, our state variable for q' would be $S_{tq'} = (x_{tqq'}, \theta_{tq'})$.

The policy in (20.35) is a simple reactive policy (a form of PFA) with a smoothing parameter $\rho_{q'}$ that will need to be tuned. Agent q' does not know the initial estimate R^e_t (this information is private to the field agent), but we assume that q' is able to learn the actual demand $\hat{R}_{t+1}$ at the end of the period (note that this is not always true in practice). Agent q' would like to know how much the field agent is biasing his request, but her best estimate of R^e_t is $\hat{R}_{t+1}$.

Similarly, we might pose a policy for agent q

$$\Theta^\pi_q(S_{tq}|\rho_q) = (1-\rho_q)\theta_{tq} + \rho_q(x_{tqq'} - x_{tq'q}). \tag{20.36}$$

Again, this is a simple reactive policy that exploits the ability of the field agent to use the difference between what the field agent asked for, $x_{tqq'}$, and what the central agent provided, $x_{tq'q}$.

While there is not a "right" policy for either agent, we can pose the question of designing the best (or optimal) policy for one agent (say the field agent q), given an assumed policy $X^{\pi}_{q'}(S_{tq'})$ for the central agent. In practice, we should be considering all four classes of policies, although PFAs such as the policies (20.32) and (20.33) are going to be the most popular in a setting like this. This means that each agent faces a derivative-free stochastic search problem, so the methods of chapter 7 come into play, with one twist. Since the other agent is likely to be constantly adjusting their policy, each agent faces a nonstationary learning problem.

This very simple model has tremendous richness in terms of designing policies which nicely illustrate some of the challenges of multiagent settings. Some strategies that we can consider using include:

- If each agent uses a simple adjustment policy such as those depicted in equations (20.32) and (20.33), they each run the risk of the other agent learning their policy, and then making adjustments. A simple strategy to hide a policy is to introduce some noise. For example, the central agent might use the policy

$$X^{\pi}_{q'}(S_{tq}|\theta_{tq}) = x_{tqq'} - \theta_{tq'} + \varepsilon_{q'},$$

  where $\varepsilon_{q'} \sim N(0, \sigma^2_{q'})$ is a zero mean noise term. The challenge of the central agent is to choose the right value of the variance $\sigma^2_{q'}$; too low, and the field agent will be able to learn her policy; too high, and the central agent is responding with resource levels that are highly suboptimal for the central agent, and may simply result in the field agent further inflating his requests.

- The field agent may anticipate the policy $\Theta^{\pi}_{q'}(S_{tq'})$ for updating their adjustment, and build this response mechanism into their own choice. This means approximating $\partial\Theta^{\pi}_{q'}(S_{tq'})/\partial\theta_{tq}$.

- Each agent can turn the problem of optimizing their adjustment as an active learning problem. That is, consider discretizing θ_{tq} (from the perspective of the field agent), and then turning this into an active learning problem, which can be approached using the methods of chapter 7.

- A common strategy is to guess at the other agent's state variable, which means understanding what information they are using to make their decisions, and then trying to manipulate it (without their knowledge, of course).

One theme that should emerge from all of these policies is that learning the behavior (that is, the policy) of other agents requires looking at their decisions.

20.5 MULTIPLE INDEPENDENT AGENTS - AN HVAC CONTROLLER MODEL

Consider the problem of controlling the air conditioning in each apartment of a large building with two goals in mind:

- The temperature of each apartment needs to stay within the temperature range E^{min} and E^{max}, with specific financial penalties (in the form of rent discounts) when the temperature falls outside of this range.

- The building pays real-time grid prices that change every 5 minutes, and the building operator would like to minimize what it pays for electricity.

The air conditioning is controlled by a thermostat in each apartment, which we are going to model as an agent. The thermostats do not communicate.

20.5.1 Model

We model the system for each thermostat q using our standard framework as follows:

State variables:

$$\begin{aligned}
E_{tq}^{in} &= \text{Current temperature of apartment } q.\\
E_t^{out} &= \text{Current outdoor temperature.}\\
H_{tq} &= \begin{cases} 1 & \text{If the air conditioner in apartment } q \text{ is currently on.}\\ 0 & \text{If the air conditioner is currently off.}\end{cases}\\
S_{tq} &= \text{State of the air conditioning system in apartment } q\\
&= (E_{tq}^{in}, E_t^{out}, H_{tq}).
\end{aligned}$$

Decision variables:

$$x_{tq} = \begin{cases} +1 & \text{If we wish to turn the AC on (requires } H_{tq}=0).\\ 0 & \text{If we wish to leave it unchanged.}\\ -1 & \text{If we wish to turn the AC off (requires } H_{tq}=1).\end{cases}$$

We let $X_q^\pi(S_{tq})$ be the policy that determines x_{tq} given the information in S_{tq}.

Exogenous information variables:
There are two ways to model our system:

- Model-free - We use this approach if we do not wish to assume that we know anything about the dynamics about how the temperature E_{tq}^{in} evolves over time. In this case, we assume only that we observe $E_{t+1,q}^{in}$, which means that this is the exogenous information, so $W_{t+1,q} = E_{t+1,q}^{in}$. We know $H_{t+1,q}$ since we know H_{tq} and x_{tq} which then determines $H_{t+1,q}$, which means that we could write $W_{t+1,q} = S_{t+1,q}$ (that is, our exogenous information is the state variable).

- Model-based - This opens the door to a much richer model. Assume that we have access to the external temperature (call this E_t^{out}), and that we have access to a dynamic model based on the thermodynamic properties of the apartment. In this case, our exogenous information would be $W_{t+1,q} = E_{t+1}^{out}$ (assuming the thermostat has access to the outside temperature).

Transition function:
If we use our model-free formulation, then $S_{t+1,q} = W_{t+1,q}$, which is to say that we simply observe the state at time $t+1$ rather than calculating how we reach this state given x_{tq} and observing $EW_{t+1,q}$.

A model-based representation might allow us to observe the external temperature E_{t+1}^{out} and then use a dynamic equation such as

$$E_{t+1,q}^{in} = E_{tq}^{in} + \rho_q\big(E_{t+1}^{out} - E^0\big) + \varepsilon_{t+1,q}, \tag{20.37}$$

where E^0 is a base temperature (around 65 degrees) and ρ_q is a coefficient that reflects the thermal transfer for apartment q.

Objective function:
A natural way to evaluate our performance is to measure deviations outside of our range $(\theta^{min}, \theta^{max})$, as in

$$C_q(S_{tq}, x_{tq}) \quad = \quad c^u \max\{\theta^{min} - E_t, 0\} + c^o \max\{E_t - \theta^{max}, 0\}.$$

The coefficients c^u and c^o would have to be chosen to reflect the discomfort of a building that is too cold (in the summer time) versus too warm. We could, if we wish, also include operating cost, giving us

$$C(S_t, x_t) \quad = \quad c^u \max\{\theta^{min} - E_t, 0\} + c^o \max\{E_t - \theta^{max}, 0\} + c^{oper} H_t.$$

Since c^{oper} reflects an actual operating cost, c^u and c^o would then have to be scaled to capture the relative importance of operating cost, versus the discomfort of being cold or hot. This is a nice example of a multiobjective cost function.

20.5.2 Designing policies

Stationary controllers are, of course, the simplest, but even with our basic temperature-controlling HVAC controller, it may still make sense to consider all four classes of policies:

Policy function approximation - These are easily the most widely used policies in practice. A natural policy (for air conditioning) would be to use

$$X_q^{AC}(S_t|\theta) \quad = \quad \begin{cases} 0 & \text{if } E_t < \theta^{min}, \\ +1 & \text{if } E_t > \theta^{max}, \\ H_t & \theta^{min} \leq E_t \leq \theta^{max}. \end{cases}$$

We intentionally left θ constant across all the apartments. The building manager, who is not strictly an agent (since she plays no role other than setting θ), might set this policy by solving

$$\min_{\theta} \sum_{t=0}^{T} \sum_{q \in \mathcal{Q}} C_q(S_{tq}, X_{tq}^{PFA}(S_{tq}|\theta)). \tag{20.38}$$

There are several ways that this simple policy can be generalized:

- θ can depend on each agent to capture different thermal transfer rates among apartments.
- θ can be made time-dependent to capture time-of-day effects.

A major limitation of PFAs can arise if we can anticipate changes in the temperature (as might happen every morning in winter) that produce changes in our internal temperature E_{tq}^{in} that are larger than we can compensate for using our HVAC system. It may be necessary to begin pre-heating (in winter) or pre-cooling (in summer) to compensate for peak periods.

Cost function approximation - CFAs open the door to using deterministic rolling horizon procedures, as we did in our energy example in section 13.3.3. This just leaves the

need for performing tuning. We can do this offline as we did in the example in section 13.3.3, but a nice challenge would be to do this online, in the field, which means we would be optimizing cumulative performance. We can apply the techniques of derivative-free stochastic search in chapter 7.

Value function approximation - It is natural in this type of application, with clear time-of-day patterns, to expect that we would need a policy that depends on time of day. We hinted at this above when we suggested a time-dependent control parameter θ in our PFA, but this turned a two-dimensional search problem into what might be a much higher dimensional search (especially if we make θ depend on 5-minute increments). VFA-based policies, on the other hand, tend to be naturally time-dependent. We can use any of the approximate dynamic programming algorithms described in either chapter 15 (backward approximate dynamic programming) or chapters 16-17 (forward approximate dynamic programming).

Direct lookahead - Time-dependent applications are natural candidates for lookahead policies, as we saw with the energy storage problem in section 13.3.3, which is actually a combination DLA-CFA, since the lookahead was parameterized. Imagine that we are given a forecast of daily temperatures that allows us to optimize the process of tuning the air conditioning on and off to maintain proper temperatures. This would involve solving an integer program over a rolling horizon. While this is not a particularly difficult problem using modern integer programming solvers, it is introducing substantially greater computations compared to the other policies. If this is done using a central server, this approach could be possible, but if the calculations have to be performed on each thermostat in each apartment, it would be out of the question.

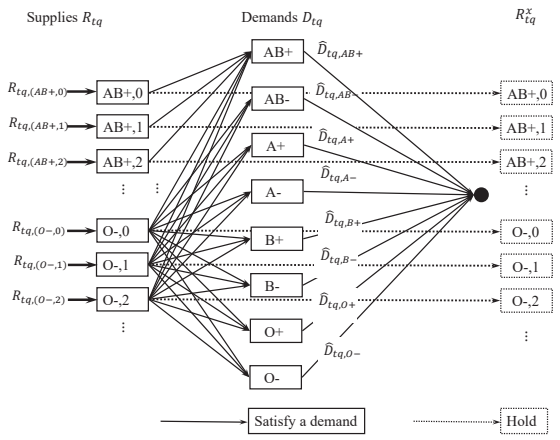
20.6 COOPERATIVE AGENTS - A SPATIALLY DISTRIBUTED BLOOD MANAGEMENT PROBLEM

In section 8.3.2 we introduced the problem of managing inventories of blood, where we captured the eight different types of blood, the age (0 to 5 weeks), and modeled the ability to substitute different types of blood. However, the model did not capture location. Now assume that each hospital is managing its own inventories of blood, but has the ability to send blood from one location to another. Further assume (for the moment) that hospitals are altruistic - a life is a life and each hospital wants to make sure its blood inventories are best used, whether it is at a hospital or another hospital.

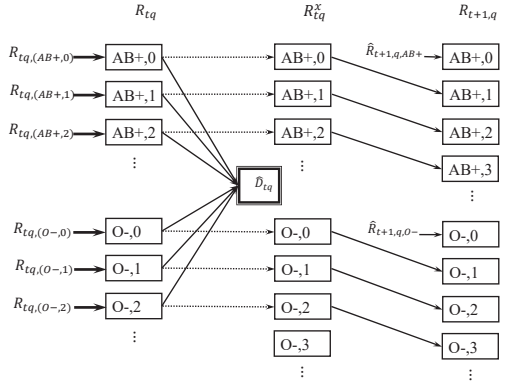
We are going to use the strategy of separable, piecewise linear value function approximations that we introduced in section 18.5. We could use Benders cuts (described in section 18.6) but the use of separable piecewise linear approximations can be accomplished with much simpler communication between agents.

Since the mathematical model has already been described (in section 8.3.2), as has the logic for doing separable piecewise linear approximations (in section 18.5). For this reason, we are going to review the model and approximation methodology graphically. Figure 20.4(a) shows the network model that we would use for a single hospital q, where we have a node for each blood type and age, with arcs to demands for each blood type (if substitution is allowed).

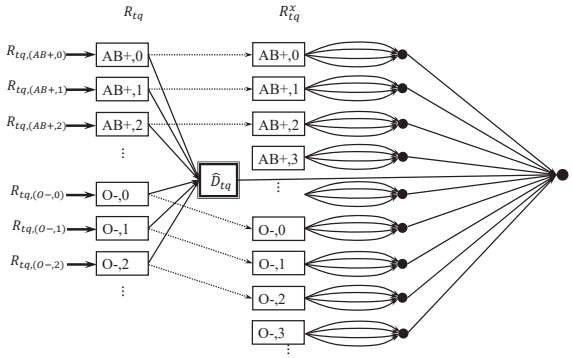
Figure 20.4(b) cleans up the figure by sweeping all the demands into a single node, making it easier to see the blood that is held, first flowing into the node for the post-



(a)



(b)



(c)

Figure 20.4 (a) Blood network showing assignments of blood supplies to demands of different blood types. (b) Blood network with demand arcs aggregated, and showing holding to post-decision R^x_t and the following pre-decision R_{t+1}. (c) Blood network showing piecewise linear value functions attached to post-decision resource nodes.

decision state (the leftover blood before new demands become known), and then into the pre-decision state of the next week (at which point the blood "ages" by one week). The policy for a single agent using separable, piecewise linear value functions would be written

$$X^{VFA-PWL}(S_t) = \arg\max_{x_t \in \mathcal{X}_t} \left(C(S_t, x_t) + \sum_{a \in \mathcal{A}} \overline{V}^x_{ta}(R^x_{ta}(x_t)) \right), \tag{20.39}$$

where $R^x_{ta}(x_t)$ is the number of units of blood with attribute a at the end of week t (that is, the post-decision resource state), where $\overline{V}^x_{ta}(R^x_{ta}(x_t))$ is the piecewise linear value function for $R^x_{ta}(x_t)$ units of blood.

Finally, figure 20.4(c) illustrates piecewise linear value functions being attached to each post-decision node, producing a deterministic linear program that is easy to solve, assuming that each hospital has access to this type of computing resource (it is easier to make this assumption for a hospital than a thermostat!).

These value function approximations are only for the value of blood held at the same hospital for the following week. We compute these piecewise linear approximations by taking the marginal value of an additional unit of blood of each type and age, and use this with algorithms such as the CAVE or Leveling algorithms (reviewed in section 18.3) to create concave (if maximizing) approximations of the value of additional units of blood (for each type and age).

Now imagine that we are applying this logic for hospital q. The piecewise linear value function approximations require only that we obtain the marginal value $\hat{v}_{tqa}$ for an additional unit of blood with attribute a (this would include blood type and age) for blood stored at hospital q. This is used to update the piecewise linear value functions for the value function approximations $\overline{V}_{t-1,qa}(R_{t-1,qa})$ as we step back in time.

In addition to making blood choice decisions at hospital q, we can also decide to move blood to another hospital q' (or from another hospital q' to hospital q). To help this process, we not only need the value of blood stored at hospital q, but also blood stored at other hospitals q'. This means that hospital q, for example, needs estimates $\overline{V}^x_{tq'a}(R^x_{tq'a})$ for each hospital q' to help decide if blood should be moved to q'.

We do not have to solve this using a global optimization over all hospitals (there are thousands in the U.S.). Instead, we can use a multiagent formulation and solve a single small model for each hospital, communicating marginal values between hospitals.

When we allow interhospital transfers, we produce the network depicted in figure 20.5 where hospital q is now allowed to move each unit of blood (identified by type and age) to other hospitals, that are represented only as the piecewise linear value functions. Mathematically, we would write the policy for hospital q as

$$X^{VFA-PWL}_q(S_t) = \arg\max_{x_{tq} \in \mathcal{X}_{tq}} \left(C(S_{tq}, x_{tq}) + \sum_{q' \in \mathcal{Q}} \sum_{a \in \mathcal{A}} \overline{V}^x_{tq'a}(R^x_{tq'a}(x_t)) \right). \tag{20.40}$$

Comparing the single agent policy in equation (20.39) to the multiagent policy in (20.40), we see they are basically the same, aside from making the decisions for each hospital separately at time t.

We have to emphasize that using this approach requires that the hospitals be willing to share their marginal values $\hat{v}_{tqa}$ for each unit of blood with each other. This approach could easily be scaled over thousands of hospitals if necessary.

This logic, carefully implemented, can produce a near-optimal solution if we are trying to optimize across all the hospitals. But what if we do not believe that our hospitals are

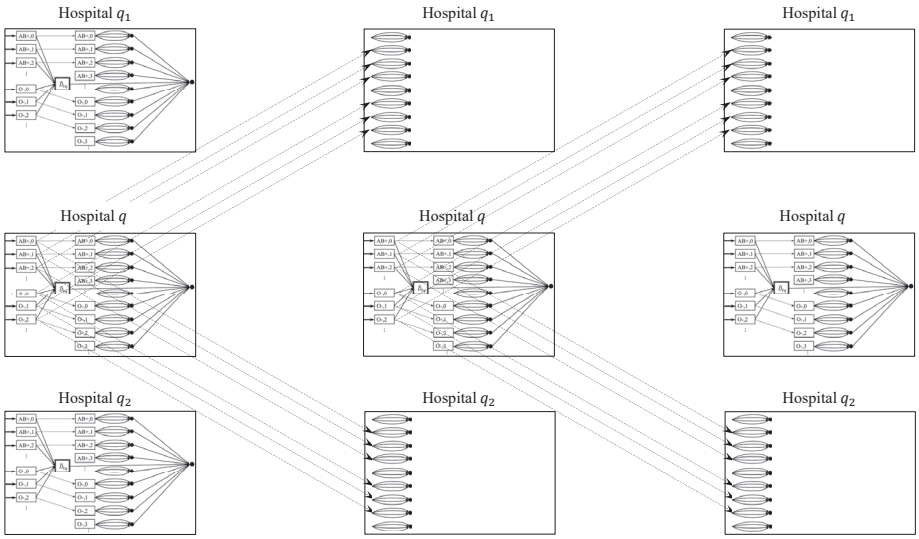


Figure 20.5 Blood management for multiple hospitals as a multiagent system

completely altruistic? We can achieve greedy behaviors by simply discounting the value of blood at hospitals other than our own. If the discount factor is set all the way down to zero, then we are back to completely greedy behaviors.

20.7 CLOSING NOTES

There is an extensive literature on multiagent systems. Much of this is centered on devices (drones, robots, unmanned aerial vehicles) where belief formation about the environment is the central challenge. These issues are largely outside of the scope of this book.

There is an equally substantial literature that focuses on specific contextual domains, where the behavior of the agents (that is, the policies) is designed for the specific problem domain such as modeling teams of robotic soccer players, drones investigating an urban environment, flocks of birds and swarms of insects. Often these models combine the design of policies with the detailed control of the physics of the device (accelerating/decelarating, turning, taking off and landing).

Our brief presentation here is designed to provide some basic notation and sketch some of the issues that arise in multiagent control. The goal here was primarily to demonstrate how to extend our modeling framework, to highlight the potential for using all four classes of policies.

20.8 WHY DOES IT WORK?

20.8.1 Derivation of the POMDP belief transition function

This section derives Bellman's equation given by equation (20.27).

The belief transition function is an exercise in Bayes' theorem. Let $b_t(s)$ be the probability we are in state $S_t = s'$ (this is our prior) at time t. We assume we have access to the

distribution

$$P^{obs}(w^{obs}|s) \quad = \quad \text{the probability that we observe } W^{obs} = w^{obs} \text{ if we are in state } s.$$

Assume we are in state $S_t = s$ and take action x_t and observe $W^{obs}_{t+1} = w^{obs}$. The updated belief distribution $b_{t+1}(s')$ would then be

$$\begin{aligned}
b_{t+1} \quad & (s'|b_t, x_t, W^{obs}_{t+1} = w^{obs}) = B^M(b_t, x, W^{obs}_{t+1} = w^{obs})(s') \\
= \quad & Prob[S_{t+1} = s'|b_t, x_t, W^{obs}_{t+1} = w^{obs}] \\
= \quad & \frac{Prob[W^{obs}_{t+1} = w^{obs}|b_t, x_t = x, S_{t+1} = s']Prob[S_{t+1} = s'|b_t, x_t]}{Prob[W^{obs} = w^{obs}|b_t, x_t]} && (20.41) \\
= \quad & \frac{Prob[W^{obs}_{t+1} = w^{obs}|b_t, x_t, S_{t+1} = s'] \sum_{s\in\mathcal{S}} Prob[S_{t+1} = s'|S_t = s, b_t, x_t]Prob[S_t = s|b_t, x_t]}{Prob[W^{obs}_{t+1} = w^{obs}|b_t, x_t]} && (20.42) \\
= \quad & \frac{Prob[W^{obs}_{t+1} = w^{obs}|S_{t+1} = s'] \sum_{s\in\mathcal{S}} Prob[S_{t+1} = s'|S_t = s, x_t]b_t(s)}{Prob[W^{obs}_{t+1} = w^{obs}|b_t, x_t]} && (20.43) \\
= \quad & \frac{P^{obs}(w^{obs}|s') \sum_{s\in\mathcal{S}} P(s'|s, x_t)b_t(s)}{P^{obs}(w^{obs}|b_t, x_t)} && (20.44) \\
& && (20.45)
\end{aligned}$$

Equation (20.41) is a straightforward application of Bayes' theorem, where all probabilities are conditioned on the decision x_t and the prior b_t (which has the effect of capturing history). Equation (20.42) handles the transition from conditioning on the belief $b(s)$ that $S_t = s$, to the state $S_{t+1} = s'$ from which observations are made. The remaining equations reduce (20.42) by recognizing when conditioning on $b_t(s)$ does not matter, and substituting in the names of the variables for the differenet probabilities.

We compute the denominator in equation (20.41) using

$$\begin{aligned}
Prob[W^{obs}_{t+1} = w^{obs}|b_t, x_t] \quad = \quad & \sum_{s'\in\mathcal{S}} Prob[W^{obs}_{t+1} = w^{obs}|S_{t+1} = s'] \\
& \cdot \sum_{s\in\mathcal{S}} Prob[S_{t+1} = s'|S_t = s, x_t]Prob[S_t = s|b_t, x_t] && (20.46) \\
= \quad & \sum_{s'\in\mathcal{S}} P^{obs}(w^{obs}|s') \sum_{s\in\mathcal{S}} P(s'|s, x_t)b_t(s). && (20.47)
\end{aligned}$$

Equation (20.44) is fairly straightforward to compute as long as the state space $\mathcal{S}$ is not too large (actually, it has to be fairly small), the observation probability distribution $P^{obs}(w^{obs}|S = s, x)$ is known, and the one-step transition matrix $P(s'|s, x)$ is known. Knowledge of $P^{obs}(w^{obs}|S = s, x)$ requires an understanding of the structure of the process of observing the unknown system. For example, if we are sampling the population to learn about who has the flu, we might use a binomial sampling distribution to capture the probability that we sample someone with the flu. Knowledge of the one-step transition matrix $P(s'|s, x)$, of course, requires an understanding of the underlying dynamics of the physical system.

This said, note that we have three summations over the state space to compute a single value of $b_{t+1}(s')$. This has to be repeated for each $s' \in \mathcal{S}$, and it has to be computed for each action x_t and observation W^{obs}_{t+1}. That is a lot of nested loops. The problem is that we are modeling two transitions: the evolution of the state S_t, and the evolution of the belief vector $b_t(s)$. This would not be an issue if we were just simulating the two systems. Equations (20.44) and (20.47) require computing expectations to find the transition probabilities for both the physical state S_t and the belief state b_t.

The POMDP community then approaches solving this dynamic program through Bellman's equation (for steady state problems) that can be written

$$V(b_t) = \max_x \big(C(b_t,x) + \mathbb{E}\{V(B^M(b_t,x,W^{obs}_{t+1}))|b_t,x\}\big).$$

It is better to expand the expectation operator over the actual random variables that are involved. Assume that we are in physical state S_t with belief vector b_t. The expectation would then be written

$$V(b_t) = \max_x \big(C(b_t,x) + \mathbb{E}_{S_t|b_t}\mathbb{E}_{S_{t+1}|S_t}\mathbb{E}_{W^{obs}_{t+1}|S_{t+1}}\{V(B^M(b_t,x,W^{obs}))|b_t,x\}\big). \tag{20.48}$$

Here, $\mathbb{E}_{S_t|b_t}$ integrates over the state space for S_t using the belief distribution $b_t(s)$. $\mathbb{E}_{S_{t+1}|S_t}$ takes the expectation over S_{t+1} given S_t. Finally, $\mathbb{E}_{W^{obs}_{t+1}|S_{t+1}}$ integrates over the space of observations given we are in state S_{t+1}. These expectations would be computed using

$$V(b_t) = \max_x \left(C(b_t,x) + \sum_{s\in\mathcal{S}} b_t(s) \sum_{s'\in\mathcal{S}} p(s'|s,x) \sum_{w^{obs}\in\mathcal{W}^{obs}} P^{obs}(w^{obs}|s',x) V(B^M(b_t,x,w^{obs})) \right). \tag{20.49}$$

20.9 BIBLIOGRAPHIC NOTES

Section 20.1 - The idea of modeling control as being distributed among multiple agents has an endless array of applications, including traffic and transportation (Chen et al. (2010)), building control systems (Zhao et al. (2013)), animal science (Tang & Bennett (2010)), agriculture (Tang & Bennett (2010)), and energy (González-briones et al. (2018)), to name just a few. There are many books on the topic (Tecuci (1998) and D'Inverno & Luck (2001) are two examples), and a growing number of tutorial articles using the umbrella of "multi-agent reinforcement learning" (Chen et al. (2010), Busoniu et al. (2011) and Dorri et al. (2018)). Abara et al. (2017) provides an introduction to agent-based modeling and software.

The presentation of multiagent systems in this book follows a different style, first sketched in Powell (2021), where each agent is modeled using our universal framework. We augment our universal framework, which is inherently single-agent, with the dimension of communication. We distinguish major classes of agents, where controlling agents can use any of the four classes of policies (which is new).

Section 20.2 - The flu mitigation model is taken from Powell (2020).

Section 20.3 - This presentation of POMDPs, and the observation that the classical POMDP framework makes the assumption that the controlling agent knows the transition function for the environment, is taken from is taken from Powell (2020)

Section 20.4 - Figure 20.3 was prepared by Gunter Schemmann and has been used in a course, ORF 411, taught by Warren Powell for many years. There is an extensive literature on the newsvendor problem, and yet virtually no papers that reference

"two-agent newsvendor." This section is based on work by Brian Cheung while working as a Ph.D. student at Princeton.

Section 20.5 - While the idea of modeling a HVAC controller as an agent is quite common (see, for example, Dorri et al. (2018) and Zhao et al. (2013)), our presentation uses the framework in this book, which is new.

Section 20.6 - The blood management model uses the nonlinear value functions first presented in Godfrey & Powell (2001) and adapted for fleet management problems in Powell & Godfrey (2002). This approximation method has then been used in a spatial decomposition approach in Shapiro & Powell (2006), where the spatial decomposition can be viewed as a multiagent decomposition. This idea was then applied to a real application for managing locomotives (a system installed in 2006 and still running as of the writing of this book), described in Bouzaiene-Ayari et al. (2016). The adaptation to the blood management problem here is new, and was used because the problem is much simpler.

EXERCISES

Review questions

20.1 What types of agents are there?

20.2 We can use our universal modeling framework to model each agent, but multiagent systems do introduce new elements that only arise when we have two or more agents. At a high level, what are these new elements?

20.3 Give the state variables for agents S_{tq} and $S_{tq'}$ for the two-agent newsvendor problem in section 20.4.

Modeling questions

20.4 What has to be specified to fully describe the communication architecture of a multiagent system?

20.5 What observable information might agent q use to learn about the behavior of an agent q'? By "behavior" we are referring to the policy $X^{\pi}_{q'}(S_{tq'})$ used by agent q'.

20.6 Multiagent systems require that agents develop beliefs about other agents. Think of all the dimensions of our universal framework that we would use to describe an agent. Using the elements of this framework as a starting point, list every belief that one controlling agent might form about another controlling agent.

20.7 Model all five elements of each agent of the two-agent newsvendor problem in section 20.4 as sequential decision problems. Label the agents A and B, and put these labels in the subscripts of the variables associated with each agent. You do not have to specify the policy.

Problem solving questions

20.8 Assume there are two agents that we will designate A and B are trying to mitigate flu in their respective communities (perhaps neighboring states). We could model agents A and B using the models in section 20.2 if there were no interactions, but the populations of the two countries travel between each other, spreading infections. Let

P_{ti}^H = the true number of healthy people in state i at time t,
P_{ti}^I = the true number of infected people in state i at time t,
P_{ti}^V = the number of vaccinated people in state i at time t (this would be known to agent i),
μ_{ti} = the true fraction of the population of state i that has the flu at time t, for $i = A, B$,
ρ_{ij} = fraction of customers who move from state i to state j each time period, which applies equally to the entire population,
x_{ti}^{vac} = number of vaccinations administered in state i at time t,
x_{tij}^{share} = the number of vaccinations that state i sends to state j at time t,
x_{ti}^{test} = the number of people who are tested in state i between t and $t+1$, producing noisy estimates $\hat{P}_{t+1,i}^H$ of P_{ti}^H and $\hat{P}_{t+1,i}^I$ of P_{ti}^I (deciding to test the population at time t produces observations by time $t+1$),
$\bar{P}_{ti}^H$ = the estimated number of healthy people in state i at time t,
$\bar{P}_{ti}^I$ = the estimated number of infected people in state i at time t.

Assume initially that there is no information sharing between the two agents (e.g. of the estimates $\bar{P}_t^H$ or $\bar{P}_t^I$).

People are vaccinated without regard to their status as healthy or infected. Healthy people are protected from further infections, but infected people remain infected if the vaccination comes after their initial infection. Create three agents: environment (that covers both states), and each controlling agent. For each agent, do the following:

a) Define the state variables, decision variables, exogenous information variables and transition function (assume ρ_{ij} is known to agent i (and ρ_{ji} is known to agent j).

b) Assume the objective of each controlling agent is to minimize the number of infected people in their state. Write out the objective function for each controlling agent.

c) Keep in mind that the populations might be quite different, which means that if $P_{ti}^H \gg P_{tij}^H$, state j may have an incentive to send vaccinations to state i, given that a fraction ρ of the people in state i will travel to state j. Given this, design a simple PFA to determine x_{ti}^{vac} and x_{tij}^{share} for each state.

20.9 Assume that each agent in our flu problem is willing to share their observations $\hat{P}_t^H$ and $\hat{P}_t^I$. Further assume that the transition rates ρ_{ij} and ρ_{ji} are unknown to both agents. Describe how this affects the formulation of the state variables, exogenous information variables and the transition functions. Note that each agent will have to create their own estimate of ρ_{ij}.

20.10 The central agent for the two-agent newsvendor problem can blend requests from the field agent with external sources of knowledge as is done in equation (20.34). This

process (which we believe is quite common when dealing with people) raises a number of questions:

a) It would seem that a field agent could always do better by constantly raising his requests relative to the estimate R_t^e to stay one step ahead of the estimates of the central agent. Describe how the blending process can discourage this.

b) Describe in words how being too predictable can hurt the field agent.

c) If being too predictable can hurt the field agent, describe a noise and biasing strategy that could help the field agent. Contrast the long run effectiveness of bias versus noise.

20.11 Assume that the central agent in the two-agent newsvendor problem (section 20.4) is not allowed to see $\hat{R}_{t+1}$, but does see total cost (combining overage costs and underage costs), as might be typical in many organizations.

a) Update your models for the central agent.

b) Suggest a learning policy for the central agent given the information available to her.

20.12 For the two-agent newsvendor problem (section 20.4) we are going to introduce two changes: First, we now assume the field agent can hold excess inventory to a later time period and b) the cost of underage c^u now varies randomly over time, so we will model it as c_t^u. This means that there will be some time periods where c_t^u may be much higher than other time periods. This variation sets up an incentive for the field agent to *hoard* resources during periods where the cost of underage is low so that they might be available when the underage cost is high.

a) Update your model for the field agent. Assume that the central agent is unaware of resources being held for future time periods.

b) Suggest a policy (for the field agent) for holding inventory. Note that you may hold resources for future periods while not meeting demand in the current period.

20.13 Assume that the field agent in our two-agent newsvendor problem (section 20.4) is trying to anticipate the behavior of the central agent (there is an initial discussion of this in the text, but without details). Your goal in this exercise is to fill in the details.

a) Suggest a belief model for the field agent of the behavior of the central agent.

b) Incorporate this belief model into your state variable, and outline the five elements of a sequential decision system for the field agent. Be sure to include the updating equations for all the state variables in your transition function, including any that are needed for your belief model of the central agent.

c) Design an ordering policy for the field agent. If your policy required introducing additional state variables, be sure to update your model in part (b).

20.14 Assume that the central agent in our two-agent newsvendor problem (section 20.4) is trying to anticipate the behavior of the field agent, without any assumptions that the field

is trying to anticipate the central agent (as we did in exercise 20.13). Your goal in this exercise is to fill in the details.

a) Suggest a belief model for the central agent of the behavior of the field agent.

b) Incorporate this belief model into your state variable, and outline the five elements of a sequential decision system for the central agent. Be sure to include the updating equations for all the state variables in your transition function, including any that are needed for your belief model of the field agent.

c) Design an ordering policy for the central agent. If your policy required introducing additional state variables, be sure to update your model in part (b).

20.15 (Advanced) Repeat exercise 20.14 assuming that you have already done exercise 20.13, which means you are modeling the central agent while anticipating that the field agent is trying to anticipate the behavior of the central agent.

Sequential decision analytics and modeling

These exercises are drawn from the online book *Sequential Decision Analytics and Modeling* available at `http://tinyurl.com/sdaexamplesprint`.

20.16 This exercise will focus on the multiagent problems in chapters 10 and 11, so please begin by reviewing this material. This exercise will use the Python module "TwoNewsvendor" available at `http://tinyurl.com/sdagithub`.

a) Run the code with the basic learning model and see what the two players settle for in the end (the biases that get chosen more often).

b) Fix the central command bias at -4. Make the code run the problem with the field agent treating it as a learning process with several values for the UCB parameter (1, 2, 4, 8, 16, 32) and make a graph round vs choice to see how the learning policy selects what biases to use.

c) Using code from the first two-newsvendor model and code from the modules for the learning approach, write your own module where the field agent treats the problem as a learning problem, each round choosing a bias in $[0, 10]$, and the central command treats the problem using the strategy from the first model (computes bias, adds his own bias plus some noise).

d) Consider now a two-newsvendor problem where the central command also has some external information about the demand. What he has is a much noisier estimate of the demand (say the noise is, for our spreadsheet data where the demand is always between 20 and 40, 3 times bigger than the noise the noise from the source communicating with the field agent). Redefine the bias from the central command as the quantity that he adds to the estimate he gets. Try a learning approach where the bias he selects is chosen in the interval $[-5, 5]$. Run the program and compare the results with the old learning process.

e) (Punishing strategy 1) Consider the case where the field agent is using a learning approach and the central command is using a punishing strategy. Since he knows the field gets a bigger penalty for providing less than the demand, the central command

will compute the previous field bias (for time $t-1$) and if it is positive, it will apply a bias twice as big in magnitude and of opposite sign to the field's request. Run this experiment and see what the field's prevalent bias will be after the 4000 rounds.

Diary problem

The diary problem is a single problem you chose (see chapter 1 for guidelines). Answer the following for your diary problem.

20.17 If your diary problem has either the dimension of learning (such as learning an unknown environment), or multiple decision makers, you can use the framework in this chapter. Given this, do the following

a) Describe each agent, and characterize each as either an environment agent, a controlling agent (which may or may not have learning), and possibly a pure learning agent.

b) For each controlling agent, identify belief variables that capture beliefs that the controlling agent would have to create about unknown parameters, quantities. If there is more than one controlling agent, it will be necessary for each agent to develop beliefs about other controlling agents, but leave this for part (c). While it is important to introduce notation, start by describing the beliefs and unknown quantities in words.

c) If there is more than one controlling agent, it will be necessary to create beliefs about other controlling agents.

d) For each agent, create a full model including all five elements (decisions and objective functions will be missing for agents other than controlling agents).

Bibliography

Abara, S., Lemarinier, G., K.Theodoropoulos, P. & M.P.O'Hared, G. (2017), 'Agent Based Modelling and Simulation tools: A review of the state-of-art software', *Computer Science Review* **24**, 13–33.

Bouzaiene-Ayari, B., Cheng, C., Das, S., Fiorillo, R. & Powell, W. B. (2016), 'From single commodity to multiattribute models for locomotive optimization: A comparison of optimal integer programming and approximate dynamic programming', *Transportation Science* **50**(2), 1–24.

Busoniu, L., Babuska, R. & Schutter, B. D. (2011), 'A Comprehensive Survey of Multiagent Reinforcement Learning', *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews* **38**(2), 156–172.

Chen, B., Cheng, H. H. & Member, S. (2010), 'A Review of the Applications of Agent Technology in Traffic and Transportation Systems', *IEEE Transactions on Intelligent Transportation Systems* **11**(2), 485–497.

D'Inverno, M. & Luck, M. (2001), *Understanding Agent Systems*, Springer, New York.

Dorri, A. L. I., Member, S., Kanhere, S. S. & Member, S. (2018), 'Multi-Agent Systems : A Survey', *IEEE Transactions on Industry Applications* **49**(1), 28573–28593.

Godfrey, G. A. & Powell, W. B. (2001), 'An Adaptive, Distribution-Free Algorithm for the Newsvendor Problem with Censored Demands, with Applications to Inventory and Distribution', *Management Science* **47**(8), 1101–1112.

González-briones, A., Prieta, F. D. L., Omatu, S. & Corchado, J. M. (2018), 'Multi-Agent Systems Applications in Energy Optimization Problems : A State-of-the-Art Review', *Energies* **11**, 1–28.

Powell, W. B. (2020), 'On state variables, bandit problems and POMDPs'.

Powell, W. B. (2021), 'From reinforcement learning to optimal control: A unified framework for sequential decisions', *Handbook on Reinforcement Learning and Optimal Control, Studies in Systems, Decision and Control* pp. 29–74.

Powell, W. B. & Godfrey, G. A. (2002), 'An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times', *Transportation Science* **36**(1), 40–54.

Shapiro, J. A. & Powell, W. B. (2006), 'A metastrategy for large-scale resource management based on informational decomposition', *INFORMS Journal on Computing* **18**(1), 43–60.

Tang, W. & Bennett, D. A. (2010), 'Agent-based Modeling of Animal Movement : A Review', *Geography Compass* **7**, 682–700.

Tecuci, G. (1998), *Building Intelligent Agents*, Academic Press.

Zhao, P., Member, S., Suryanarayanan, S., Member, S., Simões, M. G. & Member, S. (2013), 'An Energy Management System for Building Structures Using a Multi-Agent Decision-Making Control Methodology', *IEEE Transactions on Industry Applications* **49**(1), 322–330.