REINFORCEMENT LEARNING AND STOCHASTIC OPTIMIZATION

A unified framework for sequential decisions

Warren B. Powell

June 30, 2021

WILEY-
INTERSCIENCE

A JOHN WILEY & SONS, INC., PUBLICATION

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

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created ore extended by sales representatives or written sales materials. The advice and strategies contained herin may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services please contact our Customer Care Department with the U.S. at 877-762-2974, outside the U.S. at 317-572-3993 or fax 317-572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print, however, may not be available in electronic format.

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 III - STATE-DEPENDENT PROBLEMS

We now transition to a much richer class of dynamic problems where some aspect of the *problem* depends on dynamic information. This might arise in three ways:

- The objective function depends on dynamic information, such as a cost or price.
- The constraints may depend on the availability of resources (that are being controlled dynamically), or other information in constraints such as the travel time in a graph or the rate at which water is evaporating.
- The distribution of a random variable such as weather, or the distribution of demand, may be varying over time, which means the parameters of the distribution are in the state variable.

When we worked on state-independent problems, we often wrote the function being maximized as $F(x, W)$ to express the dependence on the decision x or random information W, but not on any information in our state S_t (or S^n). As we move to our state-dependent world, we are going to write our cost or contribution function as $C(S_t, x_t)$ or, in some cases, $C(S_t, x_t, W_{t+1})$, to capture the possible dependence of the objective function on dynamic information in S_t. In addition, our decision x_t might be constrained by $x_t \in \mathcal{X}_t$, where the constraints $\mathcal{X}_t$ may depend on dynamic data such as inventories, travel times or conversion rates.

Finally, our random information W may itself depend on known information in the state variable S_t, or possibly on hidden information that we cannot observe, but have beliefs about (these beliefs would also be captured in the state variable). For example, W might be the number of clicks on an ad which is described by some probability distribution whose parameters (e.g. the mean) is also uncertain. Thus, at time t (or time n), we may find ourselves solving a problem that looks like

$$\max_{x_t \in \mathcal{X}_t} \mathbb{E}_{S_t}\mathbb{E}_{W|S_t}\{C(S_t, x_t, W_{t+1})|S_t\}.$$

If the cost/contribution function $C(S_t, x_t, W_{t+1})$, and/or the constraints $\mathcal{X}_t$, and/or the expectation depends on time-dependent data, then we have an instance of a state-dependent problem.

We are not trying to say that all state-dependent problems are the same, but we do claim that state-dependent problems represent an important transition from state-independent problems, where the only state is the belief B_t about our function. This is why we also refer to state-independent problems as learning problems.

We lay the foundation for state-dependent problems with the following chapters:

- State-dependent applications (chapter 8) - We begin our presentation with a series of applications of problems where the function is state dependent. State variables can arise in the objective function (e.g. prices), but in most of the applications the state arises in the constraints, which is typical of problems that involve the management of physical resources.

- Modeling general sequential decision problems (chapter 9) - This chapter provides a comprehensive summary of how to model general (state-dependent) sequential decision problems in all of their glory.

- Modeling uncertainty (chapter 10) - To find good policies (to make good decisions), you need a good model, and this means an accurate model of uncertainty. In this chapter we identify different sources of uncertainty and discuss how to model them.

- Designing policies (chapter 11) - Here we provide a much more comprehensive overview of the different strategies for creating policies, leading to the four classes of policies that we first introduce in part I for learning problems. If you have a particular problem you are trying to solve (rather than just building your toolbox), this chapter should guide you to the policies that seem most relevant to your problem.

After these chapters, the remainder of the book is a tour through the four classes of policies which we illustrated in chapter 7 in the context of derivative-free stochastic optimization.

CHAPTER 8

STATE-DEPENDENT PROBLEMS

In chapters 5 and 7, we introduced sequential decision problems in which the state variable consisted only of the state of the algorithm (chapter 5) or the state of our belief about an unknown function $\mathbb{E}\{F(x,W)|S_0\}$ (chapter 7). These problems cover a very important class of applications that involve maximizing or minimizing functions that can represent anything from complex analytical functions and black-box simulators to laboratory and field experiments.

The distinguishing feature of state-independent problems is that the *problem* being optimized does not depend on our state variable, where the "problem" might be the function $F(x,W)$, the expectation (e.g. the distribution of W), or the feasible region $\mathcal{X}$. For these problems, the state variables impact decisions only through the policy.

There is a genuinely vast range of problems where the performance metric (costs or contributions), the distributions of random variables W, and/or the constraints, depend on information that is changing over time, either exogenously or as a result of decisions (or both). When information changes over time, it is captured in the state variable S_t (or S^n if we are counting events with n).

Examples of state variables that affect the problem itself include:

- Physical state variables, which might include inventories, the location of a vehicle on a graph, the medical condition of a patient, the speed and location of a robot, and the condition of an aircraft engine. Physical state variables are typically expressed through the constraints.

- Informational state variables, such as prices, a patient's medical history, the humidity in a lab, or attributes of someone logging into the internet. These variables might affect the objective function (costs and contributions), or the constraints. This information may evolve exogenously (e.g. the weather), or might be directly controlled (e.g. setting the price of a product), or influenced by decisions (selling energy into the grid may lower electricity prices).

- Distributional information, which means information about the how a patient might respond to a drug, or the state of the materials in a jet engine, or how the market might respond to the price of a product.

While physical resource management problems are perhaps the easiest to envision, state dependent problems can include any problem where the function being minimized depends on dynamic information, either in the objective function itself, or the constraints, or the equations that govern how the system evolves over time (the transition function).

For our state-independent problems, we wrote the objective function as $F(x, W)$, since the function itself did not depend on the state variable. For state-dependent problems, we will usually write our single-period contribution (or cost) function as $C(S_t, x_t)$, although there are settings where it is more natural to write it as $C(S_t, x_t, W_{t+1})$ or, in some settings, $C(S_t, x_t, S_{t+1})$.

We will communicate the dependence of expectations on the state variables through conditioning, by writing $\mathbb{E}\{F(\cdot)|S_t\}$ (or $\mathbb{E}\{F(\cdot)|S^n\}$). We will express the dependence of constraints on dynamic state information by writing $x \in \mathcal{X}_t$. Note that writing $C(S_t, x_t)$ means that the contribution function depends on dynamic information such as

$$C(S_t, x_t) = p_t x_t,$$

where the price p_t evolves randomly over time.

At this point, it is useful to highlight what is probably the biggest class of state-dependent problems, which is those that involve the management of physical resources. Generally known as *dynamic resource allocation problems*, these problems are the basis of the largest and most difficult problems that we will encounter. These problems are typically high-dimensional, often with complex dynamics and types of uncertainty.

In this chapter we present four classes of examples:

- Graph problems - These are problems where we are modeling a single resource that is controlled using a discrete set of actions moving over a discrete set of states.

- Inventory problems - This is a classical problem in dynamic programming which comes in a virtually unlimited set of variations.

- Information acquisition problems - These are state-dependent active learning problems that we touched on at the end of chapter 7.

- Complex resource allocation problems - Here we put our toe in the water and describe some high-dimensional applications.

These illustrations are designed to teach by example. The careful reader will pick up subtle modeling choices, in particular the indexing with respect to time. We suggest that readers skim these problems, selecting examples that are of interest. In chapter 9, we are going to present a very general modeling framework, and it helps to have a sense of the complexity of applications that may arise.

8.1 GRAPH PROBLEMS

A popular class of stochastic optimization problems involve managing a single physical asset moving over a graph, where the nodes of the graph capture the physical state.

8.1.1 A stochastic shortest path problem

We are often interested in shortest path problems where there is uncertainty in the cost of traversing a link. For our transportation example, it is natural to view the travel time on a link as random, reflecting the variability in traffic conditions on each link. There are two ways we can handle this uncertainty. The simplest is to assume that our driver has to make a decision before seeing the travel time over the link. In this case, our updating equation would look like

$$v_i^n = \min_{j \in \mathcal{I}_i^+} \mathbb{E}\{\hat{c}_{ij} + v_j^{n-1}\}$$

where $\hat{c}_{ij}$ is a random variable describing the cost of traversing i to j. If $\bar{c}_{ij} = \mathbb{E}\hat{c}_{ij}$, then our problem reduces to

$$v_i^n = \min_{j \in \mathcal{I}_i^+} \left(\bar{c}_{ij} + v_j^{n-1}\right)$$

which is a simple deterministic problem.

An alternative model is to assume that we know the cost on a link from i to j as soon as we arrive at node i. In this case, we would have to solve

$$v_i^n = \mathbb{E}\left\{\min_{j \in \mathcal{I}_i^+} \left(\hat{c}_{ij} + v_j^{n-1}\right)\right\}.$$

Here, the expectation is outside of the $\min$ operator that chooses the best decision, capturing the fact that now the decision itself is random.

Note that our notation is ambiguous, in that with the same notation, we have two very different models. In chapter 9, we are going to refine our notation so that it will be immediately apparent when a decision "sees" the random information and when the decision has to be made before the information becomes available.

8.1.2 The nomadic trucker

A nice illustration of sequential decisions is a problem we are going to call the nomadic trucker, depicted in figure 8.1. In this problem, our trucker has to move loads of freight (which fill his truck) from one city to the next. When he arrives in a city i ("Texas" in figure 8.1), he is offered a set of loads to different destinations, and has to choose one. Once he makes his choice (in the figure, he chooses the load to New Jersey), he moves the load to its destination, delivers the freight and then the problem repeats itself. The other loads are offered to other drivers, so if he returns to node i at a later time, he is offered an entirely new set of loads (that are entirely random).

We model the state of our nomadic trucker by letting R_t be his location ("Texas" in the figure). From a location, our trucker is able to choose from a set of demands $\hat{D}_t$. Thus, our state variable is $S = (R_t, \hat{D}_t)$, where R_t is a scalar (the location) while $\hat{D}_t$ is a vector

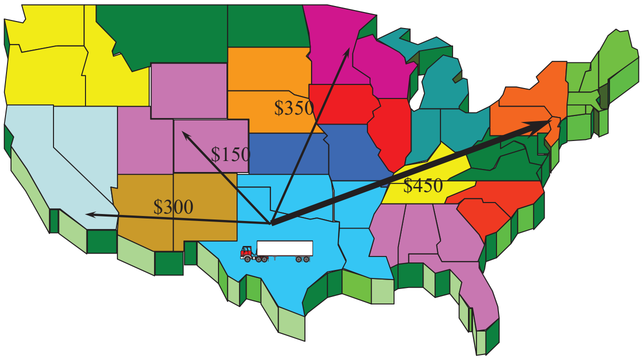


Figure 8.1 Illustration of a nomadic trucker in location "Texas" with the choice of four loads to move.

giving the number of loads from R_t to each possible destination. A decision $x_t \in \mathcal{X}_t$ represents the decision to accept a load in $\hat{D}_t$ and go to the destination of that load.

Let $C(S_t, x_t)$ be the contribution earned from being in location R_t (this is contained in S_t) and taking decision x_t. Any demands not covered in $\hat{D}_t$ at time t are lost. After implementing decision x_t, the driver either stays in his current location (if he does nothing), or moves to a location that corresponds to the destination of the load the driver selected in the set $\hat{D}_t$.

Let R_t^x be the location that decision x_t sends the driver to (we will later call this the *post-decision state*, which is the state after we have made a decision but before any new information has arrived). The post-decision state variable $S_t^x = R_t^x$ is the location the truck will move to, but before any demands have been revealed. We assume that the decision x_t determines the downstream destination, so $R_{t+1} = R_t^x$.

The driver makes his decision by solving

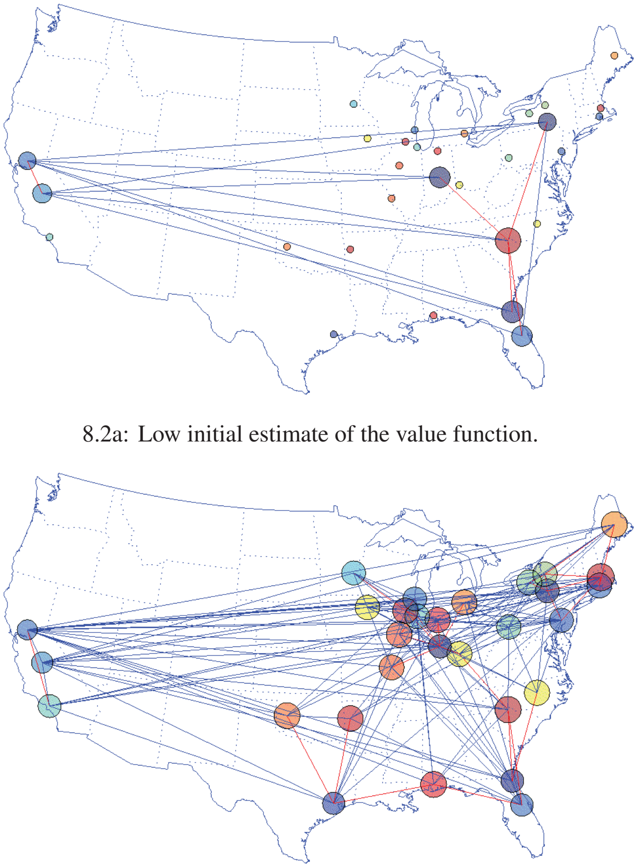
$$\hat{v}_t = \max_{x \in \hat{D}_t} \left(C(S_t, x) + \gamma \overline{V}_t^x (R_t^x) \right),$$

where R_t^x is the downstream location (the "post-decision state"), and $\overline{V}_t(R_t^x)$ is our current estimate (as of time t) of the value of the truck being in the destination R_t^x. Let x_t be the best decision given the downstream values $\overline{V}_t^x(R_t^x)$. Noting that R_t is the current location of the truck, we update the value of our previous, post-decision state using

$$\overline{V}_{t-1}^x(R_{t-1}^x) \leftarrow (1-\alpha)\overline{V}_t(R_t) + \alpha \hat{v}_t.$$

We start by initializing the value of being in each location to zero, and use a pure exploitation strategy. If we simulate 500 iterations of this process, we produce the pattern shown in figure 8.2a. Here, the circles at each location are proportional to the value $\overline{V}^{500}(R)$ of being in that location. The small circles indicate places where the trucker never visited. Out of 30 cities, our trucker has ended up visiting seven.

This problem is an illustration of the exploration-exploitation problem, which we first saw in chapter 7. However, now we have to deal with a physical state, where what we learn depends on the location of our trucker. In our derivative-free problems in chapter 7, we always faced the same set of possible choices at each step. Our nomadic trucker, on



8.2a: Low initial estimate of the value function.

8.2b: High initial estimate of the value function.

Figure 8.2 The effect of value function initialization on search process. Case (a) uses a low initial estimate and produces limited exploration; case (b) uses a high initial estimate, which forces exploration of the entire state space.

the other hand, faces different choices each time he moves from one city to the next. This changes the value of what he learns, since the decisions are different in each state.

8.1.3 The transformer replacement problem

The electric power industry uses equipment known as transformers to convert the high-voltage electricity that comes out of power generating plants into currents with successively lower voltage, finally delivering the current we can use in our homes and businesses. The largest of these transformers can weigh 200 tons, might cost millions of dollars to replace and may require a year or more to build and deliver. Failure rates are difficult to estimate (the most powerful transformers were first installed in the 1960's and have yet to reach the end of their natural lifetime). Actual failures can be very difficult to predict, as they often depend on heat, power surges, and the level of use.

We are going to build an aggregate replacement model where we only capture the age of the transformers. Let

$$
\begin{aligned}
a &= \text{the age of a transformer (in units of time periods) at time } t,\\
R_{ta} &= \text{the number of active transformers of age } a \text{ at time } t.
\end{aligned}
$$

Here and elsewhere, we need to model the attributes of a resource (in this case, the age).

For our model, we assume that age is the best predictor of the probability that a transformer will fail. Let

$$
\begin{aligned}
\hat{R}^{fail}_{t+1,a} &= \text{the number of transformers of age } a \text{ that fail between } t \text{ and } t+1,\\
p_a &= \text{the probability a transformer of age } a \text{ will fail between } t \text{ and } t+1.
\end{aligned}
$$

Of course, $\hat{R}^{fail}_{t+1,a}$ depends on R_{ta} since transformers can only fail if we own them.

It can take a year or two to acquire a new transformer. Assume that we are measuring time in quarters (three-month periods). Normally it can take about six quarters from the time of purchase before a transformer is installed in the network. However, we may pay extra and get a new transformer in as little as three quarters. If we purchase a transformer that arrives in six time periods, then we might say that we have acquired a transformer that is $a = -6$ time periods old. Paying extra gets us a transformer that is $a = -3$ time periods old. Of course, the transformer is not productive until it is at least $a = 0$ time periods old. Let

$$
x_{ta} = \text{Number of transformers of age } a \text{ that we purchase at time } t.
$$

The transition function is given by

$$
R_{t+1,a} \quad = \quad R_{t,a-1} + x_{t,a-1} - \hat{R}^{fail}_{t+1,a}.
$$

If we have too few transformers, then we incur what are known as "congestion costs," which represent the cost of purchasing power from more expensive utilities because of bottlenecks in the network. To capture this, let

$$
\begin{aligned}
\bar{R} &= \text{target number of transformers that we should have available,}\\
R^A_t &= \text{actual number of transformers that are available at time } t,\\
&= \sum_{a \geq 0} R_{ta},\\
c_a &= \text{the cost of purchasing a transformer of age } a,\\
C_t(R^A_t, \bar{R}) &= \text{expected congestion costs if } R^A_t \text{ transformers are available,}\\
&= c_0 \left(\frac{\bar{R}}{R^A_t} \right)^{\beta}.
\end{aligned}
$$

The function $C_t(R^A_t, \bar{R})$ captures the behavior that as R^A_t falls below $\bar{R}$, the congestion costs rise quickly.

The total cost function is then given by

$$
C(S_t, x_t) = C_t(R^A_t, \bar{R}) + c_a x_t.
$$

For this application, our state variable R_t might have as many as 100 dimensions. If we have, say, 200 transformers, each of which might be as many as 100 years old, then the number of possible values of R_t could be 100^{200}. It is not unusual for modelers to count the size of the state space, although this is an issue only for particular solution methods that depend on lookup table representations of the value of being in a state, or the action we should take given that we are in a state.

8.1.4 Asset valuation

Imagine you are holding an asset that you can sell at a price that fluctuates randomly. In this problem we want to determine the best time to sell the asset, and from this, infer the value of the asset. For this reason, this type of problem arises frequently in the context of asset valuation and pricing.

Let p_t be the price at which we can sell our asset at time t, at which point you have to make a decision

$$x_t = \begin{cases} 1 & \text{sell,} \\ 0 & \text{hold.} \end{cases}$$

For our simple model, we assume that p_t is independent of prior prices (a more typical model would assume that the *change* in price is independent of prior history). With this assumption, our system has two physical states that we denote by R_t, where

$$R_t = \begin{cases} 1 & \text{we are holding the asset,} \\ 0 & \text{we have sold the asset.} \end{cases}$$

Our state variable is then given by

$$S_t = (R_t, p_t).$$

Let

$\tau =$ the time at which we sell our asset.

τ is known as a *stopping time*, which means it can only depend on information that has arrived on or before time t. By definition, $x_\tau = 1$ indicates the decision to sell at time $t = \tau$. It is common to think of τ as the decision variable, where we wish to solve

$$\max_\tau \mathbb{E} p_\tau. \tag{8.1}$$

Equation (8.1) is a little tricky to interpret. Clearly, the choice of when to stop is a random variable since it depends on the price p_t. We cannot optimally choose a random variable, so what is meant by (8.1) is that we wish to choose a *function* (or *policy*) that determines when we are going to sell. For example, we would expect that we might use a rule that says

$$X_t^{PFA}(S_t|\theta^{sell}) = \begin{cases} 1 & \text{if } p_t \geq \theta^{sell} \text{ and } S_t = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{8.2}$$

In this case, we have a function parameterized by θ^{sell} which allows us to write our problem in the form

$$\max_{\theta^{sell}} \mathbb{E} \left\{ \sum_{t=0}^{\infty} \gamma^t p_t X_t^{PFA}(S_t|\theta^{sell}) \right\}, \tag{8.3}$$

where $\gamma < 1$ is a discount factor. This formulation raises two questions. First, while it seems very intuitive that our policy would take the form given in equation (8.2), there is the theoretical question of whether this in fact is the structure of an optimal policy.

The second question is how to find the best policy within this class. For this problem, that means finding the parameter θ^{sell}. This is precisely the type of problem that we addressed in our stochastic search chapters 5 and 7. However, this is not the only policy we might use. Another is to define the function

$V_t(S_t)$ = the value of being in state S_t at time t and then making optimal decisions from time t onward.

More practically, let $V^\pi(S_t)$ be the value of being in state S_t and then following policy π from time t onward. This is given by

$$V_t^\pi(S_t) = \mathbb{E}\left\{\sum_{t'=t}^{\infty} \gamma^{t'-t} p_{t'} X_{t'}^\pi(S_{t'}|\theta^{sell})\right\}.$$

Of course, it would be nice if we could find an optimal policy since this would maximize $V^\pi(S_t)$. More often, we need to use some approximation that we call $\overline{V}_t(S_t)$. In this case, we might define a policy

$$X^{VFA}(S_t) = \arg\max_{x_t} \left(p_t x_t + \gamma \mathbb{E}\{\overline{V}_{t+1}(S_{t+1})|S_t, x_t\}\right). \tag{8.4}$$

We have just illustrated two styles of policies: X^{PFA} and X^{VFA}. These are two of the four classes we first visited in chapter 7, called policy function approximation and value function approximation. We will again review all four classes of policies in chapter 11, which we will discuss in depth in chapters 12 - 19.

8.2 INVENTORY PROBLEMS

Another popular class of problems involving managing a quantity of resources that are held in some sort of inventory. The inventory can be money, products, blood, people, water in a reservoir or energy in a battery. The decisions govern the quantity of resource moving into and out of the inventory.

8.2.1 A basic inventory problem

A basic inventory problem arises in applications where we purchase product at time t to be used during time interval $t+1$. We are going to encounter this problem again, sometimes as discrete problems, but often as continuous problems, and sometimes as vector valued problems (when we have to acquire different types of assets).

We can model the problem using

R_t = the inventory on hand at time t before we make a new ordering decision, and before we have satisfied any demands arising in time interval t,

x_t = the amount of product purchased at time t which we assume arrives immediately,

D_t = the demand known at time t that we have to satisfy.

We have chosen to model R_t as the resources on hand in period t before demands have been satisfied. Our definition here makes it easier to introduce (in the next section) the decision of how much demand we should satisfy. In our most basic problem, the state variable S_t is given by

$$S_t = (R_t, D_t).$$

Our inventory R_t is described using the equation

$$R_{t+1} = R_t - \min\{R_t, D_t\} + x_t.$$

Let

$$\hat{D}_{t+1} \quad = \quad \text{new demands that we learn about during time interval } (t, t+1).$$

We assume that any unsatisfied demands are lost. This means that D_t evolves according to

$$D_{t+1} = \hat{D}_{t+1}.$$

Here we are assuming that D_{t+1} is revealed to us through the new information $\hat{D}_{t+1}$. Below we are going to introduce the ability to backlog unsatisfied demands.

We assume we purchase new assets at a fixed price p^{buy} and sell them at a fixed price p^{sell}. The amount we earn between $t-1$ and t (satisfying the demand D_t that becomes known by time t), including the decision we make at time t, is given by

$$C(S_t, x_t) = p^{sell} \min\{R_t, D_t\} - p^{buy} x_t.$$

An alternative formulation of this problem is to write the contribution based on what we will receive between t and $t+1$. In this case, we would write the contribution as

$$C(S_t, x_t, \hat{D}_{t+1}) = p^{sell} \min\{\big(R_t - \min\{R_t, D_t\} + x_t\big), \hat{D}_{t+1}\} - p^{buy} x_t. \tag{8.5}$$

It is because of problems like this that we sometimes write our contribution function as $C(S_t, x_t, W_{t+1})$.

8.2.2 The inventory problem - II

Many inventory problems introduce additional sources of uncertainty. The inventory we are managing could be stocks, planes, energy commodities such as oil, consumer goods, and blood. In addition to the need to satisfy random demands (the only source of uncertainty we considered in our basic inventory problem), we may also have randomness in the prices at which we buy and sell assets. We may also include exogenous changes to the inventory on hand due to additions (cash deposits, blood donations, energy discoveries) and subtractions (cash withdrawals, equipment failures, theft of product).

We can model the problem using

$$\begin{aligned}
x_t^{buy} &= \text{inventory purchased at time } t \text{ to be used during time interval } t+1,\\
x_t^{sell} &= \text{amount of inventory sold to satisfy demands during time interval } t,\\
x_t &= (x_t^{buy}, x_t^{sell}),\\
R_t &= \text{inventory level at time } t \text{ before any decisions are made},\\
D_t &= \text{demands waiting to be served at time } t.
\end{aligned}$$

Of course, we are going to require that $x_t^{sell} \leq \min\{R_t, D_t\}$, since we cannot sell what we do not have, and we cannot sell more than the market demand. We are also going to assume that we buy and sell our inventory at market prices that fluctuate over time. These are described using

$$\begin{aligned} p_t^{buy} &= \text{market price for purchasing inventory at time } t, \\ p_t^{sell} &= \text{market price for selling inventory at time } t, \\ p_t &= (p_{t+1}^{sell},\, p_{t+1}^{buy}). \end{aligned}$$

Our system evolves according to several types of exogenous information processes that include random changes to the supplies (inventory on hand), demands and prices. We model these using

$\hat{R}_{t+1}$ = exogenous changes to the inventory on hand that occur during time interval $(t, t+1)$ (e.g. rainfall adding water to a reservoir, deposits/withdrawals of cash to a mutual fund, or blood donations),

$\hat{D}_{t+1}$ = new demands for inventory that arises during time interval $(t, t+1)$,

$\hat{p}_{t+1}^{buy}$ = change in the purchase price that occurs during time interval $(t, t+1)$,

$\hat{p}_{t+1}^{sell}$ = change in the selling price that occurs during time interval $(t, t+1)$,

$\hat{p}_{t+1}$ = $(\hat{p}_{t+1}^{buy}, \hat{p}_{t+1}^{sell})$.

We assume that the exogenous changes to inventory, $\hat{R}_t$, occurs before we satisfy demands at time t.

For more complex problems such as this, it is convenient to have a generic variable for exogenous information. We use the notation W_{t+1} to represent all the information that first arrives between t and $t+1$, where for this problem, we would have

$$W_{t+1} = (\hat{R}_{t+1}, \hat{D}_{t+1}, \hat{p}_{t+1}).$$

The state of our system is described by

$$S_t = (R_t, D_t, p_t).$$

The state variables evolve according to

$$\begin{aligned} R_{t+1} &= R_t - x_t^{sell} + x_t^{buy} + \hat{R}_{t+1}, \\ D_{t+1} &= D_t - x_t^{sell} + \hat{D}_{t+1}, \\ p_{t+1}^{buy} &= p_t^{buy} + \hat{p}_{t+1}^{buy}, \\ p_{t+1}^{sell} &= p_t^{sell} + \hat{p}_{t+1}^{sell}. \end{aligned}$$

We can add an additional twist if we assume the market price, for instance, follows a time-series model

$$p_{t+1}^{sell} = \theta_0 p_t^{sell} + \theta_1 p_{t-1}^{sell} + \theta_2 p_{t-2}^{sell} + \varepsilon_{t+1},$$

where $\varepsilon_{t+1} \sim N(0, \sigma_\varepsilon^2)$. In this case, the state of our price process is captured by $(p_t^{sell}, p_{t-1}^{sell}, p_{t-2}^{sell})$ which means our state variable would be given by

$$S_t = \big(R_t, D_t, (p_t, p_{t-1}, p_{t-2})\big).$$

Note that if we did not allow backlogging, then we would update demands with just

$$D_{t+1} = \hat{D}_{t+1}. \tag{8.6}$$

Contrast this with our updating of the prices p_{t+1} which depends on either p_t or even p_{t-1} and p_{t-2}. To model the evolution of prices, we have an explicit mathematical model, including an assumed error such as ε_{t+1} where we assumed $\varepsilon_{t+1} \sim N(0, \sigma^2_\varepsilon)$. When we simply observe the updated value of demand (as we are doing in (8.6)), then we describe the process as "data driven." We would need a source of data from which to draw the observations $\hat{D}_1, \hat{D}_2, \dots, \hat{D}_t, \dots$. We revisit this concept in more depth in chapter 10.

The one-period contribution function is

$$C_t(S_t, x_t) = p_t^{sell} x_t^{sell} - p_t^{buy} x_t.$$

8.2.3 The lagged asset acquisition problem

A variation of the basic asset acquisition problem we introduced in section 8.2.1 arises when we can purchase assets now to be used in the future. For example, a hotel might book rooms at time t for a date t' in the future. A travel agent might purchase space on a flight or a cruise line at various points in time before the trip actually happens. An airline might purchase contracts to buy fuel in the future. In all of these cases, it will generally be the case that assets purchased farther in advance are cheaper, although prices may fluctuate.

For this problem, we are going to assume that selling prices are

$x_{tt'}$ = resources purchased at time t to be used to satisfy demands that become known during time interval between $t'-1$ and t',

x_t = $(x_{t,t+1}, x_{t,t+2}, \dots,),$

= $(x_{tt'})_{t'>t},$

$D_{tt'}$ = total demand known at time t to be served at time t',

D_t = $(D_{tt'})_{t'\geq t},$

$R_{tt'}$ = inventory acquired on or before time t that may be used to satisfy demands that become known between $t'-1$ and t',

R_t = $(R_{tt'})_{t'\geq t}.$

Now, R_{tt} is the resources on hand in period t that can be used to satisfy demands D_t that become known during time interval t. In this formulation, we do not allow x_{tt}, which would represent purchases on the spot market. If this were allowed, purchases at time t could be used to satisfy unsatisfied demands arising during time interval between $t-1$ and t.

After we make our decisions x_t, we observe new demands

$\hat{D}_{t+1,t'}$ = new demands for the resources that become known during time interval $(t, t+1)$ to be served at time t'.

The state variable for this problem would be

$$S_t = (R_t, D_t),$$

where R_t is the vector capturing inventory that will arrive in the future.

The transition equation for R_t is given by

$$R_{t+1,t'} = \begin{cases} (R_{t,t} - \min(R_{tt}, D_{tt})) + x_{t,t+1} + R_{t,t+1}, & t' = t+1, \\ R_{tt'} + x_{tt'}, & t' > t+1. \end{cases}$$

The transition equation for D_t is given by

$$D_{t+1,t'} \quad = \quad \begin{cases} (D_{tt} - \min(R_{tt}, D_{tt})) + \hat{D}_{t,t+1} + D_{t,t+1}, & t' = t+1, \\ D_{tt'} + \hat{D}_{t+1,t'}, & t' > t+1. \end{cases}$$

To compute profits, let

p_t^{sell} = the sales price, which varies stochastically over time as it did earlier,

$p_{t,t'-t}^{buy}$ = the purchase price, which depends on both time t as well as how far into the future we are purchasing.

The one-period contribution function (measuring forward in time) is

$$C_t(S_t, x_t) = p_t^{sell} \min(R_{tt}, D_{tt}) - \sum_{t'>t} p_{t,t'-t}^{buy} x_{tt'}.$$

Note that we index the contribution function $C_t(S_t, x_t)$ by time t. This is not because the prices p_t^{sell} and $p_{t,\tau}^{buy}$ depend on time. This information is captured in the state variable S_t. Rather, it is because of the sum $\sum_{t'>t}$ which is depends on t.

8.2.4 The batch replenishment problem

One of the classical problems in operations research is one that we refer to here as the batch replenishment problem. To illustrate the basic problem, assume that we have a single type of resource that is consumed over time. As the reserves of the resource run low, it is necessary to replenish the resources. In many problems, there are economies of scale in this process. It is more economical to increase the level of resources in one jump (see examples).

■ **EXAMPLE 8.1**

An oil company maintains an aggregate level of oil reserves. As these are depleted, it will undertake exploration expeditions to identify new oil fields, which will produce jumps in the total reserves under the company's control.

■ **EXAMPLE 8.2**

A startup company has to maintain adequate reserves of operating capital to fund product development and marketing. As the cash is depleted, the finance officer has to go to the markets to raise additional capital. There are fixed costs of raising capital, so this tends to be done in batches.

■ **EXAMPLE 8.3**

A delivery vehicle for an e-commerce food delivery company would like to do several deliveries at the same time. As orders come in, it has to decide whether to continue waiting or to leave with the orders that it has already accumulated.

To introduce the core elements, let

$$\begin{aligned} D_t &= \text{demand waiting to be served at time } t, \\ R_t &= \text{resource level at time } t, \\ x_t &= \text{additional resources acquired at time } t \text{ to be used during time interval } t+1. \end{aligned}$$

Our state variable is

$$S_t = (R_t, D_t).$$

After we make our decision x_t of how much new product to order, we observe new demands

$$\hat{D}_{t+1} = \text{new demands that arrive during the interval } (t, t+1).$$

The transition function is given by

$$\begin{aligned} R_{t+1} &= \max\{0, (R_t + x_t - D_t)\}, \\ D_{t+1} &= D_t - \min\{R_t + x_t, D_t\} + \hat{D}_{t+1}. \end{aligned}$$

Our one-period cost function (which we wish to minimize) is given by

$$\begin{aligned} C(S_t, x_t, \hat{D}_{t+1}) &= \text{total cost of acquiring } x_t \text{ units of the resource} \\ &= c^f I_{\{x_t > 0\}} + c^p x_t + c^h R^M_{t+1}(R_t, x_t, \hat{D}_{t+1}), \end{aligned}$$

where

$$\begin{aligned} c^f &= \text{the fixed cost of placing an order}, \\ c^p &= \text{the unit purchase cost}, \\ c^h &= \text{the unit holding cost}. \end{aligned}$$

For our purposes, $C(S_t, x_t, \hat{D}_{t+1})$ could be any nonconvex function; this is a simple example of one. Since the cost function is nonconvex, it helps to order larger quantities at the same time.

Assume that we have a family of decision functions $X^\pi(R_t)$, $\pi \in \Pi$, for determining x_t. For example, we might use a decision rule such as

$$X^\pi(R_t|\theta) = \begin{cases} \theta^{max} - R_t & \text{if } R_t < \theta^{min}, \\ 0 & \text{if } R_t \geq \theta^{min}. \end{cases}$$

where $\theta = (\theta^{min}, \theta^{max})$ are specified parameters. In the language of sequential decisio problems, a decision rule such as $X^\pi(S_t)$ is known as a *policy* (literally, a rule for making decisions). We index policies by π, and denote the set of policies by Π. In this example, a combination $(\theta^{min}, \theta^{max})$ represents an instance of our order-up-to policy, and Θ would represent all the possible values of θ^{min} and θ^{max} (this would be the set of policies in this class).

Our goal is to solve

$$\min_{\theta \in \Theta} \mathbb{E} \left\{ \sum_{t=0}^{T} \gamma^t C(S_t, X^\pi(R_t|\theta), \hat{D}_{t+1}) \right\}.$$

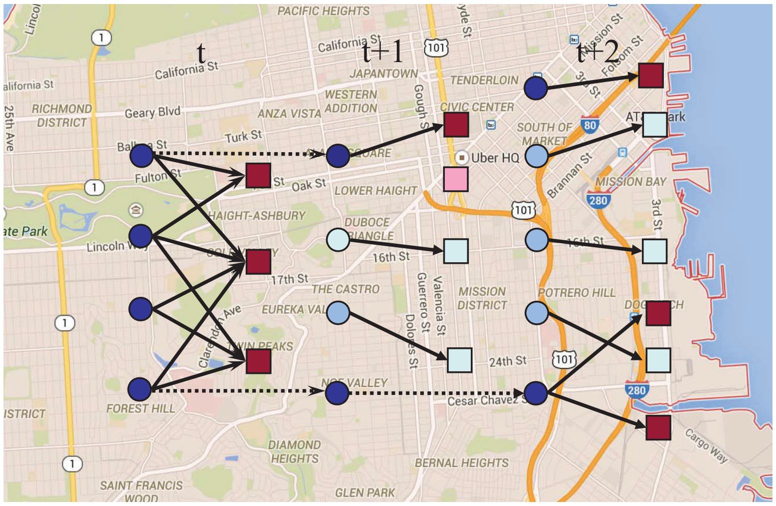


Figure 8.3 Illustration of the dynamic assignment of drivers (circles) to riders (squares).

This means that we want to search over all possible values of θ^{min} and θ^{max} to find the best performance (on average).

The basic batch replenishment problem, where R_t and x_t are scalars, is quite easy (if we know things like the distribution of demand). But there are many real problems where these are vectors because there are different types of resources. The vectors may be small (different types of fuel or blood) or extremely large (hiring different types of people for a consulting firm or the military; maintaining spare parts inventories).

8.3 COMPLEX RESOURCE ALLOCATION PROBLEMS

Problems involving the management of physical resources can become quite complex. Below we illustrate a dynamic assignment problem that arises in the context of assigning fleets of drivers (and cars) to riders requesting trips over time, and a problem involving the modeling inventories of different types of blood.

8.3.1 The dynamic assignment problem

Consider the challenge of matching drivers (or perhaps driverless electric vehicles) to customers calling in dynamically over time, illustrated in figure 8.3. We have to think about which driver to assign to which rider based on the characteristics of the driver (or car), such as where the driver lives (or how much energy is in the car's battery), along with the characteristics of the trip (origin, destination, length).

We describe drivers (and cars) using

$$a_t = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} \text{The location of the car} \\ \text{The type of car} \\ \text{Hours the driver has been on duty} \end{pmatrix}.$$

We can model our fleet of drivers and cars using

$$\begin{aligned}
\mathcal{A} &= \text{the set of all possible attribute vectors,}\\
R_{ta} &= \text{the number of cars with attribute } a \in \mathcal{A} \text{ at time } t,\\
R_t &= (R_{ta})_{a\in\mathcal{A}}.
\end{aligned}$$

We note that R_t can be very high dimensional, since the attribute a is a vector. In practice, we never generate the vector R_t, since it is more practical to just create a list of drivers and cars. The notation R_t is used just for modeling purposes.

Demands for trips arise over time, which we can model using

$$\begin{aligned}
b &= \text{the characteristics of a trip (origin, destination, car type requested),}\\
\mathcal{B} &= \text{the set of all possible values of the vector } b,\\
\hat{D}_{tb} &= \text{the number of new customer requests with attribute } b \text{ that were first learned at time } t,\\
\hat{D}_t &= (\hat{D}_{tb})_{b\in\mathcal{B}},\\
D_{tb} &= \text{the total number of unserved trips with attribute } b \text{ waiting at time } t,\\
D_t &= (D_{tb})_{b\in\mathcal{B}}.
\end{aligned}$$

We next have to model the decisions that we have to make. Assume that at any point in time, we can either assign a driver to handle a customer, or send her home. Let

$$\begin{aligned}
\mathcal{D}^H &= \text{the set of decisions representing sending a driver to her home location,}\\
\mathcal{D}^D &= \text{the set of decisions to assign a driver to a rider, where } d \in \mathcal{D}^D \text{ represents a decision to serve a demand of type } b_d,\\
d^\phi &= \text{the decision to ``do nothing,''}\\
\mathcal{D} &= \mathcal{D}^H \cup \mathcal{D}^D \cup d^\phi.
\end{aligned}$$

A decision has the effect of changing the attributes of a driver, as well as possibly satisfying a demand. The impact on the resource attribute vector of a driver is captured using the attribute transition function, represented using

$$a_{t+1} = a^M(a_t, d).$$

For algebraic purposes, it is useful to define the indicator function

$$\delta_{a'}(a_t, d) = \begin{cases} 1 & \text{for } a^M(a_t, d) = a', \\ 0 & \text{otherwise.} \end{cases}$$

A decision $d \in \mathcal{D}^D$ means that we are serving a customer described by an attribute vector b_d. This is only possible, of course, if $D_{tb} > 0$. Typically, D_{tb} will be 0 or 1, although our model allows for multiple trips with the same attributes. We indicate which decisions we have made using

$$\begin{aligned}
x_{tad} &= \text{the number of times we apply a decision of type } d \text{ to trip with attribute } a,\\
x_t &= (x_{tad})_{a\in\mathcal{A}, d\in\mathcal{D}}.
\end{aligned}$$

Similarly, we define the cost of a decision to be

$$\begin{aligned} c_{tad} &= \text{the cost of applying a decision of type } d \text{ to driver with attribute } a, \\ c_t &= (c_{tad})_{a\in\mathcal{A},d\in\mathcal{D}}. \end{aligned}$$

We could solve this problem myopically by making what appears to be the best decisions now, ignoring their impact on the future. We would do this by solving

$$\min_{x_t} \sum_{a\in\mathcal{A}} \sum_{d\in\mathcal{D}} c_{tad} x_{tad}, \tag{8.7}$$

subject to

$$\begin{aligned} \sum_{d\in\mathcal{D}} x_{tad} &= R_{ta}, & (8.8) \\ \sum_{a\in\mathcal{A}} x_{tad} &\leq D_{tb_d}, \quad d \in \mathcal{D}^D, & (8.9) \\ x_{tad} &\geq 0. & (8.10) \end{aligned}$$

Equation (8.8) says that we either have to send a driver home, or assign him to a job. Equation (8.9) says that we can only assign the driver to a job of type b_d if there is in fact a job of type b_d. Said differently, we cannot assign more than one driver per passenger. However, we do not have to cover every trip.

The problem posed by equations (8.7)-(8.10) is a linear program. Real problems may involve managing hundreds or even thousands of individual entities. The decision vector $x_t = (x_{tad})_{a\in\mathcal{A},d\in\mathcal{D}}$ may have over ten thousand dimensions (variables in the language of linear programming). However, commercial linear programming packages handle problems of this size quite easily.

If we make decisions by solving (8.7)-(8.10), we say that we are using a *myopic policy* since we are using only what we know now, and we are ignoring the impact of decisions now on the future. For example, we may decide to send a driver home rather than have him sit in a hotel room waiting for a job, but this ignores the likelihood that another job may suddenly arise close to the driver's current location.

Given a decision vector, the dynamics of our system can be described using

$$\begin{aligned} R_{t+1,a} &= \sum_{a'\in\mathcal{A}} \sum_{d\in\mathcal{D}} x_{ta'd} \delta_a(a', d), & (8.11) \\ D_{t+1,b_d} &= D_{t,b_d} - \sum_{a\in\mathcal{A}} x_{tad} + \hat{D}_{t+1,b_d}, \quad d \in \mathcal{D}^D. & (8.12) \end{aligned}$$

Equation (8.11) captures the effect of all decisions (including serving demands) on the attributes of the drivers. This is easiest to visualize if we assume that all tasks are completed within one time period. If this is not the case, then we simply have to augment the state vector to capture the attribute that we have partially completed a task. Equation (8.12) subtracts from the list of available demands any of type b_d that are served by a decision $d \in \mathcal{D}^D$ (recall that each element of $\mathcal{D}^D$ corresponds to a type of trip, which we denote b_d).

The state of our system is given by

$$S_t = (R_t, D_t).$$

	Recipient							
Donor	$AB+$	$AB-$	$A+$	$A-$	$B+$	$B-$	$O+$	$O-$
$AB+$	X							
$AB-$	X	X						
$A+$	X		X					
$A-$	X	X	X	X				
$B+$	X				X			
$B-$	X	X			X	X		
$O+$	X		X		X		X	
$O-$	X	X	X	X	X	X	X	X

Table 8.1 Allowable blood substitutions for most operations, 'X' means a substitution is allowed (from Cant (2006)).

The evolution of our state variable over time is determined by equations (8.11) and (8.12). We can now set up an optimality recursion to determine the decisions that minimize costs over time using

$$V_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(C_t(S_t, x_t) + \gamma \mathbb{E} V_{t+1}(S_{t+1}) \right),$$

where S_{t+1} is the state at time $t+1$ given that we are in state S_t and action x_t. S_{t+1} is random because at time t, we do not know $\hat{D}_{t+1}$. The feasible region $\mathcal{X}_t$ is defined by equations (8.8)-(8.10).

Needless to say, the state variable for this problem is quite large. The dimensionality of R_t is determined by the number of attributes of our driver, while the dimensionality of D_t is determined by the relevant attributes of a demand. In real applications, these attributes can become fairly detailed. Fortunately, this problem has a lot of structure which we exploit in chapter 18.

8.3.2 The blood management problem

The problem of managing blood inventories serves as a particularly elegant illustration of a resource allocation problem. We are going to start by assuming that we are managing inventories at a single hospital, where each week we have to decide which of our blood inventories should be used for the demands that need to be served in the upcoming week.

We have to start with a bit of background about blood. For the purposes of managing blood inventories, we care primarily about blood type and age. Although there is a vast range of differences in the blood of two individuals, for most purposes doctors focus on the eight major blood types: $A+$ (" A positive"), $A-$ ("A negative"), $B+$, $B-$, $AB+$, $AB-$, $O+$, and $O-$. While the ability to substitute different blood types can depend on the nature of the operation, for most purposes blood can be substituted according to table 8.1.

A second important characteristic of blood is its age. The storage of blood is limited to six weeks, after which it has to be discarded. Hospitals need to anticipate if they think they can use blood before it hits this limit, as it can be transferred to blood centers which monitor inventories at different hospitals within a region. It helps if a hospital can identify blood it will not need as soon as possible so that the blood can be transferred to locations that are running short.

One mechanism for extending the shelf-life of blood is to freeze it. Frozen blood can be stored up to 10 years, but it takes at least an hour to thaw, limiting its use in emergency situations or operations where the amount of blood needed is highly uncertain. In addition, once frozen blood is thawed it must be used within 24 hours.

We can model the blood problem as a heterogeneous resource allocation problem. We are going to start with a fairly basic model which can be easily extended with almost no notational changes. We begin by describing the attributes of a unit of stored blood using

$$\begin{aligned} a &= \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} \text{Blood type } (A+, A-, \dots) \\ \text{Age (in weeks)} \end{pmatrix}, \\ \mathcal{B} &= \text{Set of all attribute types.} \end{aligned}$$

We will limit the age to the range $0 \leq a_2 \leq 6$. Blood with $a_2 = 6$ (which means blood that is already six weeks old) is no longer usable. We assume that decision epochs are made in one-week increments.

Blood inventories, and blood donations, are represented using

$$\begin{aligned} R_{ta} &= \text{units of blood of type } a \text{ available to be assigned or held at time } t, \\ R_t &= (R_{ta})_{a \in \mathcal{A}}, \\ \hat{R}_{ta} &= \text{number of new units of blood of type } a \text{ donated between } t-1 \text{ and } t, \\ \hat{R}_t &= (\hat{R}_{ta})_{a \in \mathcal{A}}. \end{aligned}$$

The attributes of demand for blood are given by

$$\begin{aligned} d &= \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} \text{Blood type of patient} \\ \text{Surgery type: urgent or elective} \\ \text{Is substitution allowed?} \end{pmatrix}, \\ d^{\phi} &= \text{decision to hold blood in inventory (``do nothing''),} \\ \mathcal{D} &= \text{set of all demand types } d \text{ plus } d^{\phi}. \end{aligned}$$

The attribute d_3 captures the fact that there are some operations where a doctor will not allow any substitution. One example is childbirth, since infants may not be able to handle a different blood type, even if it is an allowable substitute. For our basic model, we do not allow unserved demand in one week to be held to a later week. As a result, we need only model new demands, which we accomplish with

$$\begin{aligned} \hat{D}_{td} &= \text{units of demand with attribute } d \text{ that arose between } t-1 \text{ and } t, \\ \hat{D}_t &= (\hat{D}_{td})_{d \in \mathcal{D}}. \end{aligned}$$

We act on blood resources with decisions given by

$$\begin{aligned} x_{tad} &= \text{number of units of blood with attribute } a \text{ that we assign to a demand} \\ &\quad \text{of type } d, \\ x_t &= (x_{tad})_{a \in \mathcal{A}, d \in \mathcal{D}}. \end{aligned}$$

The feasible region $\mathcal{X}_t$ is defined by the following constraints:

$$
\begin{aligned}
\sum_{d \in \mathcal{D}} x_{tad} &= R_{ta}, && (8.13)\\
\sum_{a \in \mathcal{A}} x_{tad} &\leq \hat{D}_{td}, \quad d \in \mathcal{D}, && (8.14)\\
x_{tad} &\geq 0. && (8.15)
\end{aligned}
$$

Blood that is held simply ages one week, but we limit the age to six weeks. Blood that is assigned to satisfy a demand can be modeled as being moved to a blood-type sink, denoted, perhaps, using $a_{t,1} = \phi$ (the null blood type). The blood attribute transition function $a^M(a_t, d_t)$ is given by

$$
a_{t+1} = \begin{pmatrix} a_{t+1,1} \\ a_{t+1,2} \end{pmatrix} = \begin{cases} \begin{pmatrix} a_{t,1} \\ \min\{6, a_{t,2}+1\} \end{pmatrix}, & d_t = d^{\phi}, \\ \begin{pmatrix} \phi \\ - \end{pmatrix}, & d_t \in \mathcal{D}. \end{cases}
$$

To represent the transition function, it is useful to define

$$
\begin{aligned}
\delta_{a'}(a,d) &= \begin{cases} 1 & a_t^x = a' = a^M(a_t, d_t), \\ 0 & \text{otherwise}, \end{cases}\\
\Delta &= \text{Matrix with } \delta_{a'}(a,d) \text{ in row } a' \text{ and column } (a,d).
\end{aligned}
$$

We note that the attribute transition function is deterministic. A random element would arise, for example, if inspections of the blood resulted in blood that was less than six weeks old being judged to have expired. The resource transition function can now be written

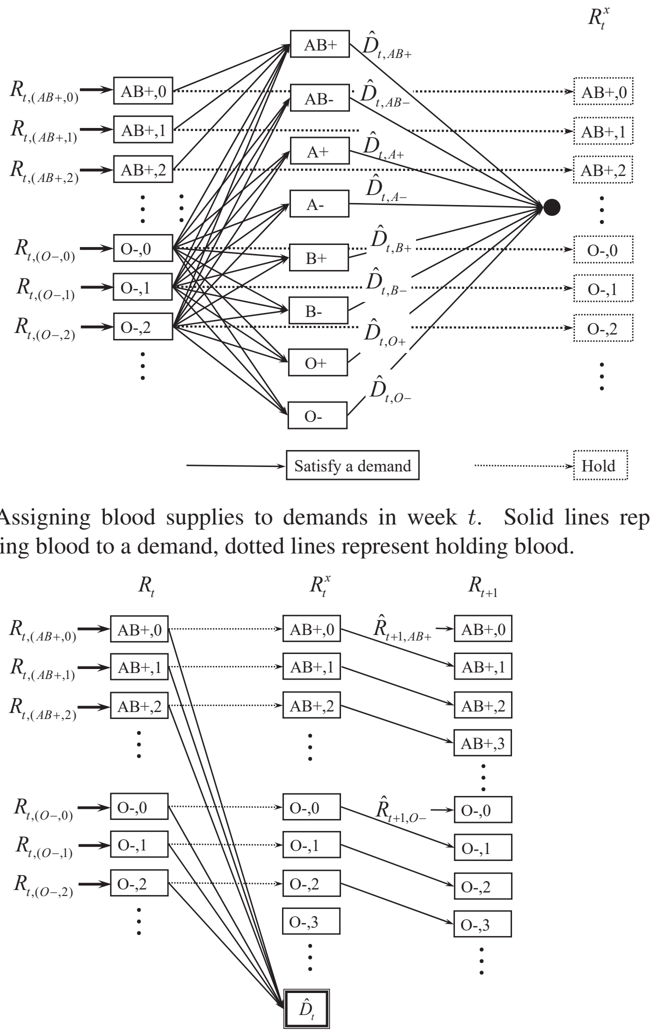
$$
\begin{aligned}
R_{ta'}^x &= \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} \delta_{a'}(a,d) x_{tad},\\
R_{t+1,a'} &= R_{ta'}^x + \hat{R}_{t+1,a'}.
\end{aligned}
$$

In matrix form, these would be written

$$
\begin{aligned}
R_t^x &= \Delta x_t, && (8.16)\\
R_{t+1} &= R_t^x + \hat{R}_{t+1}. && (8.17)
\end{aligned}
$$

Figure 8.4 illustrates the transitions that are occurring in week t. We either have to decide which type of blood to use to satisfy a demand (figure 8.4a), or to hold the blood until the following week. If we use blood to satisfy a demand, it is assumed lost from the system. If we hold the blood until the following week, it is transformed into blood that is one week older. Blood that is six weeks old may not be used to satisfy any demands, so we can view the bucket of blood that is six weeks old as a sink for unusable blood (the value of this blood would be zero). Note that blood donations are assumed to arrive with an age of 0. The pre- and post-decision state variables are given by

$$
\begin{aligned}
S_t &= (R_t, \hat{D}_t),\\
S_t^x &= (R_t^x).
\end{aligned}
$$



(a) - Assigning blood supplies to demands in week t. Solid lines represent assigning blood to a demand, dotted lines represent holding blood.

(b) - Holding blood supplies until week $t + 1$.

Figure 8.4 (a) The assignment of different blood types (and ages) to known demands in week t, and (b) holding blood until the following week.

There is no real "cost" to assigning blood of one type to demand of another type (we are not considering steps such as spending money to encourage additional donations, or transporting inventories from one hospital to another). Instead, we use the contribution function to capture the preferences of the doctor. We would like to capture the natural preference that it is generally better not to substitute, and that satisfying an urgent demand is more important than an elective demand. For example, we might use the contributions described in table 8.2. Thus, if we use $O-$ blood to satisfy the needs for an elective patient with $A+$ blood, we would pick up a -\$10 contribution (penalty since it is negative) for substituting blood, a +\$5 for using $O-$ blood (something the hospitals like to encourage), and a +\$20 contribution for serving an elective demand, for a total contribution of +\$15.

Condition	Description	Value
if $d = d^\phi$	Holding	0
if $a_1 = a_1$ when $d \in \mathcal{D}$	No substitution	0
if $a_1 \neq a_1$ when $d \in \mathcal{D}$	Substitution	-10
if $a_1 =$ O- when $d \in \mathcal{D}$	O- substitution	5
if $d_2 =$ Urgent	Filling urgent demand	40
if $d_2 =$ Elective	Filling elective demand	20

Table 8.2 Contributions for different types of blood and decisions

The total contribution (at time t) is finally given by

$$C_t(S_t, x_t) = \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} c_{tad} x_{tad}.$$

As before, let $X_t^\pi(S_t)$ be a policy (some sort of decision rule) that determines $x_t \in \mathcal{X}_t$ given S_t. We wish to find the best policy by solving

$$\max_{\pi \in \Pi} \mathbb{E} \sum_{t=0}^{T} C(S_t, X^\pi(S_t)). \tag{8.18}$$

The most obvious way to solve this problem is with a simple myopic policy, where we maximize the contribution at each point in time without regard to the effect of our decisions on the future. We can obtain a family of myopic policies by adjusting the one-period contributions. For example, our bonus of $5 for using $O-$ blood (in table 8.2), is actually a type of myopic policy. We encourage using $O-$ blood since it is generally more available than other blood types. By changing this bonus, we obtain different types of myopic policies that we can represent by the set Π^M, where for $\pi \in \Pi^M$ our decision function would be given by

$$X_t^\pi(S_t) = \arg\max_{x_t \in \mathcal{X}_t} \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} c_{tad} x_{tad}. \tag{8.19}$$

The optimization problem in (8.19) is a simple linear program (known as a “transportation problem”). Solving the optimization problem given by equation (8.18) for the set $\pi \in \Pi^M$ means searching over different values of the bonus for using $O-$ blood.

In chapter 13 we will introduce a way of improving this policy through simple parameterizations, using a class of policy we call a *cost function approximation*. We then revisit the same problem in chapter 18 when we develop a powerful strategy based on approximate dynamic programming, where we exploit the natural concavity of the value function. Finally, we touch on this problem one last time in chapter 20 when we show how to optimize the management of blood over many hospitals using a multiagent formulation.

8.4 STATE-DEPENDENT LEARNING PROBLEMS

Information acquisition is an important problem in many applications where we face uncertainty about the value of an action, but the only way to obtain better estimates of the value is to take the action. For example, a baseball manager may not know how well a particular player will perform at the plate. The only way to find out is to put him in the lineup and let him hit. The only way a mutual fund can learn how well a manager will perform may be to let her manage a portion of the portfolio. A pharmaceutical company does not know how the market will respond to a particular pricing strategy. The only way to learn is to offer the drug at different prices in test markets.

We have already seen information acquisition problems in chapter 7 where we called them active learning problems. Here we are going to pick up this thread but in the context of more complex problems that combine a hybrid of physical and informational states, in addition to belief states (which is what makes them learning problems).

Information acquisition plays a particularly important role in sequential decison problems, but the presence of physical states can complicate the learning process. Imagine that we are managing a medical team that is currently in zone i testing for the presence of a communicable disease. We are thinking about moving the team to zone j to do more testing there. We already have estimates of the presence of the disease in different zones, but visiting zone j not only improves our estimate for zone j, but also other zones through correlated beliefs.

Information acquisition problems are examples of dynamic optimization problems with belief states. These have not received much attention in the research literature, but we suspect that they arise in many practical applications that combine decisions under uncertainty with field observations.

8.4.1 Medical decision making

Patients arrive at a doctor's office for treatment. They begin by providing a medical history, which we capture as a set of attributes $a_1, a_2, \ldots$ which includes patient characteristics (gender, age, weight), habits (smoking, diet, exercise patterns), results from a blood test, and medical history (e.g. prior diseases). Finally, the patient may have some health issue (fever, knee pain, elevated blood sugar, ...) that is the reason for the visit. This attribute vector can have hundreds of elements.

Assume that our patient is dealing with elevated blood sugar. The doctor might prescribe lifestyle changes (diet and exercise), or a form of medication (along with the dosage), where we can represent the choice as $d \in \mathcal{D}$. Let t index visits to the doctor, and let

$$x_{td} = \begin{cases} 1 & \text{if the physician chooses medication } d \in \mathcal{D}, \\ 0 & \text{otherwise.} \end{cases}$$

After the physician makes a decision x_t, we observe the change in blood sugar levels by $\hat{y}_{t+1}$ which we assume is learned at the next visit.

Let $U(a, x|\theta)$ be a linear model of patient attributes and medical decisions which we write using

$$U(a, x|\theta) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(a, x),$$

where $\phi_f(a, x)$ for $f \in \mathcal{F}$ represents features that we design from a combination of the patient attributes a (which are given) and the medical decision x. We believe that we can predict the patient response $\hat{y}$ using the logistic function

$$\hat{y}|\theta \sim \frac{e^{U(a,x|\theta)}}{1 + e^{U(a,x|\theta)}}. \tag{8.20}$$

Of course, we do not know what θ is. We can use data across a wide range of patients to get a population estimate $\bar{\theta}^{pop}_t$ that is updated every time we treat a patient and then observe an outcome. A challenge with medical decision-making is that every patient responds to treatments differently. Ideally, we would like to estimate $\bar{\theta}_{ta}$ that depends on the attribute a of an individual patient.

This is a classical learning problem similar to the derivative-free problems we saw in chapter 7, with one difference: we are given the attribute a of a patient, after which we make a decision. Our state, then, consists of our estimate of $\bar{\theta}^{pop}_t$ (or $\bar{\theta}_{ta}$), which goes into our belief state, along with the patient attribute a_t, which affects the response function itself. The attributes a_t represents the dynamic information for the problem.

8.4.2 Laboratory experimentation

We may be trying to design a new material. For the n^{th} experiment we have to choose

$$\begin{aligned} x^n_1 &= \text{the temperature at which the experiment will be run,} \\ x^n_2 &= \text{the length of time that the material will be heated,} \\ x^n_3 &= \text{the concentration of oxygen in the chamber,} \\ x^n_4 &= \text{the water concentration.} \end{aligned}$$

When the experiment is complete, we then test the strength of the resulting material, which we model as

$$W^{n+1} = \text{the strength of the material produced by the experiment.}$$

We use this observation to update our estimate of the relationship between the inputs x^n and the resulting strength W^{n+1} which we represent using

$$f(x|\bar{\theta}^n) = \text{statistical estimate of the strength of the material produced by inputs } x^n.$$

Our belief state B^n would consist of the estimate $\bar{\theta}^n$ as well as any other information needed to perform recursive updating (you have o pick your favorite method from chapter 3). The transition function for B^n would be the recursive updating equations for $\bar{\theta}^n$ from the method you chose from chapter 3.

For our laboratory experiment, we often have physical state variables R^n that might capture inventories of the materials needed for an experiment, and we might capture in our information state I^n information such as the temperature and humidity of the room where the experiment is being run. This gives us a state variable that consists of our physical state R^n, additional information I^n, and our belief B^n.

8.4.3 Bidding for ad-clicks

Companies looking to advertise their product on the internet need to first choose a series of keywords, such as "hotels in New York," "pet friendly hotel," "luxury hotel in New York," that will attract the people who they feel will find their offering most attractive. Then, they have to decide what to bid to have their ad featured on sites such as Google and Facebook. Assume that the probability that we win one of the limited "sponsored" slots on either platform, when we bid p, is given by

$$P^{click}(p|\theta) = \frac{e^{\theta_0+\theta_1 p}}{1+e^{\theta_0+\theta_1 p}}. \tag{8.21}$$

Our problem is that we do not know $\theta = (\theta_0, \theta_1)$, but we think it is one of the family $\Theta = \{\theta_1, \ldots, \theta_K\}$.

Assume that after n trials, we have a belief $p^n_k = Prob[\theta = \theta_k]$ that θ_k is the true value of θ. Now, we have a budget R^n that starts at R^0 at the beginning of each week. We have to learn how to place our bids, given our budget constraint, so that we learn how to maximize ad-clicks. Our state variable consists of our remaining budget R^n and our belief vector $p^n = (p^n_1, \ldots, p^n_K)$.

8.4.4 An information-collecting shortest path problem

Assume that we have to choose a path through a network, but we do not know the actual travel time on any of the links of the network. In fact, we do not even know the mean or variance (we might be willing to assume that the probability distribution is normal).

To get a sense of some of the complexity when learning while moving around a graph, imagine that you have to find the best path from your apartment in New York City to your new job. You start by having to decide whether to walk to a subway station, take the subway to a station close to your workplace, and then walk. Or you can walk to a major thoroughfare and wait for a taxi, or you can make the decision to call Uber or Lyft if the wait seems long. Finally, you can call an Uber or Lyft from your apartment, which involves waiting for the car at your apartment and takes you right to your office.

Each decision involves collecting information by making a decision and observing the time required for each leg of the trip. Collecting information requires participating in the process, and changes your location. Also, observing the wait for an Uber or Lyft at your apartment hints at how long you might have to wait if you call one while waiting for a taxi. Your location on the graph (along with other information that might be available, such as weather) represents the dynamic information. In contrast to the medical decision-making example where we have no control over the patient attributes a_t, in our dynamic shortest path problem, our current location is directly a result of decisions we have made in the past.

Information-collecting shortest path problems arise in any information collection problem where the decision now affects not only the information you collect, but also the decisions you can make in the future. While we can solve basic bandit problems optimally, this broader problem class remains unsolved.

8.5 A SEQUENCE OF PROBLEM CLASSES

Eventually, we are going to show that most stochastic optimization problems can be formulated using a common framework. However, this seems to suggest that all stochastic optimization problems are the same, which is hardly the case. It helps to identify major problem classes.

- Deterministically solvable problems - These are optimization problems where the uncertainty has enough structure that we can solve the problem exactly using deterministic methods. This covers an important class of problems, but we are going to group these together for now. All remaining problem classes require some form of adaptive learning.

- Pure learning problems - We make a decision x^n (or x_t), then observe new information W^{n+1} (or W_{t+1}), after which we update our knowledge to make a new decision. In pure learning problems, the only information passed from iteration n to $n+1$ (or from time t to time $t+1$) is updated knowledge, while in other problems, there may be a physical state (such as inventory) linking decisions.

- Stochastic problems with a physical state - Here we are managing resources, which arise in a vast range of problems where the resource might be people, equipment, or inventory of different products. Resources might also be money or different types of financial assets. There are a wide range of physical state problems depending on the nature of the setting. Some major problem classes include

  Stopping problems - The state is 1 (process continues) or 0 (process has been stopped). This arises in asset selling, where 1 means we are still holding the asset, and 0 means it has been sold.

  Inventory problems - We hold a quantity of resource to meet demands, where leftover inventory is held to the next period.

  Inventory problems with static attributes - A static attribute might reflect the type of equipment or resource which does not change.

  Inventory problems with dynamic attributes - A dynamic attribute might be spatial location, age or deterioration.

  Multiattribute resource allocation - Resources might have static and dynamic attributes, and may be re-used over time (such as people or equipment).

  Discrete resource allocation - This includes dynamic transportation problems, vehicle routing problems, and dynamic assignment problems.

- Physical state problems with an exogenous information state - While managing resources, we may also have access to exogenous information such as prices, weather, past history, or information about the climate or economy. Information states come in three flavors:

  - Memoryless - The information I_t at time t does not depend on past history, and is “forgotten” after a decision is made.
  - First-order exogenous process - I_t depends on I_{t-1}, but not on previous decisions.
  - State-dependent exogenous process - I_t depends on S_{t-1} and possibly x_{t-1}.

- Physical state with a belief state - Here, we are both managing resources while learning at the same time.

This list provides a sequence of problems of increasing complexity. However, each problem class can be approached with any of the four classes of policies.

8.6 BIBLIOGRAPHIC NOTES

All of the problems in this chapter are popular topics in the operations research literature. Most of the work in this chapter is based on work with former students.

Section 8.1.1 - The stochastic shortest path problem is a classic problem in operations research (see, for example, Bertsekas et al. (1991)). We use it to illustrate modeling strategy when we make different assumptions about what a traveler sees while traversing the network.

Section 8.1.2 - The "nomadic trucker" problem was first introduced in Powell (2011).

Section 8.1.3 - Equipment replacement problems are a popular topic. This section was based on the work of Johannes Enders (Enders et al. (2010).

Section 8.2.4 - Batch replenishment problems are a popular topic in operations research, often arising in the context of bulk service queues. This section was based on the work of Katerina Padaki (Powell & Papadaki (2002)), but see also Puterman (2005*b*).

Section 8.3.1 - The material on the dynamic assignment problem is based on the work of Michael Spivey (Spivey & Powell (2004)).

Section 8.3.2 - This model of the blood management problem is based on the undergraduate senior thesis research of Lindey Cant (Cant (2006)).

Section 8.4.4 - The work on the information collecting shortest path problem is based on the work of Ilya Ryzhov (Ryzhov & Powell (2011)).

Section 8.4.3 - This section is based on the work of Weidong Han (Han & Powell (2020)).

EXERCISES

Review questions

8.1 What is meant by a "state-dependent" problem? Give three examples.

8.2 You are moving over a static graph. At each time period, you arrive at another node. Why is this a "state-dependent problem"?

8.3 What are the essential differences between a shortest path problem with random costs, and an inventory problem with random demands (and deterministic prices and costs)?

8.4 Consider a discrete inventory problem where you can order at most 10 items at a point in time, but where you can order them to arrive in 1 day, 2 days, ... 5 days. Give the state variable, and compute how many states we may have for this (very simple) problem.

8.5 For the dynamic assignment problem in section 8.3.1, assume that space has been divided into 200 zones, there are three types of cars, and drivers may be on duty up to 10 hours (treat hours as an integer starting at 1). To understand the complexity of the problem, answer the following:

a) What is the dimensionality of the state variable?

b) What is the dimensionality of the decision vector?

c) What is the dimensionality of the exogenous information vector?

8.6 For the blood management problem in section 8.3.2, answer the following:

a) What is the dimensionality of the state variable?

b) What is the dimensionality of the decision vector?

c) What is the dimensionality of the exogenous information vector?

Modeling questions

8.7 Consider a discrete inventory problem with deterministic demands, but (possibly) random costs. Starting with an inventory of 0, sketch a few time periods of this problem assuming you cannot order more than 2 items per time period, and show that this can be modeled as a dynamic shortest path problem.

8.8 What is the distinguishing characteristic of a state-dependent *problem*, as opposed to the state-independent problems we considered in chapters 5 and 7? Contrast what we mean by a solution to a stochastic optimization problem with a state-independent function, versus what we mean by a solution to a stochastic optimization problem with a state-dependent function?

8.9 Describe the gambling problem in section 14.4.1 as a decision tree, assuming that we can gamble only 0, 1 or 2 dollars in each round (this is just to keep the decision tree from growing too large).

8.10 Repeat the gambling problem assuming that the value of ending up with S^N dollars is $\sqrt{S^N}$.

8.11 Section 8.2.1 describes an inventory problem that uses a contribution function $C(S_t, x_t, W_{t+1})$, and shows that it can also be modeled so the single-period contribution is written $C(S_t, x_t)$. Show how to convert any problem where you are given the contribution function in the form of $C(S_t, x_t, W_{t+1})$ into a problem where the single period contribution is given by $C(S_t, x_t)$ without changing sum of the contributions over time.

Equation (8.5) in Show how to model the problem so that the contribution function at time t can be written as $C(S_t, x_t)$, which is to say, it does not depend on any information arriving between t and $t+1$. Your new formulation has to be equivalent to the old one.

8.12 Rewrite the transition function for the asset acquisition problem II (section 8.2.2) assuming that R_t is the resources on hand after we satisfy the demands.

8.13 Write out the transition equations for the lagged asset acquisition problem in section 8.2.3 when we allow spot purchases, which means that we may have $x_{tt} > 0$. x_{tt} refers to purchases that are made at time t which can be used to serve unsatisfied demands D_t that occur during time interval t.

Theory questions

8.14 Consider three variations of a shortest path problem:

Case I - All costs are known in advance. Here, we assume that we have a real-time network tracking system that allows us to see the cost on each link of the network before we start our trip. We also assume that the costs do not change during the time from which we start the trip to when we arrive at the link.

Case II - Costs are learned as the trip progresses. In this case, we assume that we see the actual link costs for links out of node i when we arrive at node i.

Case III - Costs are learned after the fact. In this setting, we only learn the cost on each link after the trip is finished.

Let v_i^I be the expected cost to get from node i to the destination for Case I. Similarly, let v_i^{II} and v_i^{III} be the expected costs for cases II and III. Show that $v_i^I \leq v_i^{II} \leq v_i^{III}$.

Problem solving questions

8.15 We are now going to do a budgeting problem where the reward function does not have any particular properties. It may have jumps, as well as being a mixture of convex and concave functions. But this time we will assume that $R = 30$ dollars and that the allocations x_t must be in integers between 0 and 30. Assume that we have $T = 5$ products, with a contribution function $C_t(x_t) = cf(x_t)$ where $c = (c_1, \ldots, c_5) = (3, 1, 4, 2, 5)$ and where $f(x)$ is given by

$$f(x) = \begin{cases} 0, & x \leq 5, \\ 5, & x = 6, \\ 7, & x = 7, \\ 10, & x = 8, \\ 12, & x \geq 9. \end{cases}$$

Find the optimal allocation of resources over the five products.

8.16 You suddenly realize towards the end of the semester that you have three courses that have assigned a term project instead of a final exam. You quickly estimate how much each one will take to get 100 points (equivalent to an A+) on the project. You then guess that if you invest t hours in a project, which you estimated would need T hours to get 100 points, then for $t < T$ your score will be

$$R = 100\sqrt{t/T}.$$

That is, there are declining marginal returns to putting more work into a project. So, if a project is projected to take 40 hours and you only invest 10, you estimate that your score will be 50 points (100 times the square root of 10 over 40). You decide that you cannot spend more than a total of 30 hours on the projects, and you want to choose a value of t for each project that is a multiple of 5 hours. You also feel that you need to spend at least 5 hours on each project (that is, you cannot completely ignore a project). The time you estimate to get full score on each of the three projects is given by

Project	Completion time T
1	20
2	15
3	10

You decide to solve the problem as a dynamic program.

(a) What is the state variable and decision epoch for this problem?

(b) What is your reward function?

(c) Write out the problem as an optimization problem.

(d) Set up the optimality equations.

(e) Solve the optimality equations to find the right time investment strategy.

Diary problem

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

8.17 Describe the elements of your diary problem following the framework provided in section 2.2. Then describe the types of rules you might use to make a decision.