Chapter 19

Competitive and Intelligent Foraging

Chapter Contents

Foraging sometimes involves simply going out and finding nutrients, but often a key aspect of foraging is competing for resources with other foragers. This can involve trying to find resources before another forager. In other cases, it could involve issues in fighting other animals for a resource, or perhaps one forager is the nutrient source for another forager and then there may be issues of pursuit and evasion. Clearly, as for the case of noncompetitive foraging, many aspects of the environment affect competitive foraging behavior and as always, evolution plays a fundamental role in foraging strategy design.

In this chapter, characteristics of competitive and cooperative foraging in nature are modeled using a game-theoretic perspective. The basic definitions and rules in game theory are introduced by showing how to set up and solve static finite two-player games (matrix games) for security and saddle point strategies. Then, it is shown how to define and solve bimatrix and infinite competitive (adversarial) games for Nash equilibria, minimax solutions, and Stackelberg solutions. Pareto-optimal solutions for cooperative games are discussed to help clarify the relationships between competitive and cooperative games. We apply the methods to the study of adversarial foraging games that involve resource competition, and then resource allocation when there is cooperation among players and hence, social foraging. Next, we define a model for a dynamic game, and discuss aspects of strategies and the information space.

This chapter complements the last one. We treat cooperative foraging here, but only to show another way to view social foraging, and with the intent of contrasting it with competitive foraging. This chapter is also important in that each forager views all the other foragers as part of its environment (plant) and tries to take actions (generate control inputs) in order to succeed. The other foragers are not simply treated as uncertainty (a disturbance). Certain aspects of the other foragers are modeled (e.g., the assumption that they are rational and hence, will try to maximize their own returns).

In the last section of this chapter, we introduce "intelligent" foraging which is one example of how other aspects of intelligence affect foraging (e.g., planning, attention, and learning). The last section is meant to challenge you to think about how to integrate all the methods of this book, and indeed, its main objective is to provide a "challenge problem" that you can help define and then solve (e.g., as a final project in a class).

19.1 Competition and Fighting in Nature

Competition encountered in an environment affects organism evolution.

Competition for resources that are critical for survival affects many aspects of animal behavior and evolution. Some animals use poisoning for self-defense. Some evolve armor or grow spines. Some fight other animals for food resources or mates. Others simply seek to eliminate their competitors. Some recruit predators of their predators to obtain a defense. Some guard a resource-rich territory. Complex behaviors of groups of animals have evolved for self-defense. For instance, when a predator approaches a prey group, the normal response of the group of prey is to tighten and this can have an effect of maximizing mes-

sage transmission speed by increasing an ability to detect low strength signals. Essentially, the size of the group of organisms can change based on actions of predators, so that a group-level distributed yet coordinated defensive action can be taken. There is a vast diversity of behaviors that have evolved to ensure that animals get the resources they need. Here, we will model and analyze some of these in a game-theoretic framework and hence, will refer to them as "foraging games."

19.1.1 Foraging Games

Limitations in availability of resources naturally leads to competition in foraging.

Here, we will model certain aspects of both competitive and cooperative behavior via a game theoretic approach. We are not concerned here with the dynamics and small time-scale strategies of one-on-one "battles" between two animals (e.g., how a lion may capture, handle, kill, and eat an antelope). Our focus will be on foraging for resources via the study of "foraging games." What is a foraging game? It is a game where players (animals, humans) seek and consume (capture, occupy) "resources" (e.g., food, prey, territory, shelter, mates). In a foraging game, the "players" from classical game theory are referred to as "foragers." The foragers in such games are typically mobile and decide "where to go and what to do" (e.g., go to a certain location and consume a certain resource type). There are many types of foraging games, which can arise in nature and we outline some characteristics of these as follows:

- *Environment and resource characteristics:* Different types of games arise depending on characteristics of the environment, resources, and foragers. Some resources may be higher priority than others and these priorities may depend on the forager and its current needs (e.g., its diet). Some resources are static in the sense that they do not move, while others may be able to move quickly. Other resources are only available for a limited amount of time, and then they naturally dissipate. The foraging environment (where the forager forages) affects resource availability and likelihood of the location of resources.

Foragers may compete for food, and one forager may be food (prey) for the other.

- *Forager characteristics:* Forager capabilities significantly affect the characteristics of the game. For example, how fast a forager can move, how efficient it can move in different environments, its consumption capabilities, fighting capabilities, and cognitive capabilities (e.g., memory, learning, and reasoning) all affect its ability to forage successfully. Moreover, foraging games are sometimes dominated by whether there is competition for resources between foragers, or cooperation between multiple foragers to obtain resources ("social foraging"). Such cooperation requires communication or use of shared information in some way and hence, demands that a forager has certain capabilities.

- *Competitive foraging:* A competitive foraging game is one in which the foragers compete with each other in an adversarial relationship in order to obtain resources (e.g., if one forager gets the resource, it is no longer

available for the other forager). In the case where one forager is the predator and the other is its prey, we have a special foraging game that may be a "pursuit evasion game." We can then think of one forager as being a resource for another and one tries to capture or consume the other. It can also be the case that each forager is a resource for the other. Then, each tries to achieve a competitive advantage to capture and consume the other.

- *Cooperative foraging is social foraging:* A cooperative foraging game is one where foragers may share information to try to optimize resource acquisition for the group (or variance reduction in resource acquisition rates); the idea is that by working together, every forager does better. Cooperative foraging is the essence of "social foraging." A key concept in such cooperation is the "allocation" (distribution of a pattern) of activities that results in an allocation of resources to each group member (forager) to ensure that the *group* does as good as possible. This may involve, at times, sacrifices by one group member to increase the group's performance; however, over the long run it should be better, perhaps in an evolutionary sense, for the group to cooperate rather than compete. A group of cooperative foragers may compete with another group of cooperative foragers so that both elements of cooperative and competitive foraging are present. Indeed, there are often some elements of competition even in groups of cooperative foragers.

Cooperative and competitive foraging coexist in some foraging scenarios.

19.1.2 Intelligent Foragers

Attention, learning, and planning may help an individual forager increase energy intake per unit time spent foraging. Paying attention to the proper aspects of an environment can help the forager find nutrients and avoid predators. Learning helps you not to go back to places where you cannot find food (since it may not be likely that some food moved there). It helps the organism to remember where it has not yet gone for food (related to attention), where a past food source is, or what types of environmental "signs" typically indicate the presence of a good food source. Such learned information can be used by a planning system to further improve foraging performance. Planning involves reasoning over learned information, setting priorities, and optimizing choices locally (e.g., it allows the organism to actually perform an optimization of E/T over a short period of time, using less-than-certain learned information). Via learning and planning together, the forager can directly try to make locally optimal decisions, at least for the learned information.

"Intelligent" foraging involves the use of higher-level cognitive functions such as planning, attention, and learning.

Another relevant topic from planning theory, is the concept of "retrospective" and "prospective" coding, where it is thought that we recall where we have been and know where we have not been and use this to plan our activities. There is some evidence that when we begin planning, we use a retrospective encoding and then at some point, switch to a prospective encoding since then, we do not have to hold as much in short-term memory. Basically, early in the

process, we may be taking actions that help us to learn about the environment, while later, after we have learned a significant amount about the environment, it may be best to use the learned information to decide what actions to take, and to take actions that may not be directed towards learning more, but towards other objectives.

Some learning theorists have thought of learning as foraging for information that is then stored in "cognitive maps." For example, they may think of operant conditioning as foraging for reinforcement. Particularly relevant is the concept of "sign tracking" that has been used by learning theorists to explain foraging, and this approach motivates the use of smooth cost functions that represent where food is, and where risks are. The theory of "behavioral regulation" proposes that there are homeostatic mechanisms for behavior, where an organism tries to choose an optimal distribution of activities for survival. It is thought that if the balance of activities is upset, behavior is assumed to change to correct the deviation from the homeostatic level (this is the "behavioral bliss point approach").

19.1.3 Evolution and Foraging

Foraging strategies are fine-tuned by evolution since more successful foragers tend to have more offspring that possess aspects of their successful foraging strategies. In a sense, evolution seeks to perform a robust optimization of a forager's strategy in the face of the following constraints:

Foraging strategies evolve, and there is a complex dynamic interaction between environmental and forager changes.

- Environment and resource characteristics (e.g., a typical environment where the forager lives, along with a typical spread of the resources).
- Forager capabilities (e.g., motor and cognitive).

The environment and resources can change. Indeed, in a predator-prey situation, both the predator and the prey are evolving and hence, "coevolution" can occur, which in some cases can be thought of as a type of "arms race."

Anyone familiar with evolutionary optimization will quickly see how it can be used for robust foraging strategy design. Hence, we will not concern ourselves with that here. Another area of relevant study from theoretical biology is "evolutionary game theory." See the "For Further Study" section at the end of this part.

19.2 Introduction to Game Theory

In this section, we introduce basic definitions for concepts that form the foundation for the game-theoretic view of competition and cooperation, and show how to compute several types of player strategies.

19.2.1 Strategies and Information for Decisions

We start by defining a simple "matrix game" and explaining how it is played. This leads us to define strategies and a discussion on what information is used in making decisions.

Players, Rules, and Payoffs

Suppose that we have two "players" ("decision-makers") that we denote by P_1 and P_2. Let θ^1 and θ^2 denote the "decision variables" of the two players, respectively. Let J_1 and J_2 denote the cost functions of the players (i.e., what they gain or lose for a given set of decisions). The cost J_1 could represent, for example, a payment in cash from P_1 to P_2. In this case P_1 (P_2) wants to minimize (maximize) J_1. The problem will be, however, that P_1 (P_2) cannot unilaterally minimize (maximize) the cost. Each player's losses and gains are also influenced by the actions of the other player; that is the essence of a competitive game.

If

$$J_1(\theta^1, \theta^2) + J_2(\theta^1, \theta^2) = 0$$

for all θ^1 and θ^2, then we have a "zero sum game" so that gains of one player are losses of the other and they are in an adversarial relationship. If $J_1(\theta^1, \theta^2) + J_2(\theta^1, \theta^2) = c$ for some known constant c, then we can simply redefine the cost functions to incorporate the value of c to obtain a zero sum game.

To keep things simple, initially suppose that

$$\theta^1 \in \{1, 2, \ldots, D_1\}$$

and

$$\theta^2 \in \{1, 2, \ldots, D_2\}$$

so that there are only a finite number of D_1 decisions for P_1 and D_2 decisions for P_2. For simplicity, we will refer to the different decisions (which sometimes we will call "strategies") as $\theta^1 = i$ and $\theta^2 = j$ for P_1 and P_2, respectively. We then denote the costs by $J_1(i, j)$ and $J_2(i, j)$.

Here, we will often think of $J_1(i, j)$ as being a cash payoff (clearly, many other interpretations for cost are possible) of P_1 to P_2 given the decisions i and j by P_1 and P_2, respectively. In the zero sum case $J_1(i, j) = -J_2(i, j)$ (player 2 gets all the payoff of player 1 and vice versa) and if $J_1(i, j) \leq 0$, this represents that player 2 pays player 1 if P_1 uses a decision i and P_2 uses a decision j. In the two-player case, it is sometimes convenient to represent the payoff functions $J_1(i, j)$ and $J_2(i, j)$ as $D_1 \times D_2$ matrices, J_1^{ij} and J_2^{ij}, respectively (then $J_1(i, j) = J_1^{ij}$, an element of the matrix, for all i and j). This will be especially useful below when we consider "matrix games" where J_1^{ij} and J_2^{ij} will be called "payoff matrices."

Each player tries to maximize its own gains in a competitive game, possibly at the expense of the other player.

Suppose that the game is only played once (i.e., P_1 and P_2 only make decisions once). The players make decisions with full information about payoffs, they make their decisions simultaneously, and are "rational." Assuming that P_1 and P_2 are "rational" players, means that P_1 tries to minimize $J_1(i, j)$ and

P_2 tries to maximize it. Particular values of θ^1 and θ^2 are called "actions." A "strategy" is an established way of acting that depends on the possible actions of another player (e.g., if the player's decisions depended on gathered information about the decision process).

Generally, players are concerned with what strategies to use in playing a game. A pair of decision strategies for a two-player finite game is denoted by (i, j). The "outcome" of the game is J_1^{ij}. An "optimal" strategy, or simply one that we choose from a fixed set of possibilities, will be something that will be denoted by (i^*, j^*).

Security Strategies, Saddle Point Strategies, and Information

Let $D_1 = 5$ and $D_2 = 3$. Suppose that

$$J_1^{ij} = \begin{bmatrix} -3 & 4 & 4 \\ 0 & -5 & 2 \\ -2 & 1 & -4 \\ 2 & 3 & -4 \\ 2 & -2 & -5 \end{bmatrix} \tag{19.1}$$

Note that rows of this matrix correspond to P_1 decisions and columns correspond to P_2 decisions.

For a security strategy, a player chooses so as to minimize its maximum loss.

In a "security strategy," a player makes decisions to secure losses against whatever the other player might do (i.e., it minimizes its maximum possible loss). Hence, P_1 picks row i^* such that any value in any column of this row is no bigger than the largest value of any other row $i \neq i^*$ (i.e., pick the row that minimizes the maximum size column value). For low values of D_1 and D_2, it is possible to specify the solution by inspection of the matrix in Equation (19.1) above. For larger values of m or n, you may want to write a computer program to solve for the security strategy.

For Equation (19.1), the list of maximum values for each row is

$$\begin{matrix} 4 \\ 2 \\ 1 \\ 3 \\ 2 \end{matrix}$$

and so the security strategy is for P_1 to pick $i^* = 3$ to minimize what it has to pay to P_2. The "loss ceiling" (i.e., the most it can lose) is 1, which is less than the other possible losses, and this is called the "security level" of P_1. Similarly, P_2 can adopt a security strategy by choosing the column j^* whose row values are smaller than the smallest value found for another column $j \neq j^*$. In this case, the list of minimum values is

$$-3 \quad -5 \quad -5$$

so that the security strategy for P_2 is $j^* = 1$ and P_2 secures gains at the "gain floor" (its security level) of -3 (i.e., he pays no more than 3). Clearly the

security level of P_1 is never below the security level of P_2. The "outcome" of the game, which in this case is

$$J_1^{i^* j^*} = J_1^{31} = -2$$

will lie between the two security levels.

In the special case where the security levels of the two players are the same, the security strategies of the two players are in "equilibrium" with each other since they are "optimal" with respect to each other; in this case, they are called "saddle point strategies." A pair of strategies is said to be in a "saddle point equilibrium" if unilateral deviations by one player from its strategy will not benefit that player. In general, for a $D_1 \times D_2$ matrix game if (i^*, j^*) is the pair of chosen strategies, and if

$$J_1^{i^* j} \leq J_1^{i^* j^*} \leq J_1^{i j^*}$$

for all $i \in \{1, 2, \ldots, D_1\}$ and $j \in \{1, 2, \ldots, D_2\}$, then (i^*, j^*) constitutes a saddle point equilibrium. The value of $J_1^{i^* j^*}$ is called the "saddle point value." Is there a saddle point equilibrium for the above example matrix game?

Next, we discuss the issue of the quantity and type of information that is used by a player for its strategy, but only via our above simple example for security strategies. Note that if in the example above we changed the game so that P_1 plays first, then P_2, with P_2 *knowing* what P_1 had chosen, then the outcome will be different. We can actually deduce the outcome of the game by simple inspection of the matrix in Equation (19.1). First, note that it makes sense for P_1 to use a security strategy, since we assume that it does not know anything about the decision tendencies of P_2. This gives $i^* = 3$ just like above. If P_2 is informed of this choice, then it would pick $j^* = 2$ to get a gain of 1, which is more than what it got with a security strategy. This simple example shows that the best strategy to use depends on what information is available to the player, and when it is available. This is a fundamental principle that drives the design of strategies. Games of the type where players can use information from the process of the evolving game are called "dynamic" games, whereas the above example is called a "static game," since only a priori information is used to play it.

Strategies depend critically on what information is available to a player.

The development of strategies above assumes that each player is rational and that each player knows this about the other player. If we assume that a probability distribution is known by one player P_1 about the other player's decisions (i.e., that it knows the probability that it will make each decision), then we can design a so-called "mixed strategy" (as opposed to the cases above, which are sometimes called "pure strategies"), where the decisions each player makes are based on the outcome of random events (i.e., a derived probability distribution on P_1 decisions, given the information on P_2). For example, the design of the strategy of P_1 could involve choosing a probability distribution on its decisions so that it would be most likely to minimize what it pays to P_2.

The security strategy concept extends to having more than two players and then results in a multidimensional matrix game. For instance, we could imagine

adding a third player to the above game (which we could call "nature") that makes random decisions, and then develop security strategies for P_1 and P_2. The above example, and several others in this section, are "noncooperative games," where there is a clear adversarial relationship between the players (e.g., one's losses are the other's gains). There are also games where there is not a diametrically opposed relationship between the two players or where they are truly "cooperative" and try to help each other achieve their goals (so long as they also do well). Some such games are called "cooperative games" and typically, they involve sharing information so that each gains as much as possible. These will be discussed below when we discuss the Pareto-optimal solution.

Extensive Forms and Decision Trees

The matrix form of a game is called its "normal form." The normal form does not depict a complete representation of a game, for example, if there are repeated steps of play where players use different information from past player decisions. An "extensive form" of a game involves creating a type of labeled "decision tree" to represent the game. We discuss the extensive form via a simple example.

The game we consider is depicted in Figure 19.1. Figure 19.1(a) shows the extensive form representation for the matrix game in Equation (19.1), so consider it first. The game starts at the "top" of the (upside down) tree and evolves to the tip of one of its branches by a sequence of decisions starting with P_1 in "level 1" of decision-making and then P_2 at level 2. In level 1, P_1 chooses among its five alternatives, each represented with a separate labeled branch, and in level 2, P_2 chooses among its three alternatives, represented by the three labeled branches. The outcomes for the various decisions are shown at the tips of the branches. For example, in the security strategy case, we had $(i^*, j^*) = (3, 1)$ so we get an outcome of -2 as shown in Figure 19.1(a). The dashed line encirclement of the "decision nodes" at level 2 depicts the "information set" of P_2. It represents that in Figure 19.1(a), P_2 does not know which branch (decision) P_1 has chosen. This is then equivalent to the two players *simultaneously* making their decisions (how we interpreted the matrix game).

Figure 19.1(b) shows the case discussed in the last subsection where P_1 makes the decision first, then P_2 makes its decision, knowing the choice of P_1. The information sets in this case are again shown with the dashed encirclements of the decision nodes at level 2. As we noted above, due to the change in available information, there is a change in strategy and hence, outcome. The dashed encirclements clearly represent the inherent difference in the two games in Figure 19.1, whereas the normal form representation does not. This is one more reason why the extensive form is a convenient and intuitive representation for games, at least finite games without too many decision alternatives so that the trees are not too "bushy."

Finally, note that in general, decision trees may have many levels (multiple decisions), for example, with P_1 and P_2 taking turns so that at odd levels, P_1 would act and at even levels, P_2 would act. Moreover, it should be clear how to make an extensive form for the case where there are more than two players.

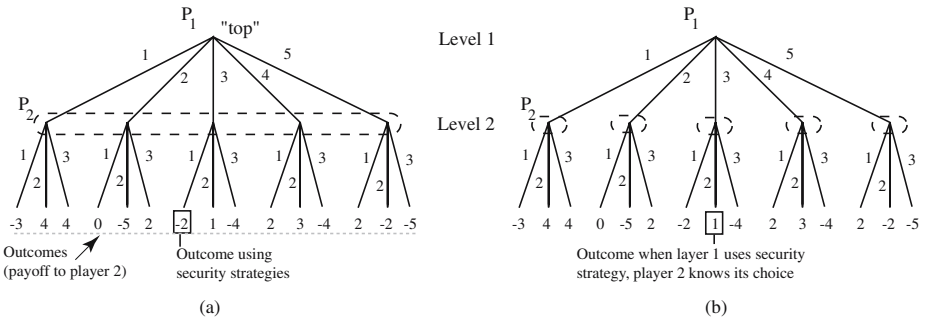

Figure 19.1: Example of an extensive representation, in this case in (a) for the matrix game in Equation (19.1) and in (b) for the same payoff matrix, but when P_1 chooses first and then P_2 knows its decision before choosing.

Decisions Versus Strategies

Figure 19.1 allows us to define the concept of a "strategy" relative to a "decision" more clearly. Suppose that we let $G^i(\cdot)$ denote the strategy for player i, where "$\cdot$" is the information that is available for decision-making. Note that the security strategies for P_1 and P_2 in Figure 19.1(a) are simply

$$G^1 = 3,\ G^2 = 1$$

(Note that there is no argument for the G^i functions, since there is no information from the decision-making *process*, in addition to available a priori information that is used to make the choices.)

The player strategies for the game in Figure 19.1(b) are $G^1 = 3$ (the security strategy), and the strategy of P_2 is what we might call "pick the best payoff given the decision of P_1." The representation for this strategy could be

$$G^2(i) = \begin{cases} 2 & \text{if } i = 1 \\ 3 & \text{if } i = 2 \\ 2 & \text{if } i = 3 \\ 2 & \text{if } i = 4 \\ 1 & \text{if } i = 5 \end{cases}$$

(Of course, we could use $G^2(1) = 3$ also, since it results in the same payoff to P_2.) Note that since there is information used from the decision-making process, particularly the decision of P_1 at level 1, the strategy is defined in terms of what decision i that P_1 might use.

The definition of a strategy is in general quite different from the meaning of a "decision" (action) as long as there is more than one "information set" (i.e., a dashed encirclement in Figure 19.1). The strategy is a mapping that specifies what action to take (decision to make) depending on what is known, as specified via an information set. If a player has an ability to distinguish something about a decision process as it evolves, then to keep the strategy of

the player independent of what the other player does, it must have different ways of reacting, depending on what the other player did. For example, this is perhaps clarified if we considered a strategy for P_1 that was simply based on a random choice for its options. In this case, P_2 would have to be ready to react no matter what P_1 did, and the above strategy $G^2(i)$ would define that.

Finally, note that if there are multiple levels, with many players, there are many options for how to define information sets, and hence, strategies for games. Finding strategies for the case where there are such dynamic games is, in general, a challenging problem.

19.2.2 Nash, Minimax, and Stackelberg Strategies

In this section, we introduce the Nash equilibrium strategy and provide an example of how to compute it. Moreover, we introduce infinite games (i.e., ones where there are an infinite number of strategies by at least one player), the concept of a reaction curve, and use an example to illustrate the basic ideas. This is followed by an introduction to minimax and Stackelberg strategy design.

Nash Equilibrium Strategies

Up till now, we had considered the zero sum case, and now we move beyond that to consider the nonzero sum case, that is, when it can be that

$$J_1(i,j) + J_2(i,j) \neq 0$$

Now, we have two payoff matrices J_1^{ij} and J_2^{ij} denoting losses of P_1 and P_2, respectively. Assume that both players are rational, so they try to minimize their losses. Unless otherwise stated, we assume that there is no cooperation and decisions are made independently.

The basic problem for each player is that the outcome resulting from their decision also depends on what the other player decides. So, what strategies should the players use? Recall that a pair of strategies is said to be in a "saddle point equilibrium" if unilateral deviations by one player from its strategy will not benefit that player. There are actually a variety of "equilibrium" solutions for a pair of strategies of a game.

If players use the Nash equilibrium solution, then they have no reason after playing the game to regret their decisions.

A strategy pair (i^*, j^*) is a noncooperative (Nash) equilibrium solution to a "bimatrix" game (J_1^{ij}, J_2^{ij}) if the inequalities

$$J_1^{i^*j^*} \leq J_1^{ij^*} \tag{19.2}$$

and

$$J_2^{i^*j^*} \leq J_2^{i^*j} \tag{19.3}$$

are both satisfied for all $i \in \{1, 2, \ldots, D_1\}$ and all $j \in \{1, 2, \ldots, D_2\}$. The pair $(J_1^{i^*j^*}, J_2^{i^*j^*})$ is the noncooperative (Nash) equilibrium outcome of the game.

For a given bimatrix game, there can be no Nash solutions, one Nash solution, or many Nash solutions. If $J_1^{ij} = -J_2^{ij}$ for all i and j, then we have a zero sum game, and a Nash solution is a saddle point equilibrium for the game.

As an example, let $D_1 = 5$ and $D_2 = 3$. Suppose that

$$J_1^{ij} = \begin{bmatrix} -1 & 5 & -3 \\ -2 & 5 & 1 \\ 4 & 3 & -2 \\ -5 & -1 & 5 \\ 3 & 0 & 2 \end{bmatrix}, \; J_2^{ij} = \begin{bmatrix} -1 & 2 & -3 \\ 2 & -3 & 1 \\ -2 & 3 & 1 \\ -1 & 1 & -1 \\ 4 & -4 & 1 \end{bmatrix} \tag{19.4}$$

To solve for the Nash equilibria, consider candidate (i^*, j^*) pairs in turn and test if they satisfy both the inequalities in Equations (19.2) and (19.3). To do this, consider $(1, 1)$ and see if J_1^{11} is less than all other row elements of column one to test Equation (19.2). Since it is not, $(1, 1)$ cannot be a Nash solution. If you test $(1, 2)$, you will also find it is not a Nash solution. However, if you test $(1, 3)$, you will see that $J_1^{13} \leq J_1^{i3}$ for all i so it is a candidate, so test the inequality in Equation (19.3) and you will find that $J_2^{13} \leq J_2^{1j}$ for all j; hence, $(i^*, j^*) = (1, 3)$ is indeed a Nash equilibrium. The Nash equilibrium outcome is $(-3, -3)$ so that both players *gain* 3. Show that $(4, 1)$ is also a Nash solution, with an outcome of $(-5, -1)$, but that all others are not.

Note that a Nash equilibrium solution is special since, if the players adopt it, then they have no reason after playing the game to regret their decisions. Note, however, that there can be more than one Nash equilibrium so the question of which one to use arises. However, it is not possible to totally order the Nash strategies according to the values of their outcomes, because they are defined by *pairs* of numbers. We can, however, say that "one Nash strategy is better" than another if *both* outcomes are better than the other. Then, we will call a Nash strategy "admissible" if there is no better Nash strategy. For the above example, there were two Nash solutions and each one is admissible since one is not better than the other. Note that in this case, if P_1 picks $(1, 3)$ and P_2 picks the other Nash solution (one thinks that the other is picking the other strategy to play by, since there is no reason to think that they would definitely pick the same Nash strategy to play without some type of cooperation), then the strategy pair that is employed is $(1, 1)$, which as the above example showed, is not a Nash solution. In fact, $(1, 1)$ results in an outcome of $(-1, -1)$, which is *worse* for both players. This creates a problem with implementing a Nash solution for a noncooperative game when the Nash solution is not unique.

If there is only one (admissible) Nash solution, this problem will not arise. However, if the two payoff matrices are the same, this problem can still arise. Why? There are then many cases where the situation can arise where there are multiple Nash equilibria, so that the players cannot use the solutions effectively without some type of cooperation. Essentially, this problem with the Nash solutions arises, since bimatrix games may not be "antagonistic," so that a noncooperative solution concept can be inappropriate (i.e., elements of cooperation can be reflected in the payoff matrices). We will revisit this issue when we discuss Pareto-optimal solutions below.

Infinite Games and Reaction Curves

An infinite game is one in which there are an infinite number of strategy choices by one or both of the players. The concept of Nash equilibria is also valid for this case, and we illustrate this via a simple example here.

First, suppose that the actions of P_1 can be

$$\theta^1 \in [-4, 4]$$

and for P_2 they can be

$$\theta^2 \in [-5, 5]$$

Suppose that cost functions for the two players are

$$J_1(\theta^1, \theta^2) = -\exp\left(-\frac{(\theta^1 - 2)^2}{8} - \frac{(\theta^2 - 4)^2}{2}\right)$$

and

$$J_2(\theta^1, \theta^2) = -\exp\left(-\frac{(\theta^1 - 1)^2}{1} - \frac{(\theta^2 + 1)^2}{6}\right)$$

each of which has one global minimum, and also for which, if you fix θ^1 (θ^2), there is a unique minimum point in the other value. (This will simplify the discussion.)

Define the "reaction curve" of P_1 to be

$$R_1(\theta^2) = \arg\min_{\theta_1} J_1(\theta^1, \theta^2)$$

where we are using the assumption of uniqueness of the minimum point, so that there is only one point at which the minimum is achieved. The reaction curve $R_1(\theta^2)$ defines how P_1 should react for every possible action of P_2 in order to minimize its losses. Similarly, the reaction curve of P_2 is

$$R_2(\theta^1) = \arg\min_{\theta_2} J_2(\theta^1, \theta^2)$$

and it defines how P_2 should react for every possible action of P_1 in order to minimize its losses. Contour plots of J_1 and J_2 are shown in Figure 19.2(a) for the above loss functions, along with the reaction curves $R_1(\theta^2)$ and $R_2(\theta^1)$.

It is interesting to note that any intersection point of the two curves in Figure 19.2(a) is a Nash equilibrium, so $(2, -1)$ is the unique Nash equilibrium. For other cost functions, it is possible that the curves take on different shapes and have multiple intersection points, with each intersection point corresponding to a Nash equilibrium. It is also possible that the reaction curves do not intersect, indicating that there are no Nash equilibria. In the case where there are multiple minimum points for each fixed value of θ^1 (or θ^2), we may not have connected "curves," but more generally reaction "sets" and in this case, it should be clear that there can be multiple or no intersection points. Figure 19.2(b) shows a different set of cost functions and reaction curves, but with one Nash equilibrium

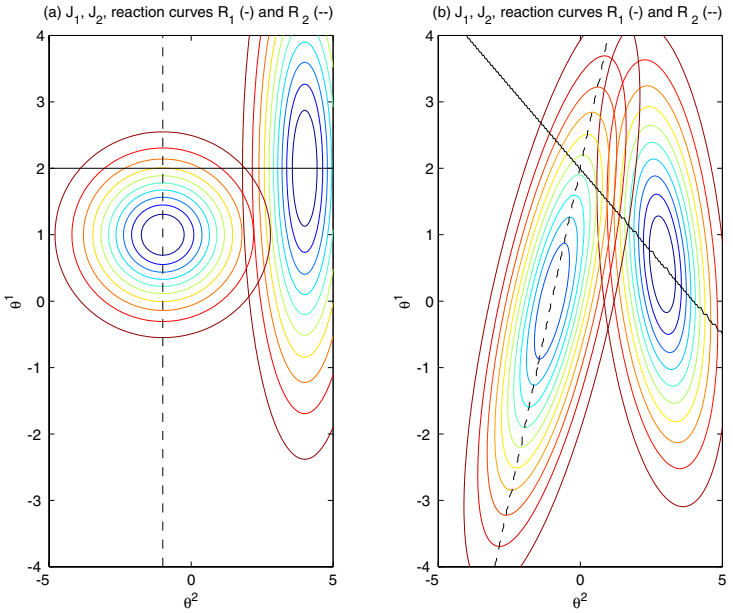

Figure 19.2: (a) Contour plots of J_1 and J_2 and reaction curves R_1 (solid) and R_2 (dashed); (b) same but for different cost functions J_1 and J_2.

corresponding to the intersection point $(2, 0)$. Note that the reaction curve for P_1 is the locus of points that is tangent to lines corresponding to fixed values of θ^2 across a range of such fixed values. Clearly, adjustment of the shape of these functions can easily result in an intersection point that lies outside the allowed ranges of decisions for the players and this would correspond to the case where there are no Nash equilibria.

Finally, note that since the plots in Figure 19.2 were generated on a digital computer, we actually discretized each of the axes and computed a finite number of cost values and points on the reaction curves (as is usual, discrete points are connected by lines in the plots to give a visual effect of continuity when it does not actually exist). Hence, we are actually discussing *finite* bimatrix games, but ones with many possible actions for each player.

Stable Nash Equilibria

It is possible to further refine the characterization of Nash equilibrium solutions and we do this in this section via a simple example. Suppose that for the game pictured in Figure 19.2(b), we have P_1 play first, then P_2, followed by P_1, and so on; hence the players alternate moves with P_1 going first. Suppose that we number the moves with an index k so that at $k = 1$, P_1 moves, at $k = 2$, P_2 moves, and so on. Suppose that each player knows the other's last move, and takes an action that minimizes its losses given this past move. For

an arbitrarily chosen first decision by P_1, the "trajectory" in decision-space is shown in Figure 19.3. Since the reaction curves were already computed for this case, it is easy to construct the trajectory since, once P_1 makes a decision $\theta^1(k)$ at iteration k (k odd), we have

$$\theta^2(k+1) = R_2(\theta^1(k))$$

and once P_2 makes a decision $\theta^2(k)$ at iteration k (k even), we have

$$\theta^1(k+1) = R_1(\theta^2(k))$$

It is this iterative sequence that results in the trajectory. Clearly, if we have nonuniqueness of minimum points and hence, reaction sets rather than simple curves, then appropriate adjustments to this discussion would be needed.

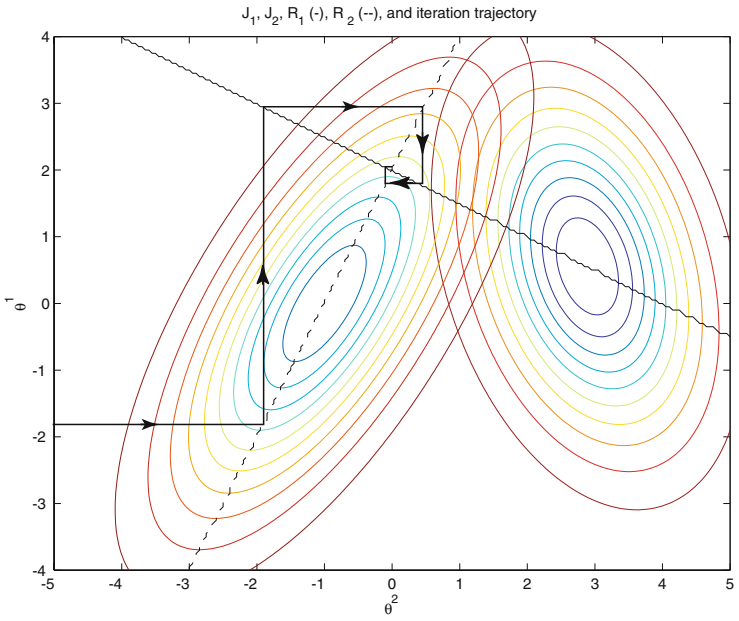

Figure 19.3: Contour plots of J_1 and J_2, reaction curves R_1 (solid) and R_2 (dashed), and iteration trajectory (arrows indicate direction of time).

If for every initial choice by P_1 (i.e., $\theta^1(1)$), the trajectory in the decision space moves to the point $(2, 0)$, then the point $(2, 0)$ is said to be a "stable Nash equilibrium." If only small variations in $\theta^1(1)$ from 2 (its ultimate value for the Nash equilibrium) are possible for it to still converge to 2, then $(2, 0)$ is said to be a "locally stable Nash equilibrium." If a Nash equilibrium point is not stable, it is said to be an "unstable Nash equilibrium." Different cost functions generally result in different reaction curves (sets), which in turn result in different trajectories and hence, different types of stability.

Note that stability is a qualitative property of the trajectories, whose generation depends on *the given iteration method.* Hence, we generally speak of stability relative to the given iteration method. Clearly, the ideas extend to the multiplayer case and then the iteration method depends on that, too. For example, the iteration may involve taking turns in some fixed or dynamically changing order so that the choice by one player at a current time may depend on the movements of some players at the last time instant, and the decisions others made at perhaps random points in the past.

Minimax Strategies

Next, we develop security strategies for each player of a bimatrix game which we will call "minimax strategies" (solutions). These solutions may or may not be the same as a Nash solution.

To find the minimax strategy, you simply find the security strategy for P_1 based on J_1^{ij} and the security strategy for P_2 based on J_2^{ij} and then, taken together, these directly provide a strategy pair that is the "minimax" strategy for the bimatrix game. (Of course, here, since we view the payoff matrices J_1^{ij} and J_2^{ij} as "loss" matrices, there is a slight difference in the mechanics of finding a security strategy for P_2, since now it also tries to minimize its loss under all possible actions of P_1, where before it tried to maximize its gain.) It is interesting to note that P_1 (P_2) does not need knowledge of J_2^{ij} (J_1^{ij}) to compute its strategy. Moreover, the minimax concept essentially ignores whether the opponent is rational or not. These facts can be important in practical applications.

The security levels of the players are called the minimax values of the bimatrix game. These minimax values are not better than those of a Nash equilibrium outcome, even if the Nash and minimax strategies are the same. Why? Also, note that if a minimax strategy is not a Nash strategy (i.e., not an equilibrium), it can still be quite useful in games where there are multiple Nash equilibria, when a player is not completely certain about values of the cost matrix, or whether the other player will be rational. On the other hand, these features can lead to very conservative strategies in some applications. Clearly, minimax strategies can be extended to a multiplayer game in an obvious way.

As an example, show that the minimax strategy for the cost matrices in Equation (19.4), where we had two Nash equilibria, is $(5, 3)$ with an outcome $(2, 1)$. Recall that for this case, the Nash solutions were $(1, 3)$ and $(4, 1)$ with outcomes $(-3, -3)$ and $(-5, -1)$. Note that the minimax strategy is not as good as either Nash solution. Note, however, that for the cost matrices

$$J_1^{ij} = \begin{bmatrix} 4 & 1 & -4 \\ -2 & 5 & 3 \\ -3 & 2 & -1 \\ 4 & 4 & 4 \\ -3 & -5 & 2 \end{bmatrix}, \; J_2^{ij} = \begin{bmatrix} 2 & -1 & 0 \\ -2 & 4 & 0 \\ -3 & 0 & -1 \\ -3 & 3 & 4 \\ -3 & 0 & -5 \end{bmatrix}$$

there is one Nash solution and it is also a minimax solution. Show this.

As another example, consider the game in Figure 19.2(a). Show that the Nash equilibrium is a minimax solution. However, show that for the game of Figure 19.2(b), a minimax solution is $(4, -1)$. In this case, is the Nash solution a minimax solution? Is the minimax solution unique?

Stackelberg Strategies

Until now, for our solution concepts we had a type of symmetry where no one player dominated the decision process. What if one player can enforce her or his strategy on the other players? With this approach we get a "hierarchical equilibrium solution" concept introduced by Stackelberg. For such games, the "policy enforcer" will be called the "leader" and the other player(s) will be called follower(s). We assume that the players act rationally. We could have multiple levels of leaders and followers, but here we just consider the case where we have one leader and one follower.

The basic idea is that P_1 tries to pick i^* to minimize its loss, assuming that with i^* enforced on P_2, P_2 will pick its strategy in a rational way (by minimizing its losses), and then this choice defines the outcome, and hence, allows P_1 to rank its alternatives. In summary, P_1 evaluates its m alternatives by considering what P_2 will pick in response to each one. There is, however, an added complication. There may be more than one P_2 strategy that minimizes its losses for a given P_1 strategy. This creates the possibility that P_1 has more than one different possible loss for each enforced strategy, since P_2 can react in different (rational) ways. Here, we will adopt the convention that we will use a security strategy approach to resolve the ambiguity (i.e., P_1, in the face of several possible answers from P_2, will pick the alternative that will minimize its maximum possible losses).

To compute a strategy pair (i^*, j^*) that is a Stackelberg solution, we execute the following two steps:

1. *Compute Follower Reactions:* For each possible strategy choice i by P_1, compute the set of rational reactions by P_2 as

$$R(i) = \left\{ j^*(i) : j^*(i) = \arg\min_j J_2^{ij} \right\}$$

   This is done by having P_2 execute an optimization over its responses for each given i. Note that $|R(i)| \geq 1$ and if, for example, $|R(i)| = 2$ for some i, then there are two possible rational strategies for P_2 which give P_2 the same loss (i.e., two optima).

2. *Leader Finds Best Strategy:* The leader, taking into account the follower reactions, chooses its strategy via

$$i^* = \arg\min_i \max_{j \in R(i)} J_1^{ij}$$

   which is the P_1 strategy that achieves the lowest loss considering rational P_2 reactions, and which is secure against ambiguities in the response of

P_2 represented in $R(i)$. Note that in finding the "max" and "min" in the computations for i^*, it is possible that there is more than one maximizer and minimizer. For each case, however, it does not matter which you choose, since the "Stackelberg cost" (defined next) is the same. Hence, a normal approach is to resolve ties with an arbitrary choice.

The "Stackelberg strategy" of P_1 is i^*. The Stackelberg solution strategy pair is $(i^*, j^*(i^*))$ and the "Stackelberg cost" for the leader is $J_1^{i^* j^*(i^*)}$. Finally, it is interesting to note that the leader P_1 never does worse in a Stackelberg game relative to if it had played a Nash game. Why?

As an example, to find the Stackelberg solution for the cost matrices in Equation (19.4), with P_1 as the leader and P_2 as the follower, we find that: if $i = 1$, $R(i) = \{3\}$; if $i = 2$, $R(i) = \{2\}$; if $i = 3$, $R(i) = \{1\}$; if $i = 4$, $R(i) = \{1, 3\}$; and if $i = 5$, $R(i) = \{2\}$. The loss to P_1 in each case is then (this takes into account the security component for the $i = 4$ case): if $i = 1$, P_1 loses -3; if $i = 2$, P_1 loses 5; if $i = 3$, P_1 loses 4; if $i = 4$, P_1 loses $5 = \max\{-5, 5\}$; and if $i = 5$, P_1 loses 0. Hence, the Stackelberg solution is $(i^*, j^*) = (1, 3)$ and the Stackelberg cost for P_1 is -3. Note that if we had, for the computation of the P_1 losses, found equal values for more than one i, then there would have been more than one valid Stackelberg solution (but this does not create the problems that we found when we had multiple Nash solutions, since the Stackelberg costs would be the same).

As another example, consider the game in Figure 19.2(b). Clearly the Stackelberg solution will lie on the curve $R_2(\theta^1)$. Why? Show in this case that the Stackelberg solution is $(3.45, 0.75)$, which can be roughly approximated from the plots in Figure 19.4. How? To see how, note that the Stackelberg solution is the pair (θ^1, θ^2) such that θ_1 minimizes $J_1(\theta^1, R_2(\theta^1))$ and this is simply the point where the $R_2(\theta^1)$ curve is tangent to the contour plot of J_1.

19.2.3 Cooperation and Pareto-Optimal Strategies

In the cases above, we generally considered the games to be noncooperative so we assumed that there was a type of adversarial relationship between the two players. In the Nash game, each player is trying to do as well as possible, taking into consideration the other player's actions. In a minimax game, each player assumes the worst possible reactions of the other player (i.e., a highly adversarial, perhaps irrational, opponent) to pick the strategy. There are, however, games where the two players may be able to share information to try to do better; that is, they may cooperate. In a certain sense, the leader-follower game with Stackelberg solutions can be considered a type of cooperative game, if you view the leader-enforced strategy as information that is used by the follower. Whether a Stackelberg game is truly cooperative depends, however, on the application domain and particularly on whether the cost functions of the players have relationships such that the leading actions by P_1 and the subsequent following actions by P_2 are achieving (or trying to achieve) the same type of goal.

In a cooperative game, a player may give up some gains so that the group it is collaborating with gains more.

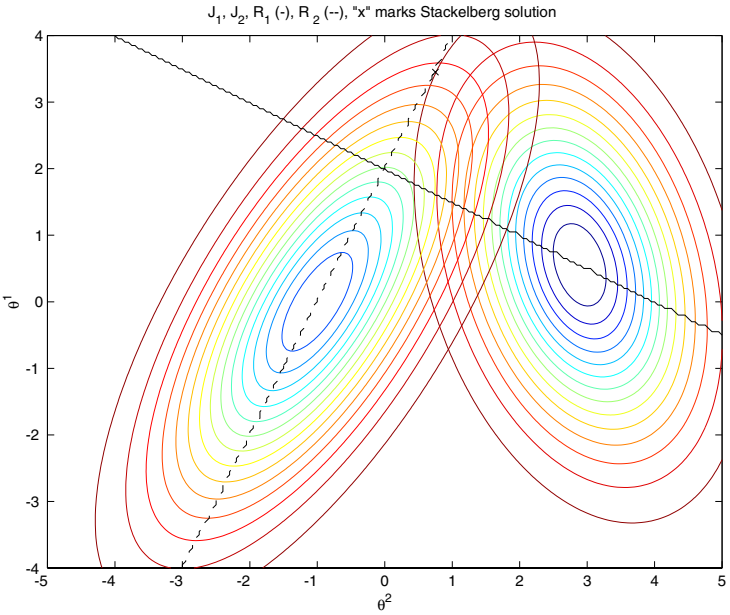

Figure 19.4: Contour plots of J_1 and J_2, reaction curves R_1 (solid) and R_2 (dashed), and "×" marks the Stackelberg solution.

At the other end of the spectrum, there are games that can be considered to be "cooperative" in the sense that each player is willing to share all information and help every other player do as well as possible, so long as it does not degrade its gains too much. A key concept for such games is the idea of a Pareto-optimal (equilibrium) solution, where no other such joint decision exists that can improve the outcome for P_1 or P_2 without degrading the outcome of the other.

Multiobjective Optimization and Pareto Optimality

Here, we begin by defining Pareto optimality in a general setting and then discuss its use in games via some examples. A *multiobjective optimization problem* is in the form of

$$\begin{aligned} &\text{minimize: } \{J_1(\theta), \ldots, J_N(\theta)\} \\ &\text{subject to: } \theta = [(\theta^1)^\top, \ldots, (\theta^N)^\top]^\top \in \Theta \end{aligned}$$

Here, we want to *simultaneously* minimize a *set* of cost functions (called a "cost function vector") by changing the same parameter vector θ. You can think of this as a type of general optimization problem, where you want to minimize not one cost function, but many, each representing the desire to achieve a different objective. Here, we assume that $\theta^i = [\theta^i_1, \ldots, \theta^i_n]^\top$, $i = 1, 2, \ldots, N$, so that decisions are $n \times 1$ vectors, rather than just scalars, and there are N costs to

minimize (e.g., in a two-player game, we will have $N = 2$). Also, Θ is used to represent constraints on the decisions (e.g., constraints on the size of the values of the elements of θ^i, often characterized via functions and inequalities).

A decision vector $\theta^* \in \Theta$ is *Pareto optimal* if there does not exist any other $\theta \in \Theta$ such that

$$J_i(\theta) \leq J_i(\theta^*)$$

for all $i = 1, 2, \ldots, N$, and at the same time

$$J_j(\theta) < J_j(\theta^*)$$

for at least one index j. A cost function vector is called Pareto optimal, if the corresponding decision vector is Pareto optimal. Hence, intuitively, a cost function vector is Pareto optimal, if you cannot improve one cost value without degrading others.

It should be clear that there can be many Pareto optimal solutions. In multiobjective optimization, there is a need to specify *preferences* to be able to pick which Pareto optimal solution specifies an acceptable solution (e.g., one that balances the wins and losses of two players). In many approaches, it is hypothesized that there is a "decision maker" who will specify these preferences in some manner. Sometimes the decision maker will specify a "value function" that can be viewed as a specification of what the decision maker wants in terms of the minimizations of each of the cost functions (i.e., it specifies the trade-offs between players), and other times the decision maker has to be repeatedly asked for its preferences. There are many ways to define such value functions for the decision maker, and the examples below will discuss one way.

Before turning to games, to illustrate the idea of Pareto-optimal solutions, consider the case where θ^1 and θ^2 are scalars, so that θ is a 2×1 vector. Assume that we have quadratic costs so they are convex. Suppose in particular that we have

$$J_1(\theta) = J_1(\theta^1, \theta^2) = \left(\theta^1 - 2\right)^2 + \left(\theta^2 - 3\right)^2$$

$$J_2(\theta) = J_2(\theta^1, \theta^2) = \left(\theta^1 + 2\right)^2 + \left(\theta^2 + 2\right)^2$$

Contour plots of these two functions are shown in Figure 19.5. In this case, it is possible to determine Pareto-optimal solutions by inspection. To do this, ignore the plotted Pareto points in Figure 19.5 and note that the (unique global) minimum points on the two cost functions are at the centers of the two sets of concentric circles. Next, note that if a line on this contour plot is tangent to a contour of *both* costs, then the point of tangency for both costs is a Pareto-optimal solution. (Of course, there are only a finite number of contour lines drawn on the plot so you must imagine where the other ones are.) Where are these "tangency points"? Simple inspection shows that they are at the "$\times$" marks on the plot. Why are these Pareto-optimal solutions? Note that the gradient at such a point θ^* is $\frac{\partial J_1}{\partial \theta}\big|_{\theta=\theta^*}$ and suppose this is pointing in the same direction as $-\frac{\partial J_2}{\partial \theta}\big|_{\theta=\theta^*}$. Imagine that you are at some Pareto-optimal solution θ^* in Figure 19.5. The direction of the negative gradient is the direction to move

from θ^* in order to get a steepest amount of decrease in the value of one cost function (the basic idea behind steepest descent gradient optimization). The key observation is that if you perturb θ^* along the gradient in direction of the minimum point for J_1 (J_2), the cost for J_1 (J_2) goes down, but the cost for J_2 (J_1) goes up. So, we cannot reduce one cost without increasing the other, which is the very definition of Pareto optimality. Note that there are actually an infinite number of points that lie on a line that is tangent to the contour of the two costs, and hence, an infinite number of Pareto-optimal solutions in this case, all of which lie on the line between the minimum points of the two cost functions. The set of all Pareto-optimal solutions is sometimes called the "family" of Pareto-optimal solutions.

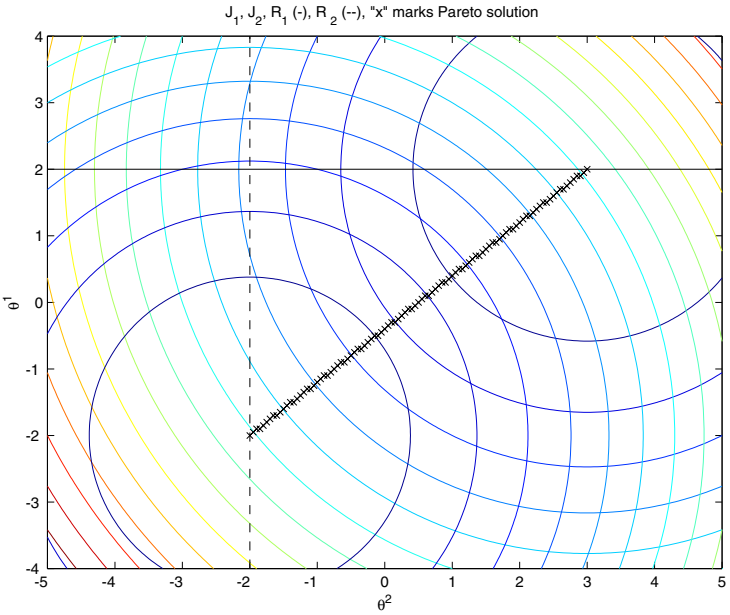

Figure 19.5: Example family of Pareto-optimal points for two quadratic cost functions ("×" marks Pareto solutions).

The scalarization approach is only one way to define the Pareto cost.

It is interesting to note that, for this example, if we define a "Pareto cost" to be

$$J_p(\theta) = pJ_1(\theta) + (1-p)J_2(\theta)$$

for $p \in [0,1]$ (this is called the "scalarization" approach to constructing the Pareto cost), then the family of Pareto points is the set of (unique) global minima for $J_p(\theta)$ as p varies from zero to one, which is just the equation for the line between the two minimum points in Figure 19.5. Hence, we can view this Pareto cost as a "value function" for the underlying multiobjective optimization problem; it is, however, a special one, since it shows how, for this special quadratic case, it is possible to find *all* Pareto-optimal solutions *via standard*

optimization of one cost function. Intuitively, note that the value of p scales the depth of one minimum and $(1-p)$, the depth of the other. We can think of the above weighting scheme via p as *interpolating* between the two minimum points on the two cost functions, with a specific value of p providing more value to minimizing one cost function over the other.

Pareto-Optimal Solutions for Games

Motivated by the above example, a standard way to define Pareto optimality for two-player bimatrix games is to define a new loss matrix (sometimes called the Pareto cost) that represents a combination of the losses for the two players,

$$J_p^{ij} = pJ_1^{ij} + (1-p)J_2^{ij}$$

for $p \in [0,1]$. This Pareto cost can be thought of as a "value function" for the decision maker in a multiobjective minimization problem. *Any* minimum point in the matrix J_p^{ij} is called a Pareto-optimal solution for the bimatrix game (and note that there may be several minimum points for any one value of p). If $p=1$ ($p=0$), all the emphasis is placed on the two players collaborating to minimize the losses of P_1 (P_2). With appropriate values for the cost functions, the value of $p = \frac{1}{2}$ may represent equal emphasis on minimizing the losses of the two players.

As an example, suppose that we use the bimatrix game with payoff matrices in Equation (19.4). Note that these cost values are not specified via a convex function, so there are additional complexities that arise here with uniqueness of the Pareto solutions for fixed values of p. Also, in general, finding a minimum of the Pareto cost J_p^{ij} for all values of p may not provide all possible Pareto-optimal solutions. Why? Returning to our example, note that the minimum element in J_1^{41} is -5, which is also the minimum of J_p for the case where $p=1$. The minimum element $J_2^{52} = -4$, which corresponds to the minimum of J_p for $p=0$. As p varies from zero to one, the minimum point (points?) of J_p will move. Assuming J_p always has a unique minimum point for every value of p, as p varies continuously from zero to one, the minimum point will move continuously from J_2^{52} to J_1^{41}. (If there are multiple minimum points for different values of p, then clearly the situation can be significantly more complex and the family of Pareto points may not all lie on a trajectory.) This is depicted in Figure 19.6, where the indices and outcomes for a set of Pareto-optimal points are shown for this case (note, however, that here we are ignoring the possibility of multiple optima and just finding one for each value of p). Also shown are the losses that result for each player due to the choice of a Pareto-optimal strategy; due to the way that the p parameter results in an interpolation between J_1^{ij} and J_2^{ij} elements, the Pareto value will lie between the two costs as shown for this example.

It should be clear that it is not possible for the outcome to go below the minimum of the two smallest values in the two payoff matrices; hence, the very fact that cooperation is taking place means that one of the two players is

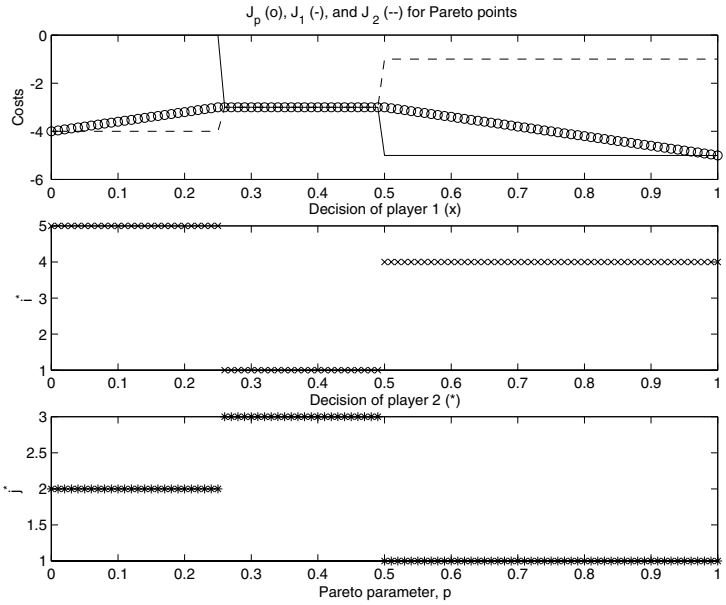

Figure 19.6: Example family of Pareto-optimal points for a bimatrix game (bottom two plots provide the indices of the points), and the resulting outcome (top plot).

sacrificing by losing more than if they were able to minimize their own loss. In applications, however, we are most often interested in the manner in which the Pareto-optimal solution "balances" the optimization to achieve a compromise that is the true goal (cooperation is for achieving a higher goal than individual selfish ones).

Finally, it is interesting to note that $(1, 3)$ and $(4, 1)$ are strategy pairs that we found earlier to be Nash equilibrium solutions for a *noncooperative* game. Recall that in our analysis of nonunique Nash equilibria, the essential problem was that bimatrix games could exhibit elements of "cooperation." Note that here for a cooperative version of the same bimatrix game, as seen in Figure 19.6, we find that for some values of p, the Pareto-optimal solution corresponds to each of the two Nash solutions. This shows in another way that the Nash solutions *required* cooperation of a certain type. For some ways of balancing objectives (values of p), cooperation results in one Nash solution, and for other values of p, it characterizes a different type of cooperation and hence, a different Nash solution.

Defining the Pareto Cost and Finding Pareto Solutions

In this section, we will outline a few problems that you encounter in trying to define Pareto costs and compute Pareto solutions. We will do this via the game

in Figure 19.2(b) and, analogous to the above example, let

$$J_p(\theta^1, \theta^2) = pJ_1(\theta^1, \theta^2) + (1-p)J_2(\theta^1, \theta^2)$$

for $p \in [0, 1]$. To give insight into the shape of the cost surface J_p, see Figure 19.7, which is the case for $p = 0.5$. Clearly there are multiple local minima so finding the global one can be challenging. Also, note that the p parameter will in this case scale the "depth" of the two minima.

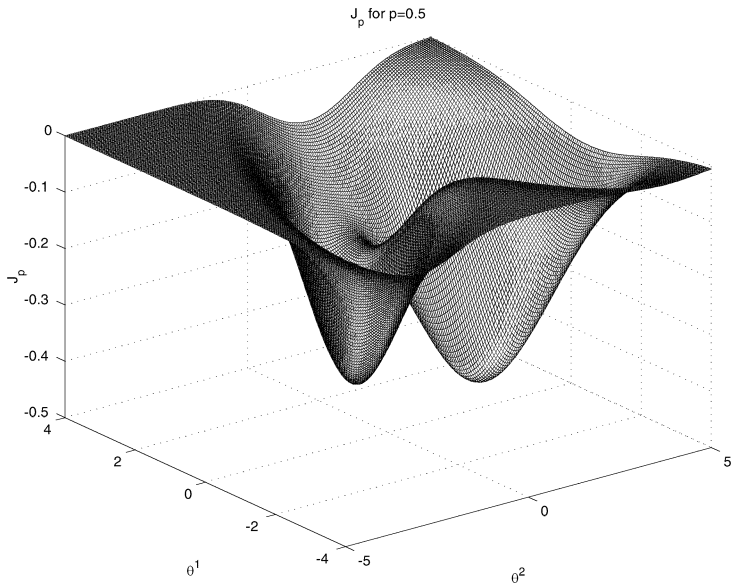

Figure 19.7: Pareto cost $J_p(\theta^1, \theta^2)$ for $p = 0.5$.

Problems with Forming Pareto Costs via Scalarization: Figure 19.8 shows two possible Pareto solutions for this case, which are $(0, -1)$ (roughly for $p \in [0, 0.455]$) and $(0.55, 2.99)$ (roughly for $p \in (0.455, 1]$). Neither of these Pareto solutions corresponds to a Nash, minimax, or Stackelberg solution for this game. Intuitively, as p varies from zero to one, there is a point at which the deepest "well" in Figure 19.7, switches from one well to another. The "family" of Pareto solutions lies only on two isolated points. Hence, in this special case, there is the curious property that the Pareto points lie on the reaction curve of one or the other player. Hence, for some values of p, even with cooperation, P_1 (P_2) ends up gaining significantly and P_2 (P_1) loses significantly (i.e., cooperation here entails sacrifice by one player for the other). For some applications this may be satisfactory; however, for others, you may need to choose a different form for $J_p(\theta^1, \theta^2)$ that allows for a *smooth balancing* between the selfish objectives of P_1 and P_2.

It can be difficult to pick an appropriate Pareto cost for an application.

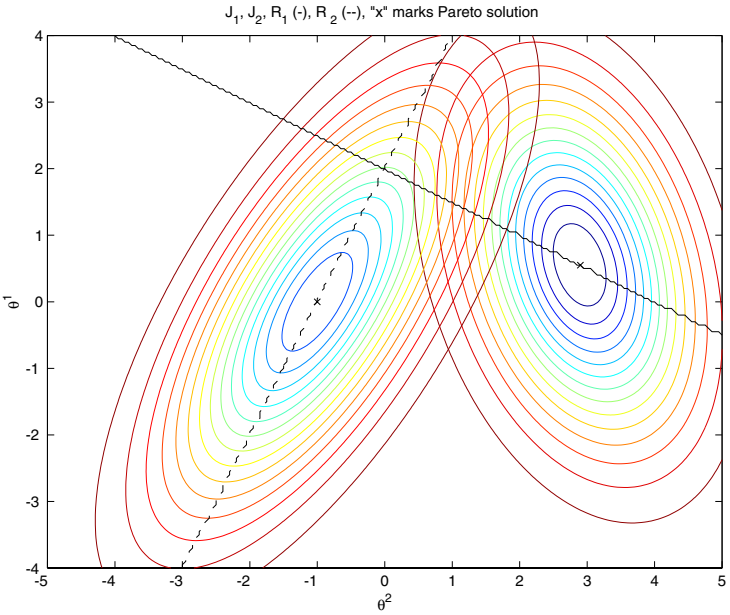

Figure 19.8: Contour plots of J_1 and J_2, reaction curves R_1 (solid) and R_2 (dashed), and "×" marks the two Pareto solutions.

Choice of Pareto cost affects how cooperation between players is achieved.

The Family of Pareto Points—Other Ways to Balance Cooperation: By inspection, it should also be clear that there are *many* other Pareto-optimal solutions. Can you sketch additional Pareto-optimal points on Figure 19.8? Would a point such that the gradients of both cost functions point in opposite directions be a Pareto point? If it were, then, where are such points on Figure 19.8? How would you *compute* all Pareto-optimal solutions for this case? A computationally intensive approach to approximating the set of all Pareto points is to simply directly apply the definition of Pareto optimality (given m and n, how many comparisons are needed to compute all Pareto solutions for a bimatrix game?). When we do this, we get all the Pareto points shown in Figure 19.9 (notice the rough edges on the contour plot due to the coarse discretization). Of course, these points include the ones that result from the scalarization approach of the above example. The others represent other ways to balance the two performance objectives. Notice that the family of Pareto points is not a point or a curve, but a set. In general, it is difficult to characterize or compute the *entire* set of Pareto points, except in certain special cases (e.g., like the one we examined earlier).

Essentially, the definition of our value function J_p via scalarization results in missing all these other Pareto-optimal points. Is this good or bad? If you consider the value function found via scalarization to specify your true preferences, then missing other Pareto-optimal solutions is not a problem. If, however, other

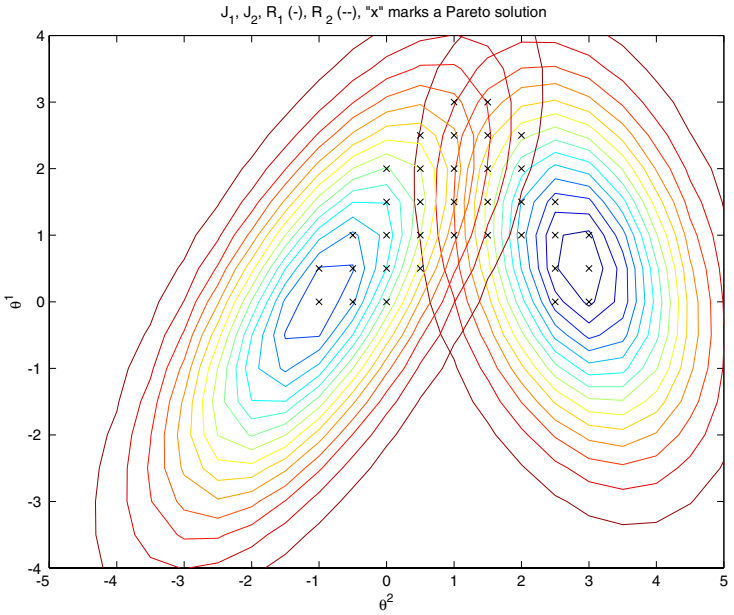

Figure 19.9: Family of Pareto points.

types of preferences are needed, it may be difficult to know how to parameterize and define the J_p function, so that conventional optimization can be used to find what you consider to be good Pareto points. One other way to parameterize the above function would be to have p and $(1-p)$ scale the minimum points of the functions; however, in practical examples you may not know the minimum points a priori. In short, this example shows some of the problems one can encounter by using a simple weighting scheme to turn a multiobjective optimization problem into a conventional single-objective optimization problem to find a Pareto-optimal solution. Other ways of forming the Pareto cost can result in a better balancing and hence, a more "fair" cooperation may be obtained.

19.3 Design Example: Static Foraging Games

In this section, we study the basics of competition and cooperation in foraging games by considering a static, full-information, one-stage (i.e., one decision) game to illustrate the Nash, minimax, Stackelberg, and Pareto solution concepts of Section 19.2.

19.3.1 Static Foraging Game Model

We start by formulating a very simple model of a foraging game; after its development it will become clear how to extend this model to more general cases.

We consider a two-forager ($N = 2$), static, discrete, full-information "foraging game on a line." This is the one-dimensional case, where we consider multiple resources to be distributed over the real line and the foragers move on that line to get the resources.

Foraging games have foragers as players and nutrients as payoffs.

Suppose that the resources are distributed in "cells" (bins) along the real line. Suppose that there are M different types of resources in Q cells and denote the initial distribution of resources of type m to be $r^m(q)$, $q = 1, 2, \ldots, Q$, $m = 1, 2, \ldots, M$. Here, we assume that $r^m(q) \geq 0$, $q = 1, 2, \ldots, Q$, but the model is easily extended to the negative resource case (where one could think of moving to avoid regions where resources are lost). Let D_1 (D_2) be the number of decisions that forager 1 (2) can make and $\theta^1 \in \{1, 2, \ldots, D_1\}$ ($\theta^2 \in \{1, 2, \ldots, D_2\}$) be those decisions, which correspond to forager 1 (2) moving to a cell q if $\theta^1 = q$ ($\theta^2 = q$), $q = 1, 2, \ldots, Q$. We assume that $D_1 = D_2 = Q$, so that each forager can move to any available cell.

Effort and Resource Consumption

Effort expenditure can be counted against nutrient returns in computing payoff.

Let z_1 (z_2) denote the effort allocated by forager 1 (2) to consume resources. For simplicity, we assume that the same amount of effort is expended for consumption of each resource type $m = 1, 2, \ldots, M$ when a forager goes to a cell. Let $P(q)$ be the set of foragers that decides to go to the same position q to consume resources there; hence,

$$P(q) = \{i : \theta^i = q\}$$

Notice that $0 \leq |P(q)| \leq N$ for all q and $\sum_{i \in P(q)} z_i$, the total consumption effort at q, is zero if $|P(q)| = 0$.

Assume that α^m, $m = 1, 2, \ldots, M$, is used to model the depletion rate of resource m in the presence of consumption effort. We model the amount of resource of type m remaining at cell q after one play (one unit of expenditure of effort) as

$$r^m(q) e^{-\alpha^m \sum_{i \in P(q)} z_i}$$

This type of model, which is used in foraging theory, represents that initial expenditures of effort in a cell yield more resources than later ones. Hence, as resources diminish in a cell, there is a need for increasing amounts of effort to get the same return. With the exponential model, effort expenditure always provides a return on the investment; other models could represent complete depletion of a resource after a finite amount of effort.

Next, we define the amount of *consumption* given that a strategy pair (θ^1, θ^2) is played by foragers 1 and 2. To do this, note that if both foragers are in the same cell expending effort to consume the same resource, then they have to split the resource, since there is a type of competition for it. Here, we simply assume that if two foragers are at the same cell, then they split the resources evenly. Let the amount of consumption of resource m for decision pair (θ^1, θ^2) for foragers 1 and 2 be defined as follows:

1. *Foragers at different locations:* If $\theta^1 \neq \theta^2$, then for $i = 1, 2$,

$$C_i^m(\theta^1, \theta^2) = r^m(\theta^i)\left(1 - e^{-\alpha^m z_i}\right)$$

2. *Foragers at the same location:* If $\theta^1 = \theta^2 = \bar{\theta}$, then for $i = 1, 2$,

Foragers going to the same location results in a type of competition.

$$\begin{aligned} C_i^m(\theta^1, \theta^2) &= \frac{1}{|P(\bar{\theta})|} r^m(\bar{\theta})\left(1 - e^{-\alpha^m \sum_{i \in P(\bar{\theta})} z_i}\right) \\ &= \frac{1}{2} r^m(\bar{\theta})\left(1 - e^{-\alpha^m (z_1 + z_2)}\right) \end{aligned} \tag{19.5}$$

So, in cases where forager 1 (2) goes to a cell that forager 2 (1) does not go to, $r^m(\theta^1)$ ($r^m(\theta^2)$) is the initial amount of resource of type m and $r^m(\theta^1)e^{-\alpha^m z_1}$ ($r^m(\theta^2)e^{-\alpha^m z_2}$) is the amount remaining after consumption. When both foragers go to the same cell, then they both expend effort, but they have to split the returns in half. This results in a resource conservation property of: "all that is consumed plus what is remaining is equal to what was initially there."

Forager Payoffs: Consumption, Energy, and Danger Avoidance

We assume that each forager has certain priorities to consume different resources. We denote these by p_1^m (p_2^m) for forager 1 (2), $m = 1, 2, \ldots, M$. You can think of these priorities as representing preferences or "tastes" for resources. One aspect of the cost to forager 1 (2) that it wants to minimize is given by the negative total consumption weighted by the priorities

$$\begin{aligned} J_{1c}^{ij} &= J_{1c}(\theta^1, \theta^2) = -\sum_{m=1}^{M} p_1^m C_1^m(\theta^1, \theta^2) \\ J_{2c}^{ij} &= J_{2c}(\theta^1, \theta^2) = -\sum_{m=1}^{M} p_2^m C_2^m(\theta^1, \theta^2) \end{aligned}$$

where $\theta^1 = i$, $\theta^2 = j$, and J_{1c}^{ij} and J_{2c}^{ij} constitute a matrix representation of the game. So the problem for forager 1 (2) is how to pick θ^1 (θ^2). The adversarial nature of the foraging game will dictate what to choose (e.g., in a competitive game, each forager may get less than if they cooperate).

Foraging often requires energy consumption to go to a cell from some initial location (e.g., for locomotion). Here, we will think of the foragers as being located at position "0" (i.e., on one edge outside the foraging area) initially. Then, we model the cost to move along the line to go to position i (j) for forager 1 (2) as

$$J_{1e}^i = J_{1e}(\theta^1) = w_{e1} i \quad \left(J_{2e}^j = J_{2e}(\theta^2) = w_{e2} j\right)$$

where $\theta^1 = i$, $\theta^2 = j$, $w_{e1} \geq 0$ and $w_{e2} \geq 0$ represent the unit amount of energy expenditure to move one unit (e.g., from cell 1 to cell 2), and we assume that

energy expenditure is not affected by the actions of the other forager. When the energy to conduct foraging is taken into account, then it may be possible that even though a resource is plentiful, a forager may choose a closer one that would provide less return on its effort investment, so that it tries to maximize its amount of resource return for a certain investment in foraging energy. Clearly, there are many ways to model this basic aspect of foraging.

For many foragers, there are areas of the foraging environment that are more "dangerous" than others. This may be due to a predator who is trying to consume the forager, or other environmental characteristics (e.g., presence of a noxious chemical). Clearly, one could model the situation where one forager "consumes" the other. Here, we consider a simple model of location-dependent danger for forager 1 (2) with

$$J_{1d}^{i} \geq 0 \ \left(J_{2d}^{j} \geq 0\right)$$

where bigger values of the costs represent worse areas to be in and actions of the other forager do not affect the danger to a forager.

A forager generally wants to get as many high priority resources for a given energy investment, while avoiding as many dangers as possible. Hence, we can think of forager 1 (2) trying to minimize

$$J_{1}^{ij} = J_{1c}^{ij} + J_{1e}^{i} + J_{1d}^{i} \ \left(J_{2}^{ij} = J_{2c}^{ij} + J_{2e}^{j} + J_{2d}^{j}\right)$$

so that it maximizes the amount of resources it gets and minimizes the energy expenditure and exposure to dangers to get them. With this, if $J_{1c}^{ij} = J_{2c}^{ij} = 0$, $J_{1e}^{i} > 0$, and $J_{2e}^{j} > 0$ for all i and j, and $J_{1d}^{i} > 0$ and $J_{2d}^{j} > 0$ for all i and j, then the foragers would not even want to move to a location, since there would be no return of resources for an energy expenditure and exposure to danger (however, for our model we force them to move, so they are not able to choose the option of not playing the game). Generally, the foragers will move farther for resources that are more important to them, but that assumes there is not too much danger.

Notice that we have set this up as a static bimatrix game. Hence, each forager knows everything about the game (e.g., the payoffs, costs of movement, dangers, the other forager's objectives, etc.). Next, for the sake of illustration, we will provide a numeric example.

19.3.2 Competition and Cooperation for a Resource

Choose $D_1 = D_2 = Q = 21$, $M = 1$, $z_1 = z_2 = 1$, $\alpha^1 = 1$, $p_1^1 = p_2^1 = 1$, and $w_{e1} = w_{e2} = 0$ (no energy required for foraging). Assume that $J_{1d}^{i} = J_{2d}^{j} = 0$ for all i and j. The initial resource distribution is shown in Figure 19.10.

The cost functions J_1^{ij} and J_2^{ij} are plotted in Figures 19.11 and 19.12. Notice in Figure 19.11, that if you hold j constant, then forager 1 generally gets more consumption and hence, more payoff if it moves to where there are more resources; however, if both foragers move to the same location, they get less,

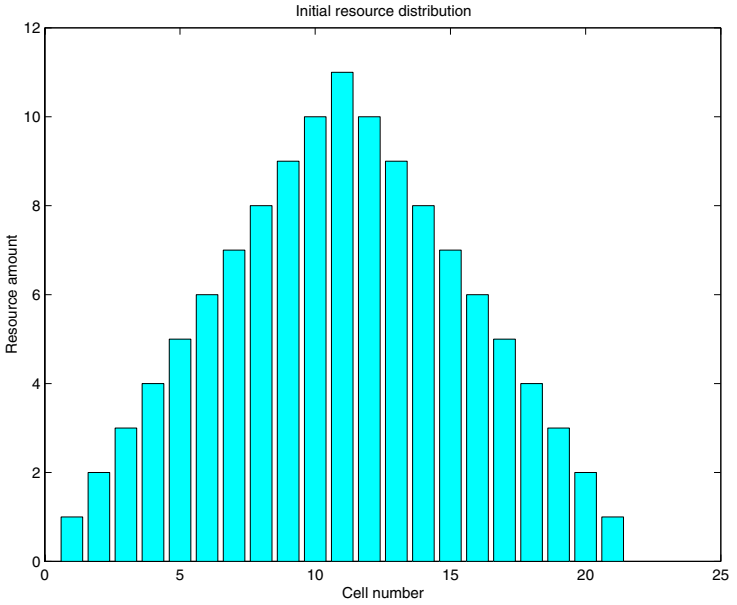

Figure 19.10: Example initial resource distribution.

since they will then compete for resources at that cell. This competition is represented by the ridges of increased cost (a competition cost) that cut diagonally through Figures 19.11 and 19.12.

In competitive foraging, forgers may gain less than if they foraged cooperatively.

First, suppose that we have an adversarial (noncooperative) game, so that the foragers do not coordinate where to go to forage. There are four Nash solutions

$$(10, 11), (11, 10), (11, 12), (12, 11)$$

Does this make sense? From Figure 19.10, the cell with the most resources is cell 11. In the presence of competition, one forager gets the most resources and the other gets the second highest amount possible and these are the four strategy pairs that represent this. Note, however, that the problem of nonunique Nash solutions arises. Forager 1 may pick 11, and with no communication and coordination, forager 2 may also pick that point and there will then be *less* payoff than if the above solutions were chosen. In fact, the minimax solution in this case is $(11, 11)$, since if each forager tries to minimize its maximum possible losses, then it will go to the location with a maximum number of possible resources, since if it goes to a cell with fewer resources, then the other forager can go there also and both would get even fewer resources. A Stackelberg solution with forager 1 as the leader is $(11, 12)$. Why?

Next, consider a cooperative foraging game. First, suppose that the two foragers cooperate by using a Pareto cost found via scalarization as $J_p^{ij} = pJ_1^{ij} + (1-p)J_2^{ij}$ with p as the Pareto parameter that balances the cooperation. Pareto

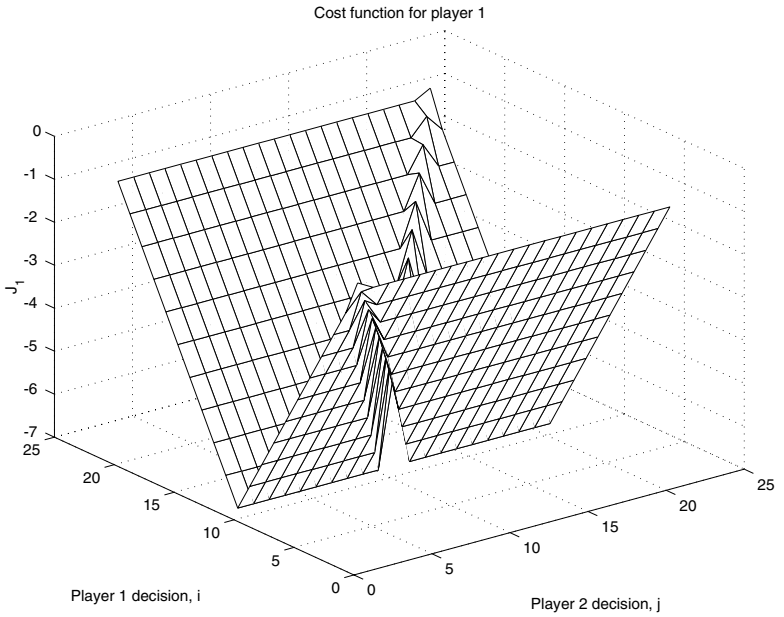

Figure 19.11: Cost for forager 1, J_1^{ij}.

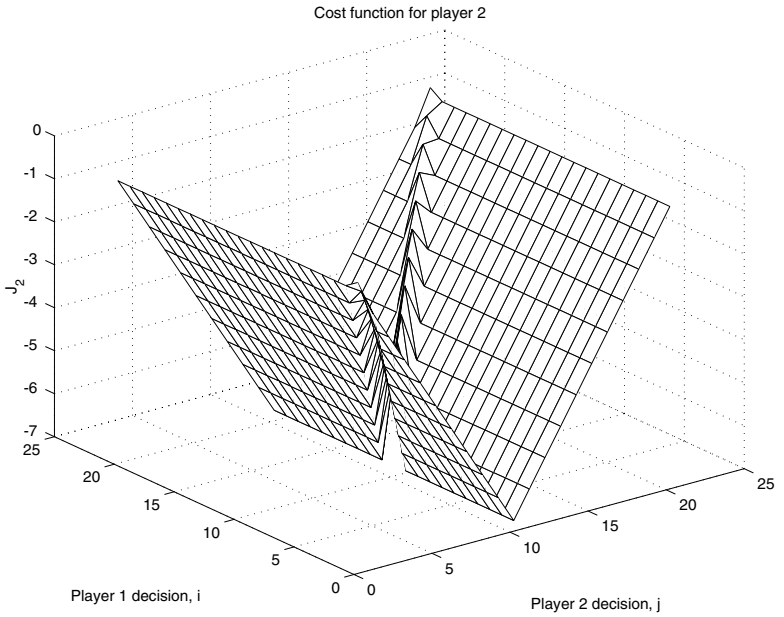

Figure 19.12: Cost for forager 2, J_2^{ij}.

points found in this case are shown in Figure 19.13. We get Pareto points (which are also Nash solutions) $(10, 11)$ or $(11, 10)$, depending on the Pareto parameter p so long as $p \in (0, 1)$. The two foragers would communicate to decide who goes to which location, which, as opposed to the Nash game, is possible since the two foragers are cooperating. The one that goes to position 11 will get the most resources. When p is close to zero, it favors forager 2, so forager 2 goes to position 11, and when p is close to one, it favors forager 1, so forager 1 goes to position 11. The p parameter can be used to balance the cooperation to favor one forager or the other. What happens in the case where $p = 0$? Then, the J_1^{ij} cost does not enter into J_p^{ij} since it is multiplied by $p = 0$. This means that forager 2 makes its best decision and goes to position 11, and forager 1 can go *anywhere* else. In Figure 19.13, it goes to position 1, simply due to how the code was written. The case for $p = 1$ is explained similarly.

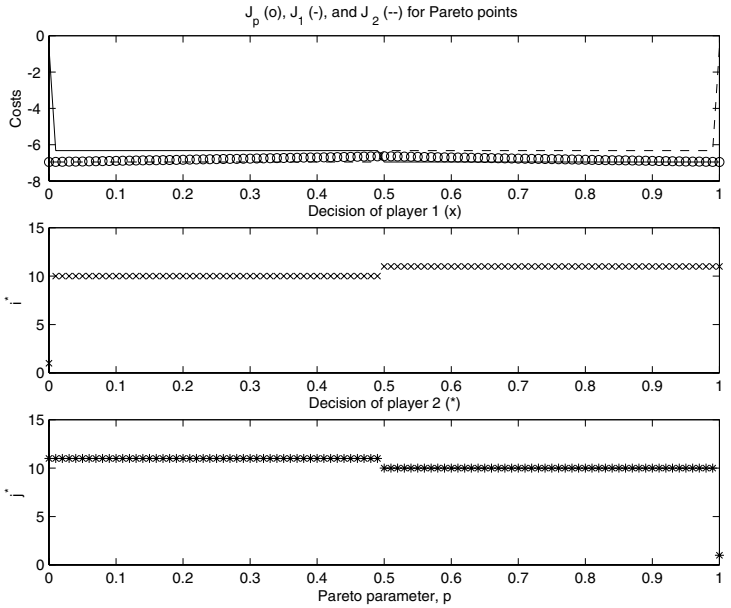

Figure 19.13: Set of all Pareto points for a cooperative foraging game, scalarized Pareto cost.

The scalarization approach is, however, only one way to form the Pareto cost. The set of all Pareto points for the game is shown in Figure 19.14, and you can see that the ones that arise from the above scalarization approach are a subset of all possible Pareto points. These other Pareto points represent different ways to balance the payoffs to each of the two foragers. First, notice that all the Nash solutions are a subset of the Pareto points. Why do the other Pareto points make sense (e.g., the one at $(11, 20)$)? What type of Pareto cost might be used for those cases?

It is interesting to note that you can view a cooperative foraging game as one

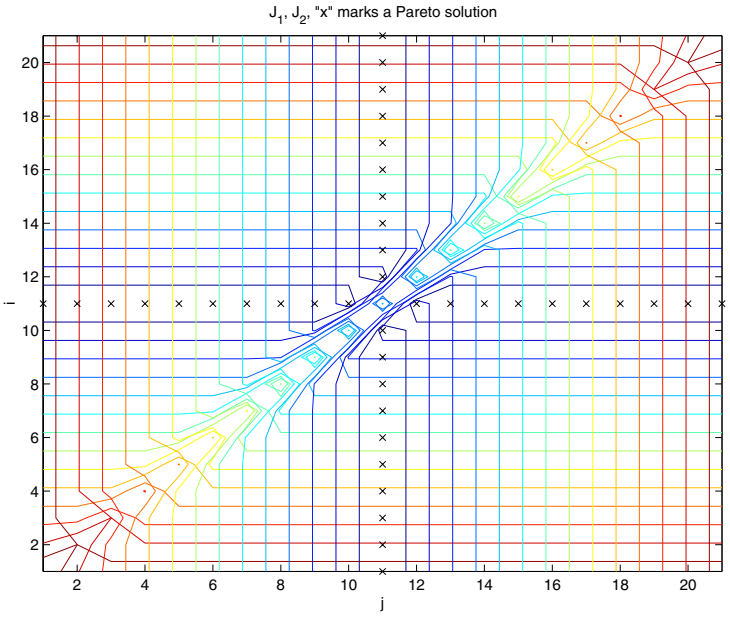

Figure 19.14: Set of all Pareto points for a cooperative foraging game.

where you try to *allocate* resources to all the foragers so that everyone "wins," with the relative payoffs given by which Pareto points you choose. In this case, scalarization provides a nice way to balance the allocation.

19.3.3 Energy Constraints and Multiple Resources

Energy expenditure considerations drastically influence foraging decisions.

Choose $M = 2$, $p_1^1 = p_1^2 = 1$, $p_2^1 = 1$, and $p_2^2 = 2$, so that forager 2 places a high priority on getting resource 2. We let $w_{e1} = w_{e2} = 0.1$, so that moving to cell locations with higher values of q costs more energy. As before, we have $D_1 = D_2 = Q = 21$, $z_1 = z_2 = 1$, $\alpha^1 = \alpha^2 = 1$, and $J_{1d}^i = J_{2d}^j = 0$ for all i and j. The initial resource distribution for the two resources is shown in Figure 19.15.

The cost functions J_1^{ij} and J_2^{ij} are plotted in Figures 19.16 and 19.17. The "ridge" arises as in the last section and it represents the case where the two foragers choose the same cell and hence, compete. Focus on Figure 19.16, and notice that, even though it sets an equal priority for both resources, the costs generally increase as i increases (ignoring the ridge) due to the presence of the J_{1e}^i term that represents the energy needed to forage at each position. This raises the cost of the second resource. Notice that in Figure 19.17, we have the presence of this same effect, *and* the effect of the higher priority of resource 2 for forager 2 so that for forager 2, even though it has to travel farther to get resource 2, since it is higher priority, it may be willing to do that.

Consider the competitive case first. The unique Nash solution is $(5, 14)$.

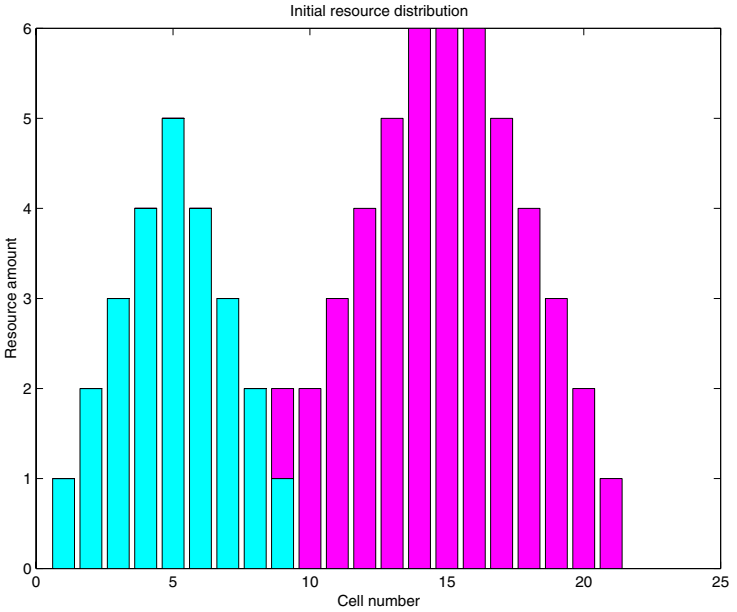

Figure 19.15: Initial resource distribution (darker shaded bars on the right are the second resource) with an "overlap" of resources in the middle designated by "stacking" the plots).

Essentially, with the above choices, forager 1 chooses resource 1 since it is close, but forager 2 picks resource 2, since its level of priority is high, so it is willing to expend the energy to get it. Notice that the maximum for the second resource is achieved at three contiguous positions, but forager 2 picks the smallest of these to minimize energy. The minimax and Stackelberg strategies are both $(5, 14)$. Why?

If the foragers enter into a cooperative game, with a scalarized Pareto cost $J_p^{ij} = pJ_1^{ij} + (1-p)J_2^{ij}$, then we get the Pareto solutions all at $(5, 14)$ for all p. In this case, the two foragers' objectives are so different that there is nothing to be gained by cooperation (and nothing to be lost by competition) and hence, there is really no need for communication.

Cooperative foraging strategies can be viewed as a type of allocation of foragers to resources to maximize payoff to the group.

19.4 Dynamic Games

Dynamic games are ones where players use information about how the game has evolved in order to make decisions. This notion is perhaps closer to the common notion of a game, where there are repeated observations, decisions, and actions by each player, and a resulting dynamic interaction between players in some "arena."

Dynamic games consider repeated decisions, actions, and observations by the players.

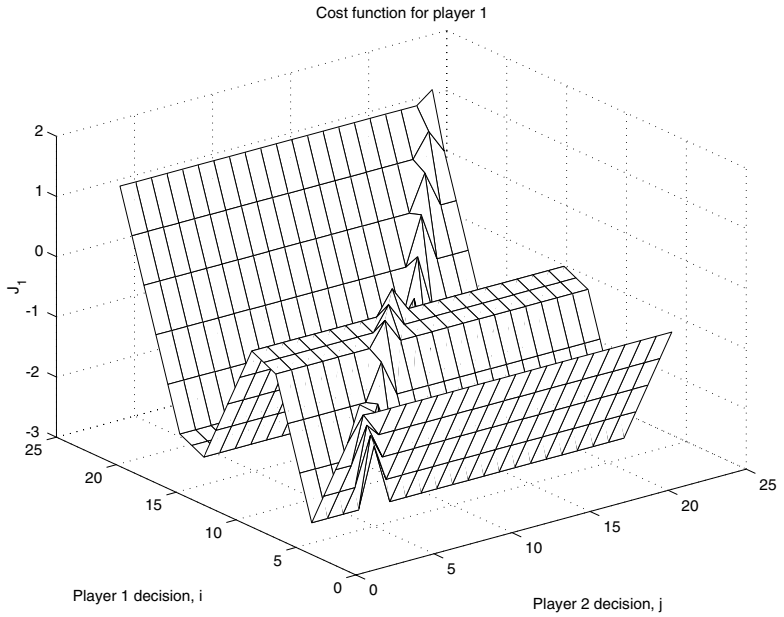

Figure 19.16: Cost for forager 1, J_1^{ij}.

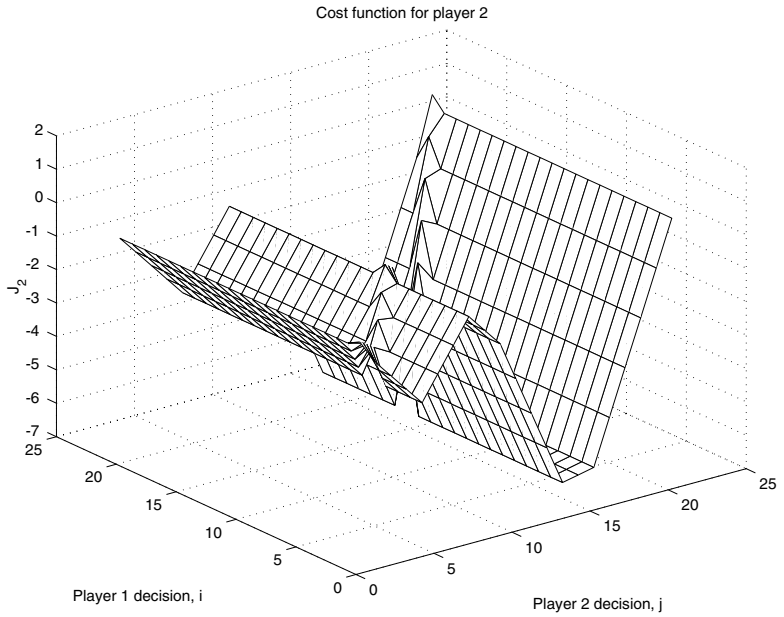

Figure 19.17: Cost for forager 2, J_2^{ij}.

19.4.1 Modeling the Game Arena and Observations

We assume that there are N players and use a discrete-time formulation. To define the dynamic game, we will produce a model of the game, including all the players, rules, and payoffs. First, let

$$x(k) \in X \subset \Re^{n_x}$$

denote the state of the game at time k, $k \geq 0$. The admissible controls (actions) by player i are for $k \geq 0$

$$u^i(k) \in U^i(k) \subset \Re^{n_u}$$

The outputs (measurements of what is happening in the game) are, for $k \geq 0$,

$$y^i(k) \in Y^i(k) \subset \Re^{n_y}$$

Let

$$u(k) = \left[(u^1(k))^\top, (u^2(k))^\top, \ldots, (u^N(k))^\top\right]^\top$$

and

$$y(k) = \left[(y^1(k))^\top, (y^2(k))^\top, \ldots, (y^N(k))^\top\right]^\top$$

Define the "arena" in which the game is played as f where

$$x(k+1) = f(x(k), u(k), k) \tag{19.6}$$

and suppose that the initial state of the game is $x(0) \in X$. This is a deterministic game model, but it can be time-varying. A stochastic game could be represented with $x(k+1) = f(x(k), u(k), w(k), k)$ where $w(k)$ is used to model stochastic effects.

The observations that player i can make about the arena of the game are specified by the function $y^i(k) = h^i(x(k), k)$ for $k \geq 0$, and if we let $h(x(k), k) = \left[(h^1)^\top, (h^2)^\top, \ldots, (h^N)^\top\right]^\top$, then

$$y(k) = h(x(k), k) \tag{19.7}$$

You could view h as part of the representation of the arena of the game as it models what can be observed by each player while the game is played. The dynamic game evolves by players iteratively making a sequence of decisions and taking a sequence of actions. Observations lead to decisions, which lead to actions, which lead to observations, and so on.

Let $J_i(x(k), u(k))$ denote the loss (cost) function of the i^{th} player at the k^{th} stage of play. When there are multiple stages of play (e.g., N_s stages), one typical choice for the loss of each player is the additive one,

$$J_i^{N_s} = \sum_{k=0}^{N_s-1} J_i(x(k), u(k)) \tag{19.8}$$

Hence, each player tries to choose a sequence of $u^i(k)$ that will minimize its own loss $J_i^{N_s}$ after N_s actions, within the constraints of the game listed above. One other typical choice for the loss function is

$$J_i^{N_s} = \sum_{k=0}^{N_s-1} J_i(x(k+1), x(k), u(k)) \tag{19.9}$$

so that losses are assigned based on the type of change in the state and the player actions, for N_s actions. Note that in general, player i does not know its own cost function $J_i(x(k), u(k))$ since it may not know $x(k)$ and $u(k)$. Moreover, use of other players' cost functions $J_j(x(k), u(k))$, $j \neq i$, in the strategy of a player i requires special assumptions.

19.4.2 Information Space and Strategies

How are player's strategies defined? This is a bit more complicated than in the static game case. Why? Because, now each player may make decisions based on "what they know and when they know it" and hence, it is not assumed that each player knows everything at one time and only one action is taken by each player at that time. To make this more precise, it should be clear that if a player has memory, it can store and recall past observations. Then, for any player i, at the k^{th} stage of play, it may base its decisions to choose $u^i(k)$ on a subset of

$$\begin{aligned}&\{y^1(0), \ldots, y^1(k); \cdots; y^N(0), \ldots, y^N(k);\\ &u^1(0), \ldots, u^1(k-1); \cdots; u^N(0), \ldots, u^N(k-1)\}\end{aligned}$$

(clearly the elements of the subset are defined by what information is available to each player i and when it is available). Note that $u^i(k)$, $i = 1, 2, \ldots, N$, is not allowed in the above set, since these are the decisions that the players are trying to reach based on the available information at time k. Also, if $k = 0$, then there are no u^i elements available. Each such subset is called an "information structure" and the information structure of the game is the set of all such subsets that are used. Let

$$\begin{aligned}I^i(k) \quad \subset \quad &(Y^1(0) \times \cdots \times Y^1(k)) \times \cdots \times (Y^N(0) \times \cdots \times Y^N(k)) \times\\ &(U^1(0) \times \cdots \times U^1(k-1)) \times \cdots \times (U^N(0) \times \cdots \times U^N(k-1))\end{aligned}$$

denote the "information space" of player i at time $k \geq 0$ (again, for $k = 0$, the information space along the u^i dimensions collapses, since there is no past decision information).

Note that the information space $I^i(k)$ is implemented via an appropriately defined communication network between the players and memory within each player to store past values. In an adversarial game, there may be no communication links between the players, but there may need to be memory to hold past information that was encountered. However, in a cooperative game, the sharing of information that is necessary for real-time cooperation comes via communication. A communication network has a "topology" of communication links

between players, each of which may provide for uni- or bi-directional transfer of information between players. (It is sometimes useful to think of the communication network as a graph with nodes as players, arcs as communication links, and the topology as being defined via the interconnection pattern of the nodes.) There can be bandwidth constraints and random but bounded communication delays on any link. Moreover, the topology may be "dynamic," in that it may change based on aspects of the environment or player actions.

In some cases, the constraints of the game may specify $I^i(k)$, but other times, the designer may be able to choose it. For example, if all players only make decisions based on their own current observations of the arena of play and the previous actions of all other players and itself, then

$$I^i(k) \subset (Y^i(k)) \times (U^1(k-1) \times \cdots \times U^N(k-1))$$

A strategy for a player i at the k^{th} stage of play is G_k^i, $k \geq 0$,

$$G_k^i : I^i(k) \to U^i(k)$$

The design of strategies involves designing *both* $I^i(k)$ and G_k^i. For example, if each player i can observe at stage k, only $y^i(k)$ (i.e., its only observation) and all actions $u^i(k-1)$, $i = 1, 2, \ldots, N$, the strategies of the players are defined via a G_k^i mapping for each player that specifies its actions,

$$u^i(k) = G_k^i(y^i(k), u^1(k-1), \ldots, u^N(k-1))$$

(and at $k = 0$, there are no elements in the u^i slots). The "full state feedback" case is when $y^i(k) = x(k)$ for all i and k and is the case where each player has "perfect information" about the arena of play. Note that for each fixed initial state $x(0)$ and fixed player strategies, unique sequences of $u^i(k)$, $x(k)$, and $y^i(k)$ and loss values are generated.

The standard concepts of saddle point, Nash, and Stackelberg equilibrium solutions can be extended to dynamic games. For instance, if the initial state is known, the strategies are fixed, and there are a fixed number of stages of play, then the above provides a normal form description. For this case, we get unique loss values for all the players for a given strategy. Note, however, that there are additional solution concepts, depending on the information space that is assumed. For instance, depending on whether you know the initial state, the sequence of states up to the current one, or only the current state, you get different solution concepts (e.g., the "feedback Nash solution"). Here, we will not consider these additional solution concepts and methods for solving for strategies for those cases. Our focus will be on using only the basic game concepts of the last section, but within the context of dynamic games. This choice is driven by practical issues, such as a desire not to consider only the well-understood "linear system with quadratic cost" case (it is covered in detail elsewhere), yet the need to avoid very general dynamic game formulations that cannot be solved analytically, or that sometimes demand computationally intractable solutions.

19.4.3 Decision and Action Timing

It is useful to clarify some issues related to timing of when decisions (actions) are taken in a dynamic game. First, we think of the game as proceeding over a finite number of N_s stages (time steps), or in some cases it may be appropriate to consider $k \to \infty$. Second, while we use a discrete-time model, the actions need not be *synchronous*, in the sense that you think of the index k as being associated with real time $t = kT$ and $t' = kT + T$ for a fixed sampling period T for all k. The actions of the players may occur *asynchronously*, in the sense that the real time duration between indices k and $k + 1$ may be nondeterministic. Third, note that for the above model, all players act at each stage k. We can often, however, for specific applications, define a "null play" choice that corresponds to an action that is equivalent to doing nothing. Fourth, in some games, "simultaneous play" by two or more players may be possible.

Combining these four points, we see that we can represent a dynamic game where players can independently act at random points in time and do not have to be in "lockstep" with each other (e.g., in a two-player game with the two players taking turns). There is a new index $k + 1$ whenever *any* player acts. If at that time no other player acts, we represent this via using their null plays. These issues of representation can be important in practical games, where the timing of play is a critical aspect of the arena of play.

19.5 Example: Dynamic Foraging Games

We begin by using the model of a dynamic game to represent a foraging game. Next, we use biomimicry of foraging in nature to specify some candidate foraging strategies.

19.5.1 Dynamic Foraging Game Model

To define the model of a dynamic foraging game, we explain each part of the dynamic game model in the last section.

State and Inputs

We have $N \geq 2$ foragers. The state $x \in \Re^{n_x}$ is composed of aspects of the foraging environment and the positions of the foragers in that environment. Assume that you have a two-dimensional foraging environment (a "foraging plane"). Extension to the three-dimensional case is straightforward. The position of the i^{th} forager is given by

The state holds information on the environment and foragers.

$$x^i(k) = \left[x_1^i(k), x_2^i(k)\right]^\top \in \{1, 2, \ldots, Q_1\} \times \{1, 2, \ldots, Q_2\} = F$$

with $x_1^i(k)$ its horizontal and $x_2^i(k)$ its vertical position on a discrete grid. Here, Q_1 (Q_2) sets the upper boundary for horizontal (vertical) movements. The variable F is used here to denote the set of all points in the foraging environment.

The decisions by forager i are commands to move itself to each of the cells that are adjacent to the current position, and which resource type to consume there. That is, at time k, so long as the movement is in the valid foraging region so that $x^i(k) \in F$, we have that $u_p^i(k)$ is in the set

$$\left\{ \left[x_1^i(k), x_2^i(k)\right]^\top, \left[x_1^i(k)+1, x_2^i(k)\right]^\top, \ldots, \left[x_1^i(k)+1, x_2^i(k)+1\right]^\top \right\} \bigcap F$$

which we will denote by $U_p^i(k)$. Here $U_p^i(k)$ is then the set of feasible moves at time k by forager i. The first element in the above set indicates that the forager should stay at the same location, the second indicates that it should move to the right horizontally, and not vertically, and so on (to all positions around the current one). Clearly, in this case, there are nine possible locations that any forager can move to at each step, *provided that* the forager is well within the region F. Disallowing movements outside the foraging region F is represented by the intersection with F. In particular, if a forager is at the upper horizontal boundary and tries to move to the right, we will say that it stays at the same place, thereby employing the basic idea of "projection" used in optimization theory. For convenience, we let $u_p(k) = \left[(u_p^1(k))^\top, (u_p^2(k))^\top, \ldots, (u_p^N(k))^\top\right]^\top$.

Assume that there are M resources that are indexed with the variable m. We represent the choice of resource by player i at time k as $u_r^i(k)$, $i = 1, 2, \ldots, N$, where

$$u_r^i(k) \in U_r^i(k) \subset \{1, 2, \ldots, M\}$$

represents the resource type m that forager i chooses to consume at time k, and $U_r^i(k)$ can be used to model the set of resources that it can choose from (extension to the case where each player can consume more than one resource at a time is straightforward). Define $u_r(k) = \left[u_r^1(k), u_r^2(k), \ldots, u_r^N(k)\right]^\top$. The decision $u^i(k)$ of forager i at time k is composed of a position choice and resource choice. In particular, we let

The input holds the decision of the foragers, where to go and what to do.

$$u^i(k) = \left[(u_p^i(k))^\top, u_r^i(k)\right]^\top \in U_p^i(k) \times U_r^i(k)$$

and $u(k) = \left[(u^1(k))^\top, (u^2(k))^\top, \ldots, (u^N(k))^\top\right]^\top$.

The distribution of resources is also part of the state. Extending the development of the static case, let

$$q = [q_1, q_2]^\top \in F$$

denote a cell in the foraging plane. Let z_i^m denote the effort allocation to consume resource m by forager i. Let

$$P^m(q) = \left\{ i : u_p^i = q, u_r^i = m \right\}$$

be the set of foragers that decide to go to position q to consume resource m at time k. Notice that $0 \leq |P^m(q)| \leq N$, but below, we will only use $P^m(q)$ for $q = u_p^i$ for some $i = 1, 2, \ldots, N$, so $|P^m(q)| > 0$.

We use the depletion rate α^m, $m = 1, 2, \ldots, M$, for the m^{th} resource. The amount of resource at time k of type m at cell q is $r^m(q,k)$ with $r^m(q,0)$ the initial distribution. The resources change over time due to growth (e.g. plants), weather, disease, farming, and foraging. For foraging, resources may diminish due to consumption, and in some cases, such consumption may result in the increase of other resources (e.g., since the resources may be living, so foraging influences their competitive balance). In other cases, foraging for one type of resource at one time may make it possible to forage for other resources later (e.g., if one forager eats one type of resource and this gives rise to other resources due to, for example, a forager leaving behind remains). Suppose that we consider the effects due to foraging where resources diminish according to

$$r^m(q,k+1) \quad = \quad r^m(q,k) e^{-\alpha^m \sum_{i \in P^m(q)} z_i^m} \tag{19.10}$$

for all $q \in F$. For this equation, notice that $P^m(q)$ is a function of u. Let

Forager actions affect the environment and hence, subsequent decisions.

$$x_p(k) = \left[(x^1(k))^\top, (x^2(k))^\top, \ldots, (x^N(k))^\top \right]^\top$$

denote the vector of places where the foragers are located. Let

$$x_r(k) = \begin{bmatrix} r^1([1,1]^\top, k) \\ \vdots \\ r^1([Q_1,Q_2]^\top, k) \\ \vdots \\ r^M([1,1]^\top, k) \\ \vdots \\ r^M([Q_1,Q_2]^\top, k) \end{bmatrix}$$

be a vector that holds a vectorized representation of the resource distribution (maps). The state of the game is

$$x(k) = \begin{bmatrix} x_p(k) \\ x_r(k) \end{bmatrix}$$

Foragers naturally operate somewhat independently of each other, at least with respect to some of their decisions and timing of actions.

Finally, we need to define how to generate the next state (to define f in the game model in Equation (19.6)). First, note that $x_p(k+1) = u_p(k)$, so that at the next time instant, each forager will have moved to the position that it was commanded to move to at the current time step. This represents that we are assuming no dynamics and kinematics for our forager (e.g., constraints on how fast it can move, turn, etc.), or at least, that the time scale is sufficiently slow relative to such physical phenomena. Second, note that $x_r(k+1)$ is defined via Equation (19.10). This completes the definition of how to generate the next state given the current state and the current input to the game; however, we still need to clarify issues related to the timing of decisions by the players.

First, we assume that the real time between k and $k+1$ is fixed so that the real time is $t = kT$, where T is a sampling period. Then, the real time at the next

sampling instant is $t' = kT + T$. So, we require that time proceeds according to a clock with a certain tick-length. This is necessary, due to how we model depletion of resources. Why? Because, it makes the effort allocation taken at each step for each resource by each player a constant as we had specified. (If we had random time lengths between decision times, then the fact that one forager makes a decision would affect the consumption rate of other foragers.) We have, however, still created a type of *asynchronous* game model, in the sense that if a forager does not make a move at time k, then it "chooses" its next position to be the same as its current one (the "null play" discussed in Section 19.4.2), so that it continues to forage at the same position. Additionally, the model allows for multiple (up to N) players to simultaneously take actions at each time step and the above formulas define how the state evolves with such simultaneous actions. So, our decisions occur asynchronously, but only at times given by the tick of some clock. If the clock period T is very small, then we can approximate fully asynchronous behavior.

Sensing and Outputs

The observations that forager i can make about the foraging environment at time k are denoted by $y^i(k)$. Clearly, the physiology of the animal constrains what sensing is possible. For instance, some animals can only sense via sampling chemicals in their immediate surrounding environment (e.g., certain bacteria), while others can sense light or sound and hence "see" for long distances. In terms of the mathematical representation, some possibilities for representing the feasible observations are the following:

1. *Full observations:* If for each forager, $i = 1, 2, \ldots, N$, and time k,

$$y^i(k) = x(k) \tag{19.11}$$

   then each forager can sense the distribution of all the resources over the entire foraging environment and the positions of all the other foragers at each time step k.

Sensing is never perfect, so foragers always make decisions under uncertainty.

2. *Resource observations and own position:* If

$$y^i(k) = h^i(x(k), k) = \begin{bmatrix} x^i(k) \\ x_r(k) \end{bmatrix}$$

   then each forager knows its own position and where all the resources are, but does not know the positions of the other foragers.

3. *Range-constrained sensing:* Let $S(q)$ denote the set of cell locations that a forager can sense resources in, or other forager positions, when it is located at cell q. This set can be used to specify characteristics of the sensing capabilities of the foragers. For example, suppose that foragers have constraints on how far they can sense resources that are independent of time and resource type (you could also make sensing range depend on

resource type and time). Then, as the forager moves, the set of cells that it can sense resources in changes. Suppose that this set of cells is defined via a circular region with radius R_s about the current location of the forager, provided that this sensing region is within the foraging region F. In this case, we could define

$$S(q) = \left\{ \bar{q} : \sqrt{(q-\bar{q})^\top (q-\bar{q})} \leq R_s \right\} \bigcap F$$

First, form a vector of the forager locations, for foragers that can be sensed, from elements of x_p, as $x_p^{s_i}$ with elements $x^j(k)$, where $x^j(k) \in S(x^i(k))$ for all $j = 1, 2, \ldots, N$. Second, form a new vector of the currently sensed cells from elements of x_r, as $x_r^{s_i}$ with elements $r^m(q,k)$, where $q \in S(x^i(k))$ for all $m = 1, 2, \ldots, M$. If

$$y^i(k) = h^i(x(k), k) = \begin{bmatrix} x_p^{s_i} \\ x_r^{s_i}(k) \end{bmatrix}$$

then the forager can sense resources in a region around its current location and it knows its own position, and the positions of the other foragers within its sensing range. If R_s is large enough, so that the forager can sense the whole environment no matter where it is in the environment, then this reduces to case 1 above.

Clearly, there are many other possible sensor models. For instance, sensing quality could depend on the resource, forager type, position in the environment, or time (e.g., to represent aging effects). The size of the sensing range may change over time. Different foragers may have different sensing capabilities.

Consumption, Energy, and Payoff to Foragers

Next, we must define the payoff for each forager. To do this, first define the amount of consumption of resource m by forager i, $i = 1, 2, \ldots, N$, at time k for a set of forager decisions u^1, u^2, $\ldots$, u^N, as

$$\begin{aligned} & C_i^m(u^1(k), u^2(k), \ldots, u^N(k)) \\ = \; & \frac{1}{|P^m(u_p^i(k))|} \left(r^m(u_p^i(k), k) - r^m(u_p^i(k), k+1) \right) \\ = \; & \frac{1}{|P^m(u_p^i(k))|} r^m(u_p^i(k), k) \left(1 - e^{-\alpha^m \sum_{i \in P^m(u_p^i(k))} z_i^m} \right) \end{aligned}$$

Notice that $|P^m(u_p^i(k))| > 0$. The factor $\frac{1}{|P^m(u_p^i(k))|}$ is used to represent that if there are $|P^m(u_p^i(k))|$ foragers at location $u_p^i(k)$ foraging for resource m at time k, then the returns from foraging are split evenly among those foragers. (Other definitions of splitting the resource returns could represent more capable foragers winning more returns when they forage next to some less capable forager.)

The cost due only to consumption for one move is, given that forager i has priority p_i^m for resource m,

$$J_{ic}(x(k+1), x(k), u(k)) = -\sum_{m=1}^{M} p_i^m C_i^m(u^1(k), u^2(k), \ldots, u^N(k))$$

Each forager must expend energy to forage, and we define this via

$$J_{ie}(x(k+1), x(k)) = w_{ie}\left(x^i(k+1) - x^i(k)\right)^\top \left(x^i(k+1) - x^i(k)\right)$$

where $w_{ie} \geq 0$ sets the amount of energy needed to move a certain distance. We assume that energy is independent of resource type being sought and consumed. The "danger" aspect could be modeled as in the last section, but we ignore this possibility here. Our total payoff to forager i at time k is

$$J_i(x(k+1), x(k), u(k)) = J_{ic}(x(k+1), x(k), u(k)) + J_{ie}(x(k+1), x(k))$$

If there are N_s steps in the game, we have a payoff $J_i^{N_s}$ for playing the entire multistage game as given in Equation (19.9). Each forager wants to minimize $J_i^{N_s}$ and thereby maximize consumption with minimal energy expenditure. This can require considerable finesse, as it may be a good strategy to give up payoffs at some points in time in order to realize more benefits at some other later time.

Information Space and Strategy Design Challenges

Designing a strategy involves picking the information space $I^i(k)$ and strategy G_k^i, and of course, there are many possibilities. Here, for the remainder of this section, we pick one simple approach and invite you to consider others. Suppose that Equation (19.11) holds and each forager only uses $y^i(k)$ so

$$I^i(k) = Y^i(k)$$

and we need to choose G_k^i where $u^i(k) = G_k^i(x(k))$. This corresponds to allowing each forager to observe all forager positions and resources in all cells at each time k. The forager does not, however, have memory so it cannot store, for example, sequences that characterize the pattern of resource depletion or motion of the other players (which could be useful in estimating the intent of other players).

Who knows what and when they know it significantly affects distributed decision-making.

Recall that the information space is also defined via specification of a communication network between the players. In an adversarial game, there may be no communications or communications may be present, but the adversaries may try to mislead each other so that they can gain more themselves. However, in many cooperative games, there may be significant sharing of information over the network. Recall that the network may be defined via a communication topology that says who can communicate with whom. (Think of the topology as a directed graph with nodes as the foragers and arrows pointing from any forager to a forager that can receive or sense information about it.) There can be bandwidth constraints for the communication links, or random but bounded

delays in transmitting/sensing information. The topology and communication network characteristics may depend on the locations of the foragers (e.g., if two foragers move too far away from each other, then their communication link may get "broken" and, if two other foragers get close enough to each other, they may be able to "create" a communication link). Moreover, activities of the group of foragers may dictate how the topology should be configured and dynamically reconfigured. Here, we will assume that the communication topology enables the sharing of sensed information for $I^i(k)$ above. This will allow each forager to compute all the decisions of all the other foragers, so that they do not need to share information on $u(k)$.

19.5.2 Biomimicry for Foraging Strategies

Biomimicry may provide a way to define practical (computationally feasible and scalable) distributed and cooperative controllers for autonomous vehicles.

Here, we introduce the idea of using biomimicry of foraging strategies found in nature.

Rules, Planning, Learning

How do we define G_k^i? There are many approaches. One, which may correspond to how simply organisms find food in some environments, would be to use simple "rules" to search for and find food. For instance, such rules may use environmental cues to tell them where to move to be likely to find food.

Another more sophisticated approach is to observe the environment and use past information about how a typical environment holds food, and then to use this model in a planning strategy. The traditional approach in game theory in engineering is in fact to assume some ordering for player decisions and to then use dynamic programming to find optimal paths for N_s steps. Clearly, this can be computationally prohibitive, especially for high values of N, Q_i, L, and N_s. So typically, one approach is to use a "receding horizon" controller (i.e., a planning system), where you use a (perhaps simplified) model of the plant and simulate ahead in time, find the best input sequence, implement the first decision in each sequence, and then repeat. Clearly, this can also have computational problems, except perhaps for the case where you look ahead only one or two steps, or where the model used to simulate ahead in time has appropriate simplifications that reduce complexity, yet still lead to good decisions. (It is generally quite difficult to balance these objectives to get a good simplified model.) This is, moreover, the common approach that has been investigated in many contexts for many problems in the past (e.g., for linear systems with quadratic costs, "model predictive control" (MPC), and more general decision-making systems).

Some animals actually learn the strategy of their opponent, then take appropriate actions in order to optimize their gains. For this, they may have a model of a typical opponent and then observe their actions to guess what type of strategy they are currently using. Then, if they assume that the opponent will not switch strategies soon, they may be able to work more effectively against the opponent.

A Generic Saltatory Strategy

In order to be more concrete about biomimicry of competitive and cooperative foraging, suppose that we want to model "saltatory search" and foraging, where an animal alternates moving and thinking about where to move next. This takes into account physiological constraints for many animals, where they alternate between moving and stopping to sense and decide where to move next.

The set of steps we use to model a "saltatory strategy" is the following:

1. Play a static matrix game of the type studied in the last section at $k = 0$ to determine where each forager should seek to go. Call the resulting goal position $x^i_*(\bar{k})$, where $\bar{k}$ is the index of the times when the forager stops to sense and decide where to go next. Hence, the game played at $k = 0$ results in $x^i_*(0)$ for all $i = 1, 2, \ldots, N$ and these specify the first set of goal positions that the foragers try to move to. (Issues in path choice from the current forager position to the goal position are discussed below.)

2. If player i gets to $x^i_*(\bar{k})$ at time k', it plays a matrix game with all other players even if they did not get to their goal positions that time. This gives us $x^i_*(\bar{k}+1)$, the next set of goal positions. This can result in some foragers switching from one goal position to another if they evaluate that to be profitable.

3. Go back to step 2 if the termination condition (e.g., N_s time steps) is not achieved.

This is a type of "asynchronous" saltatory strategy, where the time it takes between decisions in the m index depends on how far it decides to move each time. How do the foragers move from $x^i_*(\bar{k})$ to $x^i_*(\bar{k}+1)$ (i.e., what path do they take)? This depends on many factors. If the foragers can sense and make decisions during movement, then they may try to move towards the new goal position along a path that will maximize their consumption. Depending on the forager's goals, it may be willing to make significant deviations from a straight-line path between $x^i_*(\bar{k})$ and $x^i_*(\bar{k}+1)$, where the goal is simply to minimize energy consumption to get to the goal position. For instance, the forager may compute a type of optimal path, one that minimizes energy consumption while maximizing resource consumption, between $x^i_*(\bar{k})$ and $x^i_*(\bar{k}+1)$, and thereby obtain more resources (i.e., it tries to do some consumption along the way to its goal cell). It should be clear that overall this strategy could be viewed as a type of planning (or receding horizon) strategy, since it does involve looking ahead in time; however, it is only using very simplified information to decide successive goal positions. Moreover, extension to the case where it looks ahead more than one step across the $\bar{k}$ index should be clear.

The type of game that is played at the times $\bar{k}, \bar{k}+1, \ldots$ depends on what information is used to play the game. For instance, a forager may inherently know that another forager will go to some region and then spend significant time there, since it will minimize its energy expenditures by staying in that region for some time. A forager may then try to "keep its distance" from another

forager and pick a "foraging region" rather than a point (of course, this depends on the physical dimensions that correspond to our cell sizes). How can we represent this? One approach would be to define an "abstraction" of the resource distribution part of the state, where elements of the abstraction correspond to, for example, sums of resources of a certain type in sets of contiguous cells. Then decisions about where to go can depend on "super-cells" created by the abstraction. This will result in a computationally simpler game, since there are fewer super-cells to consider moving to. Moreover, it is possible to define a "nested" strategy where once a region is chosen, a game is played between all players that decide to go to that region (and multiple levels of abstraction, and hence, nested games can be played).

It is also the case that some types of foragers know what type of separation to keep with other foragers, especially in the case of cooperative (social) foraging. They do not want to be too close, so they crowd each other and each forager does not get enough resources to survive, but they do not want to be so far from each other that they cannot benefit from communicating with the other foragers so they can be led in the best directions towards the most profitable sites.

Coping with Complexity: Space and Time Abstractions

So, what are the basic concepts employed in foraging in nature that allow animals to overcome computational complexity in deciding how to forage? First, there is the prevailing fact that while foragers do not want to die, the overall species has extraordinary reproductive capability so if they do die, they can be replaced. This is a basic fact of life, but it may not have too much relevance in the case, where we use biomimicry to design automated systems in engineering (e.g., for cooperative robotics). Second, evolution essentially generates practical and robust foraging strategies by optimizing them in the face of complexity constraints (e.g., forager physiology that dictates how much memory it can have). Again, however, at the present time in engineering, it is often impractical to use such a fact. At other times, such as in the area of "evolutionary cooperative robotics," researchers try to evolve good behavior over time. Certainly, there may be the possibility that such evolution could take place a priori and in simulation rather than in actual hardware. Regardless, the problem is that there may be little guidance on how successful such an evolutionary foraging strategy design approach will be.

We cope with complexity via hierarchies and space or time abstractions.

Here, we simply assume that we will observe *existing* foraging strategies and use biomimicry to capture the essential principles of their operation that help the forager cope with complexity. While it is clear that there are many principles used to cope with complexity, and that it may be difficult to observe the basic principles used to cope with complexity for any given species, here we will focus on the principles that seem to arise when one studies how saltatory strategies operate as we discussed. There seem to be at least the two following general principles:

- *Spatial abstraction:* Decision-making often involves having animals "group"

aspects of the foraging environment in order to make decisions. For instance, an animal may look in a few directions and pick the general direction that seems to have the most resources. In this way, it avoids being too greedy by favoring resource "peaks" that may have few resources around them, and hence, does a type of prediction since it knows that it will continue to forage and try to minimize energy consumption by not moving out of a region.

- *Time abstraction:* Next, there is a type of "asynchronous time decimation," where the animal does not decide what it will do at each small time step, but only where it should be (or what it should be doing) at certain future times. As they move toward their current goals, they solve the problem of what to do along the way.

These seem to be fundamental principles driving the design of practical foraging strategies, and how to cope with complexity in decision making.

Finally, note that complexity presents significant challenges in distributed decision making. Even for the simple strategy defined in the past section, complexity can be significant, especially for large N. For instance, for a full information cooperative game, there are three decision variables (horizontal and vertical position to move to and nutrient to seek there) for each of the N players and so computing the cost involves finding and computing an optimal value for a very large matrix game. (How large is the matrix for the case described above?)

19.6 Challenge Problems: Intelligent Social Foraging

Multiple vehicle guidance and decentralized decision-making problems provide nice classes of applications to integrate the methods of this book.

Bacteria forage according to relatively simple rules that dictate how they climb up nutrient surfaces, or aggregate for the purpose of survival. Their biochemical "brain" is very simple, essentially a bag of molecules where chemical reactions implement foraging "decisions." Recall that we assume that there is a "cognitive spectrum" of intelligence in making foraging decisions. For instance, some higher animals have central nervous systems and via these they can achieve planning, attention, and learning. We can think of such animals as "intelligent foragers." Intelligent foragers typically also have an ability to communicate so that they can achieve "social foraging." For instance, they may work together as a group to improve their foraging success and survival chances. Some animals are of relatively low intelligence, but enhance their foraging success with communications (e.g., bacteria and ants) to gain an "emergent" intelligence for the group. Other animals have significant intelligence but may not exploit their communication capabilities to a great extent, since a "loner" approach in foraging in their environment is more successful.

Here, we first focus on intelligent individual (i.e., nonsocial) foraging by explaining how to use planning, attention, and learning methods for foraging. To do this, and in order to be concrete, we discuss yet another nongradient optimization method that is based on the use of "surrogate models" (in a foraging

problem, a model of the foraging landscape that includes information about where predators and prey are). We challenge the reader to solve a particular type of intelligent foraging problem with a surrogate model method, by presenting it as a "design challenge problem" for students, where they can integrate earlier methods from this book. (For this problem, there may or may not be multiple foragers and competition.) Next, we discuss the social foraging problem by introducing it as a second design challenge problem, which also requires the use of coordination (and hence communications) among multiple agents. The main challenge is to implement distributed rule-based, planning, attention, or learning strategies in order to coordinate the behavior of a group of foraging agents. Varying levels of intelligence, distribution, and communications are discussed. Aspects of competition are discussed, and in general, there are multiple teams of foragers that compete for various resources in the environment.

Finally, note that other problems not related to foraging could be used in place of the problems defined in this section to provide a challenge problem for the student (e.g., process-wide control/automation for a factory).

19.6.1 Intelligent Foraging

In this section, we explain how to develop a nongradient optimization method that relies on planning, attention, and learning and hence, provides an algorithmic approach to intelligent (nonsocial) foraging. This is only one of many possible methods that can be envisioned for the solution of this problem. We discuss intelligent foraging in the context of this method simply to be concrete about the ideas, the connections to optimization, and how a simulation might be constructed.

Surrogate Model Method Representation of Intelligent Foraging

A surrogate model method simultaneously learns an approximation to the cost function and uses it to guide where to search the cost function.

For some optimization problems, it is not only the case that it is impossible to compute or know the analytical gradient, but it can also be the case that it is *very expensive* to compute a value of the cost function for each point in the optimization space (e.g., if the computation requires extensive simulations using a very complex model, use of experimental apparatus, or physical rearrangement of physical elements in an environment to determine the value). One approach to such problems is to use a "surrogate model" to represent the cost function. The idea is that each time you compute a value of the cost function, you use the pairing between the test point in the optimization domain and the cost function value that is computed to form a training data pair, and then construct an approximator for the cost function (perfectly analogous to how we did this in Part III, but here the focus is on learning the mapping implemented by the cost function). How could an approximation of a cost function be useful? The key is to note that it can be difficult to compute points on the actual cost function, but it can be very easy to compute test points on the approximation to the actual cost function. (If the approximation is good, the values will be close.) Moreover, if the approximation is reasonably good, it can suggest points that

are good candidates for testing on the actual cost function. Remember the response surface methodology of Section 15.2, where we approximated a cost function with an approximator and used it to pick an optimal design point. The difference here is that our surrogate model method will operate in real time via sequential acquisition of information (not in the "batch-mode" that RSM typically uses).

The surrogate model method proceeds as follows:

1. Pick a test point (or set of test points) for J and compute J at this point (these points). Note that the method can be "set-based" so that it computes in parallel the cost function at several test points (e.g., via parallel processing).

2. Store the pairing(s) between the test point(s) and value(s) of J in a training data set G for an approximator f for J.

3. Construct an approximator (interpolator) for the data in G (perhaps removing some points as others are added). This approximator retuning can be achieved via repeated application of recursive least squares over a linear in the parameters approximator, or via application of a Levenberg-Marquardt method to training a nonlinear in the parameter approximator.

4. Perform an optimization *over the approximator surface* (not the cost function) to find a minimum point on that surface. (You may use gradient methods or pattern search methods to perform this optimization.) Call this a new test point, compute J at this point (and for a set-based method, perhaps at a pattern of points around it), and add this (these) to the training data set. Go back to step 3.

You can think of this as constructing an approximation of the cost function to guide you to make choices about where to explore the cost function to find the minimum (the "search" proceeds via the optimization over the learned surface with periodic updates via sampling the actual cost function and updating the approximation surface). You can think of the optimization process over the approximator surface as providing a strategy for picking points to include in the training data set G. Clearly, when applied to specific problems, the strategy for picking points to include in G may be constrained by the problem at hand (e.g., if applied to a foraging problem, the surrogate model may be a representation of some aspect of the environment and you may be constrained in choosing candidate points on the surrogate model by how fast the vehicle can move and what sensing resources it has—e.g., whether it can sense at a distance).

If you use a pattern search method, it may make sense to think of points on the pattern as predictions about J, and the selection of points as a selection of a plan, and the approximation process as learning. In this way, you can think of the surrogate model method as defining a class of learning-planning methods, where it is possible to choose a whole variety of pattern search methods as the basis for planning and gradient optimization methods as a basis for learning (of

A surrogate model method can be thought of as a method for integrated learning and planning.

course, as pointed out in the last section, it is possible to use the pattern search methods as a basis for learning).

Example: Intelligent Foraging Over Nutrient Surfaces

Here, we provide a brief explanation of one way to define aspects of the environment as a cost function that can then be used in a surrogate model method to emulate intelligent foraging.

- Define a nutrient surface, analogous to how we did it for bacteria. This is the surface that we want to learn about and plan over.
- Add an attentional map defined over that same domain that simply says where we have searched and where we have not (so if we have visited one region, then change the map to indicate that, and make it so that climbing down the attentional map surface corresponds to looking in unexplored areas; in the theory of surrogate optimization, this is sometimes called a "merit function").
- Add the nutrient surface to the attentional map and call that the cost function. Use planning over the currently learned map so that there is a type of look-ahead for the forager to decide where to move; however, it can only make its decisions on the learned map, not the actual one (but it does know the entire attentional map). This way, in seeking to minimize the cost, it will try to achieve competing objectives: (i) try to find the lowest point which corresponds to good food, and (ii) try to search the entire surface (as dictated by the attentional map). It balances a desire to search far and wide, possibly finding a better food source, with the desire to eat now. The attentional map helps it to avoid getting stuck in a local minimum.

What are the key elements to coding this? First, it seems logical to use a set-based method, some pattern of sensed points of the nutrient cost function, placed around the current position of the forager. Second, vehicle dynamics should be kept simple but must be present in some form (e.g., it cannot be that you can move the vehicle in one time step an arbitrary distance across the optimization domain—this is a key difference from standard nongradient optimization methods). One way to model this is to use a "momentum-term" (see gradient methods) in the optimization algorithm update formula. Third, it seems logical to use a linear in the parameter approximator for the nutrient function, perhaps with RLS to compute the approximator update at each step (so run RLS for each point in the pattern, at every time step). Fourth, an RBF could be used for the attentional map, with a simple strategy to update it based on where the forager actually visits (e.g., it could simply change the map to represent that the pattern of sensed points was there). Fifth, we see then that planning corresponds simply to the optimization over the surrogate model (which here would be the approximation of the nutrient cost function, plus the attentional map), and since you want the planning method to consider

many directions from the current forager position, it seems logical to try a pattern search method (e.g., simple coordinate search or multidirectional search over the combined approximator/attentional map). It should be possible to show a movie of learning a nutrient map, the updates to the attentional map, and it should be that these maps would give good insights into design of the strategy. How computationally complex will the method be? Well, this depends on the resolution you choose for your approximator for the nutrient map, and the attentional map. Also, it depends on the optimization method that you use for the approximator surface, how big the set is in the set-based method, and how long your planning horizon is.

19.6.2 Intelligent Social Foraging

In this section, you are asked to consider the wide range of possibilities for how to design the control and guidance algorithms for automating a group of intelligent social foraging *vehicles*. Hence, this section serves to specify a "capstone" design problem for this book that challenges the student to integrate the various methods to solve a particularly challenging problem. The design challenge problem of the previous subsection is relatively simple compared to the one discussed here (and is included as only a part of the more general problem here).

The intelligent social vehicular foraging problem also provides a glimpse into potential topics for further investigation. For instance, there has been little discussion on the relevance of language and communications in groups of intelligent systems, let alone aspects related to learning language. There has been little discussion of distributed learning by groups, distributed planning, distributed attention, etc. Moreover, there has been little discussion on learning and evolution of the *structure* of controllers and estimators. This challenge problem provides a framework to study such issues (of course, you may want to start by more thoroughly studying each of these topics in isolation, before confronting the more challenging social foraging problem).

The main problem for the student will be how to even attack this problem, provide a solution that shows you understand the basic methods of this book, and yet integrate the methods to solve a meaningful problem. This is a design problem where you help design the problem! Perhaps your instructor will help you, but this still makes the problem more challenging, since your objectives will not be explicitly listed. You will have to invent them, and the very design of the objectives is something you will be graded on. You should be careful to adopt a scientific approach to solving this problem. You should not simply construct some ad hoc combination of earlier methods, so that you can get lucky in simulation to show that it works. You must evaluate the methods fairly, provide biological motivations if appropriate (and if you like doing that), and it would be especially nice if you can augment your study with mathematical analysis that verifies the operation of the system (e.g., in the spirit of the stability analysis that we have studied for neural/fuzzy control, attentional systems, adaptive control, or swarm cohesion). If you have the opportunity to implement

your methods on a group of vehicles, this can provide another way to study the validity of your approach. Keep in mind that standard engineering/scientific principles apply here, as they were discussed in Part I.

The problem statement, which is very simple, is given in Design Problem 19.5. So, what is the first step to solve this problem? Read this section as its basic focus is to give you ideas on how to integrate methods and concepts from this book to specify decision-making strategies for foraging. It may also give you ideas for how to extend some of the methods in the book, and you may be particularly attracted to doing so for a method that you were particularly intrigued with. Next, study the current literature. See the "For Further Study" section at the end of this part for some ideas on where to start; however, you should search the library or Internet for other current literature. Next, see the Web site for this book where some relevant literature and ideas are posted. Finally, work hard and have fun!

Vehicles, Environment, and Objectives

Here, we define the challenge problem.

Groups of Vehicles—Dynamics, Communications, and Control Structure: The first challenge is to define the type of vehicle that you will use. In particular, you need to define the following:

- *Vehicle dynamics:* For instance, if you use an automobile as your vehicle, what are the differential equations that you will use to simulate its motion? You can choose the vehicle type. It could be any type of autonomous land (e.g., automobile, truck, cross-country), water (surface or underwater), air (e.g., helicopter or airplane), or space vehicle (e.g., mobile satellite, explorer vehicle, etc.). You probably want to pay attention to the complexity of the dynamics. If you use extremely complex dynamics, and later try to define a sophisticated control strategy with many vehicles, your simulation may be too computationally complex. Hence, it is likely that you will want to use simple point-mass dynamics for your vehicle, and perhaps saturations on turn rates and velocity.

- *Vehicle sensors/actuators:* You will need to define which sensors and actuators you need for "inner-loop" control (e.g., in order to force the vehicle to track a trajectory that it chooses to follow; for example, doing heading regulation for the tanker ship). Moreover, you need to define what it can sense about its environment (e.g., types of prey, elevation of the earth, motion of predators, etc.), and how it can change its environment (e.g., by killing a prey, or building a bridge to be able to cross a river).

- *Vehicle communications:* You need to define the characteristics of the communications that each vehicle is capable of. Do they all have the same capabilities? Are the communications noisy or bandlimited? Are there

random but bounded communication delays? Are the communication capabilities range-limited, so that if one vehicle moves too far away, it will not be able to communicate with some other vehicles (e.g., simply due to the distance, or possibly due to being behind some obstacle)?

- *Hierarchy and distribution in the group of vehicles:* Building on the last point, you need to define the allowable communication channels between vehicles, whether the structure of these channels (i.e., the topology of the interconnections between all vehicles) can change over time, and whether there is a type of hierarchy where some vehicles command others to perform tasks. For example, there may be no leaders in the group and all the vehicles may be able to communicate. Perhaps there is a single leader who is not endowed with any special communication capabilities, but who may behave differently. Perhaps there is a leader with special communication capabilities, multiple leaders with different objectives, or a hierarchy of leaders and followers for command and control. In some problems, you may be able to design the hierarchy or change it during operation of the system, and in others you may be given the hierarchy and have to work with it with no changes.

What types of vehicles are actually used in groups to achieve some objective? While there are clearly military applications, there are also commercial ones. For example, an "automated highway system" can be viewed as a group of autonomous vehicles that operate within a type of command and control structure (see Figure 1.13 on page 40). Or, groups of autonomous vehicles could be used in pollution clean-up, farming, exploration, or inventory control in manufacturing systems.

Environment Model and Goals: Your vehicles need to operate in some environment and hence, your simulation will need to represent its characteristics. For instance, you may want to consider the inclusion of the following:

- *Media:* What media do your vehicles move through? Air, water, outerspace? Are there disturbances that affect the motion of the vehicle? Solar pressure? Wind or water currents? For land vehicles, are there hills and roads?

- *Predator/prey (or noxious substance/nutrient) characteristics:* If your vehicles seek to consume prey while avoiding predators, what are the characteristics of your predators and prey? How fast can they move? What types of evasive or pursuit strategies do they use? What is their energy value if they are consumed? What is the probability that you will encounter them? What are the predator/prey densities? You could set this up as a *game*, where two students design teams of predators to operate in a single environment and compete for food sources. Alternatively, each predator could be prey for the other. Regardless, other predator and prey strategies help to define the environment for a set of vehicles.

- *Environmental changes:* Do the characteristics of the environment change over time? Do the characteristics of the media change? Does predator/prey density change? Do predator and prey strategies change?

What is the overall goal of the group of vehicles? The goal may involve characteristics, such as the following:

- *Energy consumption:* Find and ingest as much energy as possible (in the form of prey or nutrients of some type), while avoiding getting killed and eaten by some predator.
- *Achieving goal positions:* In some problems, the goal is simply for the group of vehicles to navigate their environment and achieve some goal positions.
- *Gathering information:* The group of vehicles may simply want to create as accurate a picture as possible about the environment that they operate in. For example, such an objective may be useful in space exploration via a set of vehicles.
- *Changing the environment:* There may be a collective goal to modify the environment that the group of vehicles operate in. For instance, in cooperative robotics, the problem of how to use a group of autonomous vehicles to move an object has been studied. For other problems, there may be a desire to eliminate targets, cultivate land, build a home, or gather food into a certain location.

It should be clear that there are certain principles that govern trade-offs in achieving goals. For example, typically the desire to achieve a wide-area search (e.g., to find prey) competes with a desire to focus activities in a single local region (e.g., in consuming prey). You should focus on uncovering such fundamental principles/trade-offs and illustrating them via simulations.

As an example, in the IVHS application the media is air, friction with a road that may turn, wind can influence dynamics, and there are hills and valleys. Temperature, snow, and rain may also affect vehicle dynamics. Destinations can be thought of as goal positions.

Elements of Distributed Decision-Making

While each of the foraging vehicles could have neural networks that implement various control functions, and hence, there would be a distributed neural network for instinctual control, here, we discuss the cases where groups of vehicles share information and implement distributed rule-based, planning, or attentional schemes for cooperative control that are not necessarily implemented by neural networks. In a sense, this section indicates how methods of Part II can be extended to the social foraging problem. Hence, the focus is not on use of the methods there for the development of controllers for a single control loop, but how to coordinate the use of information to meet the objectives of the group of

vehicles (for "outer-loop" control, or guidance) that possibly has an opposing team (or teams) of vehicles.

A key component of the problem of foraging for many organisms is the search for food, and hence, this can be a key component for the case where groups search for food. Hence, in all the elements of decision-making, there is an element of distributed search. There are, however, several other key components including cooperative identification of prey, cooperative avoidance of predators, and cooperative attack of prey. The approaches outlined below show different ways to look at these basic elements of the group foraging problem.

Distributed Rule-Based Foraging: Some organisms (e.g., ants) actually use simple rules to specify how to forage for food, and when taken together, a group of such organisms seems to have an "emergent intelligence." For example, such rule-based behavior can indicate when to move in certain directions, and when all the organisms follow such simple rules, the group appears to move with purposeful behavior, acting as if they were a single organism rather than many individual ones. The key difference, compared with the rule-based systems in Part II, is that communications with other organisms are possible. Rules basically have two parts: antecedents and consequents. Each of these can be different for rule-based cooperative control in the following manner:

- *Using neighbor's information in rule antecedents:* The type of information that arrives for use by each organism depends on what type of communications the organism is capable of achieving. How should the information from other group members be used? Sometimes it simply would be used as an additional term in the antecedent of the rules for behavior of an organism. For instance, without communications, an organism may simply move greedily about looking for food so its rules' antecedents simply depend on direct sensing of environmental variables. If there are communications from neighbors, this may modify the behavior. How? It could be there are rules that indicate that the organism is supposed to look for food, but also follow its neighbors. If such a desire is followed by many organisms in a group, swarm behavior may emerge (individual rules can lead to interesting higher-level emergent patterns of behavior). Notice that in this case, the organism is using rules that depend not only on the environment, but also on communications from neighbors (e.g., a rule might say to move towards food, if you do not crowd a neighbor).

- *Rules for sending information to neighbors:* Rule consequents may contain not only information about which direction to move to get food, but also a specification of what to communicate to your neighbors about your experiences, actions, or goals. For instance, if a forager decides to pursue some prey, it may send a signal to some of its neighbors to come help it (e.g., in the case of some fish when they try to kill and ingest a much larger animal). Alternatively, an organism may communicate its intent to search some region for food and rules in other organisms may then trigger

to indicate that they should not also look there (e.g., based on expected prey densities).

This provides a simple introduction as to how rules could be used in cooperative control (e.g., we did not discuss the fact that inference strategies could be communicated and shared between organisms). It is important to emphasize that even a set of simple rules implemented on each organism (identical or different rules) can lead to seemingly intelligent emergent behavior. That is why, by working together in simple ways, great things can be achieved. In this case, it can be that they simply enjoy greater foraging success. But, it is important to recognize that even what are thought of as higher-level cognitive capabilities can be achieved as the result of many simple communicating organisms (e.g., think of trail-laying by ants as implementing a type of learning to improve the success of the colony of ants). Moreover, note that rule-based strategies may be employed in hierarchies, and in conjunction with the planning and attention methods discussed below.

Distributed Planning for Foraging: You should think of distributed planning as an advanced form of distributed rule-based cooperative control, where models are used for prediction, and optimization is used for plan selection. To fully exploit the capabilities of distributed planning you may need higher bandwidth communication channels, since you may want to communicate models, plans, or plan selection strategies between organisms. The following show some ideas for how to achieve distributed planning:

- *Sharing models:* While operating independent of others, an organism that employs planning would use its own model to decide the best way to forage. In a cooperative strategy, an organism could get model information from other organisms in the group. For example, the model may indicate where other organisms have searched or what they have found. Sharing of model information between organisms essentially results in a type of adaptive planning, since models can be updated online while the planning by a single organism that uses that model is taking place. (Again, we see that learning emerges via the cooperation.)

- *Sharing plans or sets of plans:* It is also possible that an organism may share its current plan, or set of possible plans, with other organisms in the group. These may indicate where it intends to move, or the set of possible places that it intends to move. Or, it may indicate its strategies for attacking a prey or avoiding a predator.

- *Sharing plan selection strategies:* The cost function that is minimized in order to select plans could be communicated to other organisms to indicate aspects of the planning strategy that an organism is using (e.g., to indicate its intent to try to minimize the use of a certain resource).

Clearly, distributed planning can become very complex and complex behavior can emerge from even simple planning strategies. It should be clear that

hierarchical planning strategies may be useful, with similar sharing of information between planning strategies, and the possibility that subordinates execute steps of plans specified by higher-level organisms.

Distributed Attention for Foraging: If one organism is given the task of attending to a set of mobile objects, it allocates its cognitive resources by dynamically refocusing its attentional focus (and perhaps the field of view of its sensors). Suppose that we want a group of social foraging vehicles to attend to a group of mobile predators/prey. Consider the following approaches to this problem:

- *Distributed agreement on focus regions:* One approach is to try to develop a strategy to divide the region into subregions and have each organism focus only in that subregion. This is, however, a difficult problem since the foraging group can move and there may be less than perfect communications between the organisms. This can result in one organism being responsible for attending to a subregion with too many mobile predators/prey to cope with (i.e., the capacity condition for that organism may not be satisfied so that there is no way it can succeed in its task). Or, it could be that there are too many organisms focusing on one subregion so that their cognitive resources are essentially wasted.

- *Leaders and hierarchical strategies:* For the above case, our intent was to consider a predator/prey focused on if *any* organism focused on it. This did not require global communications. What can we achieve if we have global communications? First, this enhances the possibilities to allocate *organisms* to groups of predators and prey. (We think of allocating whole organisms that in turn allocate their cognitive resources over time to attending to certain predators/prey; notice that the global communications enables the implementation of a type of hierarchy in the group's attentional strategy.) Second, this enables the group of organisms to construct a composite "snapshot" of its predator/prey environment for use in making foraging decisions.

Clearly, it may be possible to enhance the effectiveness of the attentional strategies with rule-based, planning, and learning capabilities (e.g., via distributed adaptive model predictive control as an attentional strategy). We will discuss distributed learning in more detail in the next section.

Distributed Learning

This section indicates how methods of Part III could be used to augment the strategies in Part II for use in the social foraging problem. The focus here is again on the distributed implementation of the methods, where special problems arise due to, for instance, the lack of global information.

Distributed Learning in Groups of Foragers: Learning can affect all aspects of each of the strategies discussed in the last section, from the functionalities that focus on the search for prey to ones concerned with cooperative avoidance of predators. Learning should be thought of as gathering and storing information to change future behavior. Hence, rather than try to list all the ways that learning can be used in social foraging, we will focus on what types of information can be learned, and why it might be useful in enhancing foraging success. Some ideas include the following:

- *Learning characteristics of the environment:* Individuals or the group may try to learn as much about the environment as possible. For instance, they may remember locations and quality of food sources, evasive maneuvers of prey, attack patterns of predators, etc. Learning of these characteristics may be facilitated by certain instincts in each of these cases (e.g., built-in expectations about food distribution and density or understanding typical rates of movement of predators and prey).

- *Learning foraging strategies from other group members:* It may be possible for one forager to cooperatively forage with other foragers and at the same time learn how its neighbors (or predators) succeed at foraging to improve its own performance, and at the same time improve the performance of the group (i.e., it may obtain gains that are not at the expense of others in the group).

- *Learning how to communicate:* It may be possible that a forager may learn how to communicate with the group to enhance foraging success.

To make these ideas more concrete, we briefly discuss a specific example of how learning can be integrated with planning and attention in foraging.

Distributed and Integrated Planning, Attention, and Learning: You can imagine that there are many ways to combine learning, planning, and attention to achieve effective group foraging strategies. Here, only one is outlined, essentially expanding a bit on the concepts in Section 19.6.1. You could certainly generate many more; in the next section, we will discuss the issue of ranking the quality of such proposed group foraging strategies via an evolutionary perspective. Here, we ignore the quality of the resulting strategy and simply outline a *way* that what you might call "intelligent" social foraging could be achieved.

Suppose that we have a group of foragers searching on an (x, y) plane for nutrients. Consider an individual forager. Suppose that this forager has a dynamical "attentional map" that indicates, for instance, where the forager has not looked for food (i.e., where it needs to pay attention in case there is food present), and where it has (and hence, where it may have found food and need to have a mechanism to track it). Corresponding concepts work for predators. The attentional map amounts to a type of memory, and hence, its dynamic updating corresponds to a type of learning. The forager can use this map to try to focus its attention in the proper way to make sure that it has as accurate a view as

possible of the predator/prey environment, where this process is basically driven by the inherent goal of survival and reproduction.

Next, suppose that the forager maintains a "cognitive map" of its environment that it learns while moving throughout the environment. It stores information about physical characteristics of its environment (e.g., locations of rivers or forests), and perhaps also the location and characteristics of predators and prey. Next, suppose that we endow our forager with an ability to plan using both its cognitive map of the environment, and its attentional map. That is, think of the maps as constituting a type of model of the foraging environment that is learned during the foraging activity, or during the lifetime of the forager. Now, with a planning capability, the forager can use its current best information about the environment in order to project into the future and pick the best way to forage (e.g., it may predict how a prey will react to its movements or how a predator may behave). Clearly, this predictive capability depends critically on the quality of the learned information (and hence on the *process* of learning), and its own abilities to simultaneously consider a large number of possible predications and responses by the environment and predators/prey (e.g., the amount of memory and computational throughput of the forager directly constrain the plan generation and selection process).

Now, suppose that each forager has the attention, learning, and planning capabilities, and it has a certain type of ability to communicate with its neighbors. For example, suppose that each forager can communicate its own attentional and cognitive map to any other forager that is within a fixed distance from its current position. Can the group achieve more effective foraging? This seems quite plausible, since foragers will generally have more accurate attentional and cognitive maps without expending more energy (i.e., if the communications are cheaper than gathering information independently). This improved information should directly affect the quality of the attentional strategies and predictions made in planning, and hence, the quality of the foraging decisions that are made. It should be clear that if you increase the allowable range of communications, there should be corresponding improvement in the quality of information in each forager, and hence, an overall improvement in the quality of foraging by the group (assuming of course, that intergroup competition does not dominate). Clearly, it is also the case that if a forager has a poor sensor or is not honest, then if this information is passed to other team members, the overall foraging success could degrade. Indeed, when there are two teams, a key strategy may be how to deceive the opponent so that they make bad decisions.

Evolution of Foragers

In this final section, we make a few remarks about the relevance of evolution as discussed in Part IV to the intelligent social foraging problem (Chapter 18 serves as a concrete introduction to this rather philosophical section, since it integrates foraging concepts and evolution for *E. coli*).

Evolving Foraging Strategies: A key idea here is that via natural selection, the environment both influences an organism's physiology, and helps to define foraging success. Hence, the environment dictates what is an optimal group foraging strategy via the process of evolution. It should be the case then that optimal social vehicular foraging can only be achieved by careful consideration of vehicular "physiology" (e.g., amount of computing resources and communication capability) and the vehicle's environment.

It should be immediately clear that it is possible to simulate the effects of an evolutionary process on the design (redesign, "tweaking") of a social foraging strategy, and even the communications infrastructure that is used by the group of foragers. Clearly, foraging success would be a part of the fitness function, and neural, rule-based, planning, attentive, and learning strategies could all be encoded and evolved. Just because this is possible, does not mean that it is easy to do. Moreover, the focus in such studies should be on uncovering principles, rather than on the ad hoc construction of complex simulations that loosely emulate biological processes, but which may perhaps get lucky and provide an occasional good solution.

What principles? We must recognize that the design of social foraging strategies can be very difficult. Hence, it would be nice if we had some design principles for social foraging vehicles. Consider the following ways to uncover intuitions about design principles:

- *Designing parameters of the decision-making elements:* The complexity of the social foraging problem can make it quite difficult to pick some design parameters of the decision-making strategies, ones that may be relatively easy to pick in a single-forager problem. For instance, there should be an evolved optimal prediction horizon for planning strategies, and optimal resolution needed for the attentional and cognitive maps discussed in the last section.

- *Achieving balance between decision-making functionalities:* It is very difficult to know how much sophistication is needed for each type of decision-making mechanism (e.g., planning, learning, attention), and whether it is possible to use a complex learning strategy, but a simple planning strategy.

- *Evolving simple designs:* There should be a way to optimize the "cognitive complexity" (onboard computational resources) of the vehicle, to provide an effective yet simple engineering design. The question is whether we can use evolutionary design principles to realize the "keep it simple" principle in engineering design.

- *Studying trade-offs between computational and communication resources:* Could we study questions about whether it is possible to evolve a balance between how many communications are needed (e.g., bandwidth, communication range, level of locality) and how many learning, planning, and attentional capabilities are needed? Is it better to use lots of communications with simple decision-makers, or limited communications with

sophisticated decision-making? Clearly, some aspects of such a study may be constrained by the particular vehicular problem being studied.

- *Coevolution:* What we call the "environment" includes intelligent adversaries (predators and prey) and for some organisms, a type of "arms race" occurs where one predator evolves some capability, then another evolves a way to counteract it, and so on. This is called coevolution. Suppose that you work on a student team, where each student designs an artificial organism that is both a predator and prey for ones designed by other students. Could we evolve an "evolutionary stable strategy" (see the "For Further Study" section at the end of this part) for these organisms? Could you do this for two *groups* of vehicles that compete?

- *Darwinian design of the software:* Could an analogous approach to the one described in Section 15.7.2, be used to synthesize/tune the software for cooperative foraging strategies (i.e., where the software is constructed via implementation in a test bed and evolution)?

Evolving Vehicular Hardware: Suppose that the actual robot or vehicular *hardware* can evolve. What are the implications? Here, we simply ask the following questions:

- How do you make the hardware replicate itself with fecundity and variation, and instill inheritance into the process? How do you implement (un)natural selection via environmental influences? A central problem is one of available resources to support the fecundity and the waste that results from selection. Biological evolution is based on selection of a few of many to be the ones that reproduce; the rest are in a sense "wasted." This resource/waste problem may be a key limitation that may drive any truly evolutionary strategy to exist on the molecular scale. But, considering nanotechnology advances, molecular vehicles or robots may not be out of the question.

- Could you make this emulate evolution of biological organisms, however simple they might be?

- Would this be useful for understanding biological evolution?

- Could you argue that your hardware is alive?

- What is the engineering utility of performing hardware evolution for populations? Could it be a way to make a group of vehicles or robots more adaptable to changes in its environment?

- If you can achieve some of these objectives, then will biological evolution have spawned another type of evolution? Or is this just a natural progression that is expected from evolution that can be thought of as being subsumed in what we now think of as evolution? Evolution did spawn

many other types of optimization processes as we have discussed in this book, so why not another? It would seem that if it could, it would constitute an important event in evolutionary time.

Via combined hardware-software evolution, learning, planning, attention, rule-based, and neural systems approaches, could you implement a truly "intelligent" system?

19.7 Exercises and Design Problems

Exercise 19.1 (Properties of Static Games): This problem provides exercises to support the tutorial introduction to game theory in this chapter. You may use the code available at the Web site for the book to solve the problems.

(a) Using the examples given in the chapter, define a bimatrix game that has no Nash solution.

(b) Find a bimatrix game that has two Nash solutions.

(c) Find a game that has a Nash solution that is not stable.

(d) Find a bimatrix game where a Nash solution is the same as the minimax solution.

(e) Find a two-player game that has only one Pareto solution.

Exercise 19.2 (Static and Iterative Foraging Games): For the static foraging game studied in the chapter:

(a) Investigate the effect of changing the amount of energy it takes to travel to get a resource. (To do this, assume it is zero and show the results, then slowly increase the cost of travel to get a resource and each time, study the choices made in foraging.)

(b) Add a third resource type and develop a simulation to illustrate properties of the Nash, minimax, and Pareto solutions.

(c) Iterate the static foraging game and show how the resource profiles decrease. Study the effects of parameters of the costs on the rate of decrease (i.e., the rate that resources are eliminated from the environment).

Design Problem 19.1 (Static Foraging Games—Extensions):

(a) Extend the model developed in the chapter to a two-dimensional foraging plane. Define all details of the model.

(b) Introduce a way for each animal to find its path in the plane (e.g., via a shortest path method over the cells, with a cost from the energy to travel). Simulate to show that it finds the shortest path.

(c) Define foraging strategies, both cooperative and competitive, and evaluate their performance in simulation. Study the "iterated" case where multiple steps are taken.

Design Problem 19.2 (Evolution of Cooperation and the Iterated Prisoner's Dilemma)*: Read [35] and the chapter on evolution of strategies in [36].

(a) Produce a mathematical model of a two-person iterated prisoner's dilemma. Explain clearly what the allowable costs are for the game to represent an iterated prisoner's dilemma and why it is representative of some of the typical problems found in cooperation (and give specific examples of where iterated prisoner's dilemmas arise, not just in prisons).

(b) Using the "tit-for-tat," "tit-for-two-tats," and random strategies [35], plus two others of your choice, simulate 200 iterations of the strategies playing against each other. Compare. Discuss.

(c) Develop, using the ideas from [35, 36], an evolutionary algorithm for the strategies (i.e., one that in some way eliminates, after a certain number of iterations, strategies that are not performing well and generates extra copies of ones that perform well). Run the algorithm in a Monte Carlo simulation and explain the results. Which strategy wins? What does it mean for a strategy to be an "evolutionary stable strategy"?

(d) Explain the relevance of the theory of cooperation from [35, 36] on cooperation in multivehicle applications. Develop a specific simulation to illustrate your ideas.

Design Problem 19.3 (Evolutionary Stable Strategies)*: Read the parts of [245, 534, 213] relevant to evolutionary stable strategies and evolutionary dynamics. Write a brief introduction to evolutionary stable strategies using a matrix games approach to introduce the main ideas. Develop a simple example and simulation to illustrate the key ideas. Repeat, but for evolutionary dynamics. Explain the relevance of these ideas to engineering applications in general, and vehicular applications in particular.

Design Problem 19.4 (Challenge Problem: Foraging Games)*: Develop a dynamic foraging game, in two or more dimensions (i.e., it cannot be a foraging game on a line, like the one we studied in the chapter). Write out the full details of the mathematical model and strategies that you choose. The foraging strategy could involve the use of rules, planning, attention, or learning. Justify the choice for your strategy and quantify its computational complexity. Evaluate the performance of the strategy in simulation. Depending on your strategy (e.g., how it is parameterized), you may also want to study its evolution via a genetic algorithm.

Design Problem 19.5 (Challenge Problems: Design of Intelligent Social Foraging Strategies)⋆: The problems below are designed to integrate a number of the methods studied in this book.

(a) Develop a surrogate model method for intelligent foraging and evaluate its performance in simulation. Clearly, a key aspect of this challenge problem is how to formulate a problem that you can solve, and one that is amenable to at least simulation-based analysis of performance. You may need to consider nontraditional performance measures.

(b) Define a foraging problem (nonsocial) different from (a), and design a decision-making strategy for each forager. Include aspects of competition by having more than one noncooperative forager. Evaluate the design in simulation.

(c) Define a social vehicular foraging problem and design a decision-making strategy for each vehicle that leads to foraging success for the group. Include two opposing (adversarial) teams. Evaluate the design in simulation. Several approaches to this problem are discussed in Section 19.6.2.