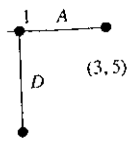

ne.

ACKWARDINDUCTION, but the equilibrium values instead of implemented through *best_util* entation of all the values that the node's children. This can simultaneously best responses. nd in the reference cited in the

tion

ade in its favor, the concept of To see why this is, consider the 5.9. (The game starts at the node ate in making decisions, at each ; the game or going "across" and g "across" also ends the game) e only SPE is for each player to r the last choice. Clearly at that wn. Since this is the case, going in the previous choice point. By e points.

except for two pesky factors. The case flies in the face of intuition ct continue to play "across" until blem is theoretical. Imagine that in the first step of the game, th you do? The SPE suggests yo ts that you would not have gon vords, you have reached a stat zero. How should you amend yo asure-zero event? It turns out th s a fundamental problem in gam here, but let us only mention th nd they depend on the probabilis edge (in particular, whether the actly how one revises one's bel uestion is intimately related to r 14.

5.2 Imperfect-information extensive-form games

Up to this point, in our discussion of extensive-form games we have allowed players to specify the action that they would take at every choice node of the game. This implies that players know the node they are in, and—recalling that in such games we equate nodes with the histories that led to them—all the prior choices, including those of other agents. For this reason we have called these *perfect-information games*.

We might not always want to make such a strong assumption about our players and our environment. In many situations we may want to model agents needing to act with partial or no knowledge of the actions taken by others, or even agents with limited memory of their own past actions. The sequencing of choices allows us to represent such ignorance to a limited degree; an "earlier" choice might be interpreted as a choice made without knowing the "later" choices. However, so far we could not represent two choices made in the same play of the game in mutual ignorance of each other.

5.2.1 *Definition*

Imperfect-information games in extensive form address this limitation. An imperfect-information game is an extensive-form game in which each player's choice nodes are partitioned into information sets; intuitively, if two choice nodes are in the same information set then the agent cannot distinguish between them.[3]

Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple* $(N, A, H, Z, \chi, \rho, \sigma, u, I)$*, where:*

- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \ldots, I_n)$*, where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is an equivalence relation on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.

Note that in order for the choice nodes to be truly indistinguishable, we require that the set of actions at each choice node in an information set be the same (otherwise, the player would be able to distinguish the nodes). Thus, if $I_{i,j} \in I_i$ is an equivalence class, we can unambiguously use the notation $\chi(I_{i,j})$ to denote the set of actions available to player i at any node in information set $I_{i,j}$.

Consider the imperfect-information extensive-form game shown in Figure 5.10. In this game, player 1 has two information sets: the set including the top choice node, and the set including the bottom choice nodes. Note that the two bottom choice nodes in the second information set have the same set of possible actions. We can regard player 1 as not knowing whether player 2 chose A or B when he makes his choice between ℓ and r.

3. From the technical point of view, imperfect-information games are obtained by overlaying a partition structure, as defined in Chapter 13 in connection with models of knowledge, over a perfect-information game.

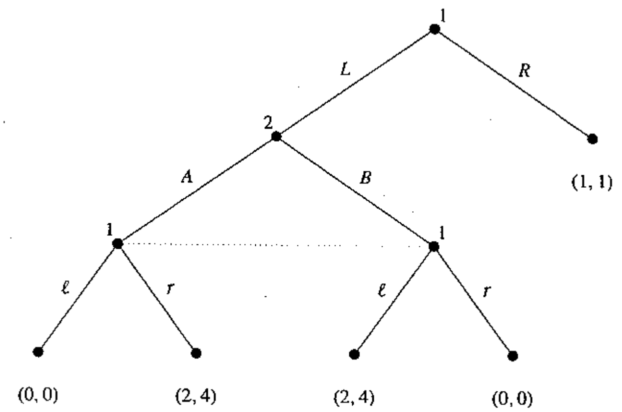

Figure 5.10 An imperfect-information game.

5.2.2 *Strategies and equilibria*

A pure strategy for an agent in an imperfect-information game selects one of the available actions in each information set of that agent.

Definition 5.2.2 (Pure strategies) *Let $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$ be an imperfect-information extensive-form game. Then the pure strategies of player i consist of the Cartesian product $\prod_{I_{i,j} \in I_i} \chi(I_{i,j})$.*

Thus perfect-information games can be thought of as a special case of imperfect-information games, in which every equivalence class of each partition is a singleton.

Consider again the Prisoner's Dilemma game, shown as a normal-form game in Figure 3.3. An equivalent imperfect-information game in extensive form is given in Figure 5.11.

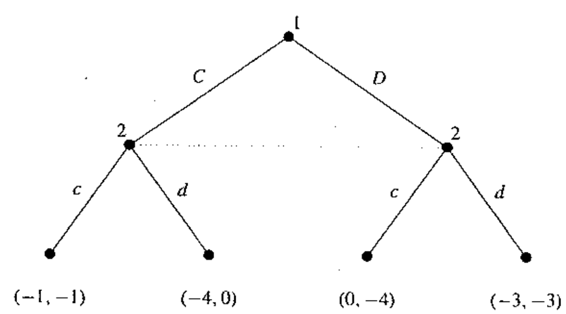

Figure 5.11 The Prisoner's Dilemma game in extensive form.

Note that we could have chosen to make player 2 choose first and player 1 choose second.

Recall that perfect-information games were not expressive enough to capture the prisoner's dilemma game and many other ones. In contrast, as is obvious from this example, any normal-form game can be trivially transformed into an equivalent imperfect-information game. However, this example is also special in

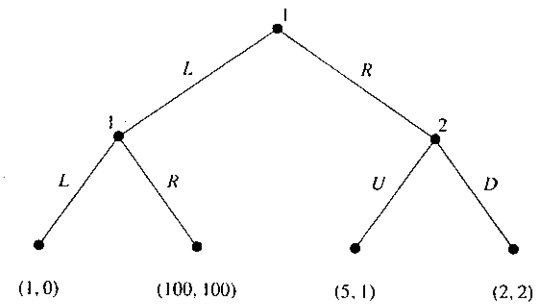

Figure 5.12 A game with imperfect recall.

that the Prisoner's Dilemma is a game with a dominant strategy solution, and thus in particular a pure-strategy Nash equilibrium. This is not true in general for imperfect-information games. To be precise about the equivalence between a normal-form game and its extensive-form image we must consider mixed strategies, and this is where we encounter a new subtlety.

As we did for perfect-information games, we can define the normal-form game corresponding to any given imperfect-information game; this normal game is again defined by enumerating the pure strategies of each agent. Now, we define the set of mixed strategies of an imperfect-information game as simply the set of mixed strategies in its image normal-form game; in the same way, we can also define the set of Nash equilibria.[4] However, we can also define the set of *behavioral strategies* in the extensive-form game. These are the strategies in which, rather than randomizing over complete pure strategies, the agent randomizes independently at each information set. And so, whereas a mixed strategy is a distribution over vectors (each vector describing a pure strategy), a behavioral strategy is a vector of distributions.

behavioral

We illustrate this distinction first in the special case of perfect-information games. For example, consider the game of Figure 5.2. A strategy for player 1 that selects A with probability .5 and G with probability .3 is a behavioral strategy. In contrast, the mixed strategy $(.6(A, G), .4(B, H))$ is not a behavioral strategy for that player, since the choices made by him at the two nodes are not independent (in fact, they are perfectly correlated).

In general, the expressive power of behavioral strategies and the expressive power of mixed strategies are noncomparable; in some games there are outcomes that are achieved via mixed strategies but not any behavioral strategies, and in some games it is the other way around.

Consider for example the game in Figure 5.12. In this game, when considering mixed strategies (but not behavioral strategies), R is a strictly dominant strategy for agent 1, D is agent 2's strict best response, and thus (R, D) is the unique Nash equilibrium. Note in particular that in a mixed strategy, agent 1 decides

4. Note that we have defined two transformations—one from any normal-form game to an imperfect-information game, and one in the other direction. However the first transformation is not one to one, and so if we transform a normal-form game to an extensive-form one and then back to normal form, we will not in general get back the same game we started out with. However, we will get a game with identical strategy spaces and equilibria.

probabilistically whether to play L or R in his information set, but once he decides he plays that pure strategy consistently. Thus the payoff of 100 is irrelevant in the context of mixed strategies. On the other hand, with behavioral strategies agent 1 gets to randomize afresh each time he finds himself in the information set. Noting that the pure strategy D is weakly dominant for agent 2 (and in fact is the unique best response to all strategies of agent 1 other than the pure strategy L), agent 1 computes the best response to D as follows. If he uses the behavioral strategy $(p, 1-p)$ (i.e., choosing L with probability p each time he finds himself in the information set), his expected payoff is

$$1 * p^2 + 100 * p(1-p) + 2 * (1-p).$$

The expression simplifies to $-99p^2 + 98p + 2$, whose maximum is obtained at $p = 98/198$. Thus $(R, D) = ((0, 1), (0, 1))$ is no longer an equilibrium in behavioral strategies, and instead we get the equilibrium $((98/198, 100/198), (0, 1))$.

There is, however, a broad class of imperfect-information games in which the expressive power of mixed and behavioral strategies coincides. This is the class of games of *perfect recall*. Intuitively speaking, in these games no player forgets any information he knew about moves made so far; in particular, he remembers precisely all his own moves. A formal definition follows.

perfect recall

Definition 5.2.3 (Perfect recall) *Player i has* perfect recall *in an imperfect-information game G if for any two nodes h, h' that are in the same information set for player i, for any path $h_0, a_0, h_1, a_1, h_2, \ldots, h_n, a_n, h$ from the root of the game to h (where the h_j are decision nodes and the a_j are actions) and for any path $h_0, a'_0, h'_1, a'_1, h'_2, \ldots, h'_m, a'_m, h'$ from the root to h' it must be the case that:*

1. *$n = m$;*
2. *for all $0 \leq j \leq n$, h_j and h'_j are in the same equivalence class for player i; and*
3. *for all $0 \leq j \leq n$, if $\rho(h_j) = i$ (i.e., h_j is a decision node of player i), then $a_j = a'_j$.*

G is a game of perfect recall if every player has perfect recall in it.

Clearly, every perfect-information game is a game of perfect recall.

Theorem 5.2.4 (Kuhn, 1953) *In a game of perfect recall, any mixed strategy of a given agent can be replaced by an equivalent behavioral strategy, and any behavioral strategy can be replaced by an equivalent mixed strategy. Here two strategies are equivalent in the sense that they induce the same probabilities on outcomes, for any fixed strategy profile (mixed or behavioral) of the remaining agents.*

As a corollary we can conclude that the set of Nash equilibria does not change if we restrict ourselves to behavioral strategies. This is true only in games of perfect recall, and thus, for example, in perfect-information games. We stress again, however, that in general imperfect-information games, mixed and behavioral strategies yield noncomparable sets of equilibria.

5.2.3 Computing eq

Because any ex form game, an is to first conve for example the because the nu of the extensiv all combinatio result when the

One way to representation. game using a c

Defining the s

The sequence extensive-forn the sequence- each of these

sequence form

Definition 5. information g a tuple $(N, \Sigma$

- N is a s
- $\Sigma = (\Sigma$
- $g = (g_1$
- $C = (C$ probabi

Now let u insight of th strategies in the game tre strategies, th each node.

sequence

Definition 5 node $h \in H$ on the path f node. The s the set of al

A seque player i w Observe tha actions that

5.2.3 *Computing equilibria: the sequence form*

Because any extensive-form game can be converted into an equivalent normal-form game, an obvious way to find an equilibrium of an extensive-form game is to first convert it into a normal-form game and then find the equilibria using, for example the Lemke–Howson algorithm. This method is inefficient, however, because the number of actions in the normal-form game is *exponential* in the size of the extensive-form game. The normal-form game is created by considering all combinations of information set actions for each player, and the payoffs that result when these strategies are employed.

One way to avoid this problem is to operate directly on the extensive-form representation. This can be done by employing behavioral strategies to express a game using a description called the sequence form.

Defining the sequence form

The sequence form is (primarily) useful for representing imperfect-information extensive-form games of perfect recall. Definition 5.2.5 describes the elements of the sequence-form representation of such games; we then go on to explain what each of these elements means.

sequence form

Definition 5.2.5 (Sequence-form representation) *Let G be an imperfect-information game of perfect recall. The* sequence-form representation *of G is a tuple (N, Σ, g, C), where:*

- *N is a set of agents;*
- *$\Sigma = (\Sigma_1, \ldots, \Sigma_n)$, where Σ_i is the set of* sequences *available to agent i;*
- *$g = (g_1, \ldots, g_n)$, where $g_i : \Sigma \mapsto \mathbb{R}$ is the payoff function for agent i; and*
- *$C = (C_1, \ldots, C_n)$, where C_i is a set of linear constraints on the realization probabilities of agent i.*

Now let us define all these terms. To begin with, what is a sequence? The key insight of the sequence form is that, while there are exponentially many pure strategies in an extensive-form game, there are only a small number of nodes in the game tree. Rather than building a player's strategy around the idea of pure strategies, the sequence form builds it around paths in the tree from the root to each node.

sequence

Definition 5.2.6 (Sequence) *A* sequence *of actions of player $i \in N$, defined by a node $h \in H \cup Z$ of the game tree, is the ordered set of player i's actions that lie on the path from the root to h. Let $\emptyset$ denote the sequence corresponding to the root node. The set of sequences of player i is denoted Σ_i, and $\Sigma = \Sigma_1 \times \cdots \times \Sigma_n$ is the set of all sequences.*

A sequence can thus be thought of as a string listing the action choices that player i would have to take in order to get from the root to a given node h. Observe that h may or may not be a leaf node; observe also that the other players' actions that form part of this path are not part of the sequence.

	$\emptyset$	A	B
$\emptyset$	0, 0	0, 0	0, 0
L	0, 0	0, 0	0, 0
R	1, 1	0, 0	0, 0
$L\ell$	0, 0	0, 0	2, 4
Lr	0, 0	2, 4	0, 0

Figure 5.13 The sequence form of the game from Figure 5.10.

	A	B
$L\ell$	0, 0	2, 4
Lr	2, 4	0, 0
$R\ell$	1, 1	1, 1
Rr	1, 1	1, 1

Figure 5.14 The induced normal form of the game from Figure 5.10.

sequence-form payoff function

Definition 5.2.7 (Payoff function) *The* payoff function $g_i : \Sigma \mapsto \mathbb{R}$ *for agent* i *is given by* $g(\sigma) = u(z)$ *if a leaf node* $z \in Z$ *would be reached when each player played his sequence* $\sigma_i \in \sigma$*, and by* $g(\sigma) = 0$ *otherwise.*

Given the set of sequences Σ and the payoff function g, we can think of the sequence form as defining a tabular representation of an imperfect-information extensive-form game, much as the induced normal form does. Consider the game given in Figure 5.10 (see p. 126). The sets of sequences for the two players are $\Sigma_1 = \{\emptyset, L, R, L\ell, Lr\}$ and $\Sigma_2 = \{\emptyset, A, B\}$. The payoff function is given in Figure 5.13. For comparison, the induced normal form of the same game is given in Figure 5.14. Written this way, the sequence form is larger than the induced normal form. However, many of the entries in the game matrix in Figure 5.13 correspond to cases where the payoff function is defined to be zero because the given pair of sequences does not correspond to a leaf node in the game tree. These entries are shaded in gray to indicate that they could not arise in play. Each payoff that *is* defined at a leaf in the game tree occurs exactly once in the sequence-form table. Thus, if g was represented using a sparse encoding, only five values would have to be stored. Compare this to the induced normal form, where all of the eight entries correspond to leaf nodes from the game tree.

We now have a set of players, a set of sequences, and a mapping from sequences to payoffs. At first glance this may look like everything we need to describe our game. However, sequences do not quite take the place of actions. In particular, a player cannot simply select a single sequence in the way that he would select a pure strategy—the other player(s) might not play in a way that would allow him to follow it to its end. Put another way, players still need to define what they would do in every information set that could be reached in the game tree.

	A	B
	0, 0	2, 4
	2, 4	0, 0
	1, 1	1, 1
	1, 1	1, 1

1 The induced normal form of the
Figure 5.10.

ction $g_i : \Sigma \mapsto \mathbb{R}$ *for agent i*
be reached when each player
rwise.

unction g, we can think of the
on of an imperfect-information
l form does. Consider the game
sequences for the two players
. The payoff function is given in
l form of the same game is given
form is larger than the induced
the game matrix in Figure 5.13
is defined to be zero because the
leaf node in the game tree. These
ould not arise in play. Each payoff
exactly once in the sequence-form
encoding, only five values would
ed normal form, where all of the
e game tree.
quences, and a mapping from se
look like everything we need to
not quite take the place of actions
single sequence in the way that
yer(s) might not play in a way that
other way, players still need to define
t that could be reached in the game

What we want is for agents to select behavioral strategies. (Since we have assumed that our game G has perfect recall, Theorem 5.2.4 tells us that any equilibrium will be expressible using behavioral strategies.) However, it turns out that it is not a good idea to work with behavioral strategies directly—if we did so, the optimization problems we develop later would be computationally harder to solve. Instead, we will develop the alternate concept of a *realization plan*, which corresponds to the probability that a given sequence would arise under a given behavioral strategy.

Consider an agent i following a behavioral strategy that assigned probability $\beta_i(h, a_i)$ to taking action a_i at a given decision node h. Then we can construct a *realization plan* that assigns probabilities to sequences in a way that recovers i's behavioral strategy β.

realization plan of β_i

realization probability

Definition 5.2.8 (Realization plan of β_i) *The* realization plan of β_i *for player $i \in N$ is a mapping $r_i : \Sigma_i \mapsto [0, 1]$ defined as $r_i(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. Each value $r_i(\sigma_i)$ is called a* realization probability.

Definition 5.2.8 is not the most useful way of defining realization probabilities. There is a second, equivalent definition with the advantage that it involves a set of linear equations, although it is a bit more complicated. This definition relies on two functions that we will make extensive use of in this section.

seq$_i(I)$: the sequence leading to I

To define the first function, we make use of our assumption that G is a game of perfect recall. This entails that, given an information set $I \in I_i$, there must be one single sequence that player i can play to reach all of his nonterminal choice nodes $h \in I$. We denote this mapping as $\text{seq}_i : I_i \mapsto \Sigma_i$, and call $\text{seq}_i(I)$ the sequence *leading to* information set I. Note that while there is only one sequence that leads to a given information set, a given sequence can lead to multiple different information sets. For example, if player 1 moves first and player 2 observes his move, then the sequence $\emptyset$ will lead to multiple information sets for player 2.

Ext$_i(\sigma_i)$: sequences extending σ_i

Ext$_i(I)$ = Ext$_i$(seq$_i(I)$)

The second function considers ways that sequences can be built from other sequences. By $\sigma_i a_i$ denote a sequence that consists of the sequence σ_i followed by the single action a_i. As long as the new sequence still belongs to Σ_i, we say that the sequence $\sigma_i a_i$ *extends* the sequence σ_i. A sequence can often be extended in multiple ways—for example, perhaps agent i could have chosen an action a_i' instead of a_i after playing sequence σ_i. We denote by $\text{Ext}_i : \Sigma_i \mapsto 2^{\Sigma_i}$ a function mapping from sequences to sets of sequences, where $\text{Ext}_i(\sigma_i)$ denotes the set of sequences that extend the sequence σ_i. We define $\text{Ext}_i(\emptyset)$ to be the set of all single-action sequences. Note that extension always refers to playing a *single* action beyond a given sequence; thus, $\sigma_i a_i a_i'$ does not belong to $\text{Ext}_i(\sigma_i)$, even if it is a valid sequence. (It *does* belong to $\text{Ext}_i(\sigma_i a_i)$.) Also note that not all sequences have extensions; one example is sequences leading to leaf nodes. In such cases $\text{Ext}_i(\sigma)$ returns the empty set. Finally, to reduce notation we introduce the shorthand $\text{Ext}_i(I) = \text{Ext}_i(\text{seq}_i(I))$: the sequences extending an information set are the sequences extending the (unique) sequence leading to that information set.

realization plan

Definition 5.2.9 (Realization plan) *A* realization plan *for player $i \in N$ is a function $r_i : \Sigma_i \mapsto [0, 1]$ satisfying the following constraints.*

$$r_i(\emptyset) = 1 \tag{5.1}$$

$$\sum_{\sigma_i' \in \mathrm{Ext}_i(I)} r_i(\sigma_i') = r_i(\mathrm{seq}_i(I)) \qquad \forall I \in I_i \tag{5.2}$$

$$r_i(\sigma_i) \geq 0 \qquad \forall \sigma_i \in \Sigma_i \tag{5.3}$$

If a player i follows a realization plan r_i, we must be able to recover a behavioral strategy β_i from it. For a decision node h for player i that is in information set $I \in I_i$, and for any sequence $(\mathrm{seq}_i(I)a_i) \in \mathrm{Ext}_i(I)$, $\beta_i(h, a_i)$ is defined as $\frac{r_i(\mathrm{seq}_i(I)a_i)}{r_i(\mathrm{seq}_i(I))}$, as long as $r_i(\mathrm{seq}_i(I)) > 0$. If $r_i(\mathrm{seq}_i(I)) = 0$ then we can assign $\beta_i(h, a_i)$ an arbitrary value from $[0, 1]$: here β_i describes the player's behavioral strategy at a node that could never be reached in play because of the player's own previous decisions, and so the value we assign to β_i is irrelevant.

Let C_i be the set of constraints (5.2) on realization plans of player i. Let $C = (C_1, \ldots, C_n)$. We have now defined all the elements[5] of a sequence-form representation $G = (N, \Sigma, g, C)$, as laid out in Definition 5.2.5.

What is the space complexity of the sequence-form representation? Unlike the normal form, the size of this representation is linear in the size of the extensive-form game. There is one sequence for each node in the game tree, plus the $\emptyset$ sequence for each player. As argued previously, the payoff function g can be represented sparsely, so that each payoff corresponding to a leaf node is stored only once, and no other payoffs are stored at all. There is one version of constraint (5.2) for each edge in the game tree. Each such constraint for player i references only $|\mathrm{Ext}_i(I)| + 1$ variables, again allowing sparse encoding.

Computing best responses in two-player games

The sequence-form representation can be leveraged to allow the computation of equilibria far more efficiently than can be done using the induced normal form. Here we will consider the case of two-player games, as it is these games for which the strongest results hold. First we consider the problem of determining player 1's best response to a fixed behavioral strategy of player 2 (represented as a realization plan). This problem can be written as the following linear program.

$$\text{maximize} \quad \sum_{\sigma_1 \in \Sigma_1} \left(\sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2) \right) r_1(\sigma_1) \tag{5.4}$$

$$\text{subject to} \quad r_1(\emptyset) = 1 \tag{5.5}$$

$$\sum_{\sigma_1' \in \mathrm{Ext}_1(I)} r_1(\sigma_1') = r_1(\mathrm{seq}_1(I)) \qquad \forall I \in I_1 \tag{5.6}$$

$$r_1(\sigma_1) \geq 0 \qquad \forall \sigma_1 \in \Sigma_1 \tag{5.7}$$

5. We do not need to explicitly store constraints (5.1) and (5.3), because they are always the same for every sequence-form representation.

n for player $i \in N$ is a func-
ints.

(5.1)

$\forall I \in I_i$ (5.2)

$\forall \sigma_i \in \Sigma_i$ (5.3)

must be able to recover a
de h for player i that is in
$(I)a_i) \in \mathrm{Ext}_i(I)$, $\beta_i(h, a_i)$ is
$r_i(\mathrm{seq}_i(I)) = 0$ then we can
ere β_i describes the player's
eached in play because of the
we assign to β_i is irrelevant.
zation plans of player i. Let
elements[5] of a sequence-form
finition 5.2.5.
orm representation? Unlike the
ar in the size of the extensive-
e in the game tree, plus the $\emptyset$
e payoff function g can be rep-
ng to a leaf node is stored only
s one version of constraint (5.2)
nstraint for player i references
se encoding.

es

ged to allow the computation of
using the induced normal form
games, as it is these games fo
ider the problem of determining
ategy of player 2 (represented a
as the following linear program

$) \, r_1(\sigma_1)$ (5.4)

(5.5)

$) \quad \forall I \in I_1$ (5.6)

$\forall \sigma_1 \in \Sigma_1$ (5.7)

(5.3), because they are always the same

This linear program is straightforward. First, observe that $g_1(\cdot)$ and $r_2(\cdot)$ are constants, while $r_1(\cdot)$ are variables. The LP states that player 1 should choose r_1 to maximize his expected utility (given in the objective function (5.4)) subject to constraints (5.5)–(5.7) which require that r_1 corresponds to a valid realization plan.

In an equilibrium, player 1 and player 2 best respond simultaneously. However, if we treated both r_1 and r_2 as variables in Equations (5.4)–(5.7) then the objective function (5.4) would no longer be linear. Happily, this problem does not arise in the dual of this linear program.[6] Denote the variables of our dual LP as v; there will be one v_I for every information set $I \in I_1$ (corresponding to constraint (5.6) from the primal) and one additional variable v_0 (corresponding to constraint (5.5)). For notational convenience, we define a "dummy" information set 0 for player 1; thus, we can consider every dual variable to correspond to an information set.

$\mathcal{I}_i(\sigma_i)$: the last information set encountered in σ_i

We now define one more function. Let $\mathcal{I}_i : \Sigma_i \mapsto I_i \cup \{0\}$ be a mapping from player i's sequences to information sets. We define $\mathcal{I}_i(\sigma_i)$ to be 0 iff $\sigma_i = \emptyset$, and to be the information set $I \in I_i$ in which the final action in σ_i was taken otherwise. Note that the information set in which each action in a sequence was taken is unambiguous because of our assumption that the game has perfect recall. Finally, we again overload notation to simplify the expressions that follow. Given a set of sequences Σ', let $\mathcal{I}_i(\Sigma')$ denote $\{\mathcal{I}_i(\sigma') | \sigma'_i \in \Sigma'_i\}$. Thus, for example, $\mathcal{I}_i(\mathrm{Ext}_i(\sigma_1))$ is the (possibly empty) set of final information sets encountered in the (possibly empty) set of extensions of σ_i.

$\mathcal{I}_i(\mathrm{Ext}_i(\sigma_1)) = \{\mathcal{I}_i(\sigma') | \sigma' \in \mathrm{Ext}_i(\sigma_1)\}$

The dual LP follows.

$$\text{minimize} \quad v_0 \tag{5.8}$$

$$\text{subject to} \quad v_{\mathcal{I}_1(\sigma_1)} - \sum_{I' \in \mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))} v_{I'} \geq \sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2) \quad \forall \sigma_1 \in \Sigma_1 \tag{5.9}$$

The variable v_0 represents player 1's expected utility under the realization plan he chooses to play, given player 2's realization plan. In the optimal solution v_0 will correspond to player 1's expected utility when he plays his best response. (This follows from LP duality—primal and dual linear programs always have the same optimal solutions.) Each other variable v_I can be understood as the portion of this expected utility that player 1 will achieve under his best-response realization plan in the subgame starting from information set I, again given player 2's realization plan r_2.

There is one version of constraint (5.9) for every sequence σ_1 of player 1. Observe that there is always exactly one positive variable on the left-hand side of the inequality, corresponding to the information set of the last action in the sequence. There can also be zero or more negative variables, each of which corresponds to a different information set in which player 1 can end up after playing the given sequence. To understand this constraint, we will consider three different cases.

First, there are zero of these negative variables when the sequence cannot be extended—that is, when player 1 never gets to move again after $\mathcal{I}_1\sigma_1$, no

6. The dual of a linear program is defined in Appendix B.

matter what player 2 does. In this case, the right-hand side of the constraint will evaluate to player 1's expected payoff from the subgame beyond σ_1, given player 2's realization probabilities r_2. (This subgame is either a terminal node or one or more decision nodes for player 2 leading ultimately to terminal nodes.) Thus, here the constraint states that the expected utility from a decision at information set $\mathcal{I}_1(\sigma_1)$ must be at least as large as the expected utility from making the decision according to σ_1. In the optimal solution this constraint will be realized as equality if σ_1 is played with positive probability; contrapositively, if the inequality is strict, σ_1 will never be played.

The second case is when the structure of the game is such that player 1 will face another decision node no matter how he plays at information set $\mathcal{I}_1(\sigma_1)$. For example, this occurs if $\sigma_1 = \emptyset$ and player 1 moves at the root node: then $\mathcal{I}_1(\mathrm{Ext}_1(\sigma_1)) = \{1\}$ (the first information set). As another example, if player 2 takes one of two moves at the root node and player 1 observes this move before choosing his own move, then for $\sigma_1 = \emptyset$ we will have $\mathcal{I}_1(\mathrm{Ext}_1(\sigma_1)) = \{1, 2\}$. Whenever player 1 is guaranteed to face another decision node, the right-hand side of constraint (5.9) will evaluate to zero because $g_1(\sigma_1, \sigma_2)$ will equal 0 for all σ_2. Thus the constraint can be interpreted as stating that player 1's expected utility at information set $\mathcal{I}_1(\sigma_1)$ must be equal to the sum of the expected utilities at the information sets $\mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))$. In the optimal solution, where v_0 is minimized, these constraints are always be realized as equality.

Finally, there is the case where there exist extensions of sequence σ_1, but where it is also possible that player 2 will play in a way that will deny player 1 another move. For example, consider the game in Figure 5.2 from earlier in the chapter. If player 1 adopts the sequence B at his first information set, then he will reach his second information set if player 2 plays F, and will reach a leaf node otherwise. In this case there will be both negative terms on the left-hand side of constraint (5.9) (one for every information set that player 1 could reach beyond sequence σ_1) *and* positive terms on the right-hand side (expressing the expected utility player 1 achieves for reaching a leaf node). Here the constraint can be interpreted as asserting that i's expected utility at $\mathcal{I}_1(\sigma_1)$ can only exceed the sum of the expected utilities of i's successor information sets by the amount of the expected payoff due to reaching leaf nodes from player 2's move(s).

Computing equilibria of two-player zero-sum games

For two-player zero-sum games the sequence form allows us to write a linear program for computing a Nash equilibrium that can be solved in time polynomial in the size of the extensive form. Note that in contrast, the methods described in Section 4.1 would require time exponential in the size of the extensive form, because they require construction of an LP with a constraint for each pure strategy of each player and a variable for each pure strategy of one of the players.

This new linear program for games in sequence form can be constructed quite directly from the dual LP given in Equations (5.8)–(5.9). Intuitively, we simply treat the terms $r_2(\cdot)$ as variables rather than constants, and add in the constraints from Definition 5.2.9 to ensure that r_2 is a valid realization plan. The program follows.

$$\text{minimize} \quad v_0 \tag{5.10}$$

$$\text{subject to} \quad v_{\mathcal{I}_1(\sigma_1)} - \sum_{I' \in \mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))} v_{I'} \geq \sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2) \qquad \forall \sigma_1 \in \Sigma_1 \tag{5.11}$$

$$r_2(\emptyset) = 1 \tag{5.12}$$

$$\sum_{\sigma_2' \in \mathrm{Ext}_2(I)} r_2(\sigma_2') = r_2(\mathrm{seq}_2(I)) \qquad \forall I \in I_2 \tag{5.13}$$

$$r_2(\sigma_2) \geq 0 \qquad \forall \sigma_2 \in \Sigma_2 \tag{5.14}$$

The fact that r_2 is now a variable means that player 2's realization plan will now be selected to minimize player 1's expected utility when player 1 best responds to it. In other words, we find a minmax strategy for player 2 against player 1, and since we have a two-player zero-sum game it is also a Nash equilibrium by Theorem 3.4.4. Observe that if we had tried this same trick with the primal LP given in Equations (5.4)–(5.7) we would have ended up with a quadratic objective function, and hence not a linear program.

Computing equilibria of two-player general-sum games

For two-player general-sum games, the problem of finding a Nash equilibrium can be formulated as a linear complementarity problem as follows.

$$r_1(\emptyset) = 1 \tag{5.15}$$

$$r_2(\emptyset) = 1 \tag{5.16}$$

$$\sum_{\sigma_1' \in \mathrm{Ext}_1(I)} r_1(\sigma_1') = r_1(\mathrm{seq}_1(I)) \qquad \forall I \in I_1 \tag{5.17}$$

$$\sum_{\sigma_2' \in \mathrm{Ext}_2(I)} r_2(\sigma_2') = r_2(\mathrm{seq}_2(I)) \qquad \forall I \in I_2 \tag{5.18}$$

$$r_1(\sigma_1) \geq 0 \qquad \forall \sigma_1 \in \Sigma_1 \tag{5.19}$$

$$r_2(\sigma_2) \geq 0 \qquad \forall \sigma_2 \in \Sigma_2 \tag{5.20}$$

$$\left(v^1_{\mathcal{I}_1(\sigma_1)} - \sum_{I' \in \mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))} v^1_{I'}\right) - \left(\sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2)\right) \geq 0 \qquad \forall \sigma_1 \in \Sigma_1 \tag{5.21}$$

$$\left(v^2_{\mathcal{I}_2(\sigma_2)} - \sum_{I' \in \mathcal{I}_2(\mathrm{Ext}_2(\sigma_2))} v^2_{I'}\right) - \left(\sum_{\sigma_1 \in \Sigma_1} g_2(\sigma_1, \sigma_2) r_1(\sigma_1)\right) \geq 0 \qquad \forall \sigma_2 \in \Sigma_2 \tag{5.22}$$

$$r_1(\sigma_1)\left[\left(v^1_{\mathcal{I}_1(\sigma_1)} - \sum_{I' \in \mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))} v^1_{I'}\right) - \left(\sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2)\right)\right] = 0 \qquad \forall \sigma_1 \in \Sigma_1 \tag{5.23}$$

$$r_2(\sigma_2)\left[\left(v^2_{\mathcal{I}_2(\sigma_2)} - \sum_{I' \in \mathcal{I}_2(\mathrm{Ext}_2(\sigma_2))} v^2_{I'}\right) - \left(\sum_{\sigma_1 \in \Sigma_1} g_2(\sigma_1, \sigma_2) r_1(\sigma_1)\right)\right] = 0 \qquad \forall \sigma_2 \in \Sigma_2 \tag{5.24}$$

Like the linear complementarity problem for two-player games in normal form given in Equations (4.14)–(4.19) on Page 91, this is a feasibility problem consisting of linear constraints and complementary slackness conditions. The linear constraints are those from the primal LP for player 1 (constraints (5.15), (5.17), and (5.19)), from the dual LP for player 1 (constraint (5.21)), and from the corresponding versions of these primal and dual programs for player 2 (constraints (5.16), (5.18), (5.20), and (5.22)). Note that we have rearranged some of these constraints by moving all terms to the left side, and have superscripted the v's with the appropriate player number.

If we stopped at constraint (5.22) we would have a linear program, but the variables v would be allowed to take arbitrarily large values. The complementary slackness conditions (constraints (5.23) and (5.24)) fix this problem at the expense of shifting us from a linear program to a linear complementarity problem. Let us examine constraint (5.23). It states that either sequence σ_1 is never played (i.e., $r_1(\sigma_1) = 0$) or that

$$v^1_{\mathcal{I}_1(\sigma_1)} - \sum_{I' \in \mathcal{I}_1(\mathrm{Ext}_1(\sigma_1))} v^1_{I'} = \sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2). \tag{5.25}$$

What does it mean for Equation (5.25) to hold? The short answer is that this equation requires a property that we previously observed of the optimal solution to the dual LP given in Equations (5.8)–(5.9): that the weak inequality in constraint (5.9) will be realized as strict equality whenever the corresponding sequence is played with positive probability. We were able to achieve this property in the dual LP by minimizing v_0; however, this does not work in the two-player general-sum case where we have both v_0^1 and v_0^2. Instead, we use the complementary slackness idea that we previously applied in the LCP for normal-form games (constraint (4.19)).

This linear complementarity program cannot be solved using the Lemke–Howson algorithm, as we were able to do with our LCP for normal-form games. However, it *can* be solved using the Lemke algorithm, a more general version of Lemke–Howson. Neither algorithm is polynomial time in the worst case. However, it is exponentially faster to run the Lemke algorithm on a game in sequence form than it is to run the Lemke–Howson algorithm on the game's induced normal form. We omit the details of how to apply the Lemke algorithm to sequence-form games, but refer the interested reader to the reference given at the end of the chapter.

5.2.4 *Sequential equilibrium*

We have already seen that the Nash equilibrium concept is too weak for perfect-information games, and how the more selective notion of subgame-perfect equilibrium can be more instructive. The question is whether this essential idea can be applied to the broader class of imperfect-information games; it turns out that it can, although the details are considerably more involved.

Recall that in a subgame-perfect equilibrium we require that the strategy of each agent be a best response in every subgame, not only (rather than L). It is immediately apparent that the definition does not apply in imperfect-information games, if for no other reason than we no longer have a well-defined notion of a subgame. What we have instead at each information set is a "subforest" or a

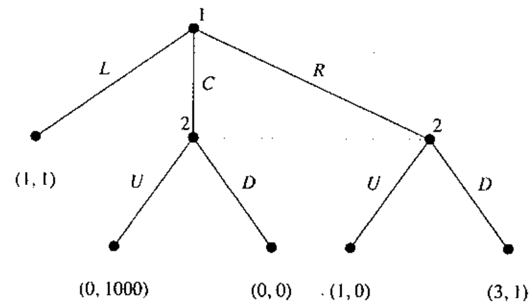

Figure 5.15 Player 2 knows where in the information set he is.

collection of subgames. We could require that each player's strategy be a best response in each subgame in each forest, but that would be both too strong a requirement and too weak. To see why it is too strong, consider the game in Figure 5.15.

The pure strategies of player 1 are $\{L, C, R\}$ and of player 2 $\{U, D\}$. Note also that the two pure Nash equilibria are (L, U) and (R, D). But should either of these be considered "subgame perfect?" On the face of it the answer is ambiguous, since in one subtree U (dramatically) dominates D and in the other D dominates U. However, consider the following argument. R dominates C for player 1, and player 2 knows this. So although player 2 does not have explicit information about which of the two nodes he is in within his information set, he can deduce that he is in the rightmost one based on player 1's incentives, and hence will go D. Furthermore player 1 knows that player 2 can deduce this, and therefore player 1 should go R. Thus, (R, D) is the only subgame-perfect equilibrium.

This example shows how a requirement that a sub strategy be a best response in all subgames is too simplistic. However, in general it is not the case that subtrees of an information set can be pruned as in the previous example so that all remaining ones agree on the best strategy for the player. In this case the naive application of the SPE intuition would rule out all strategies.

There have been several related proposals that apply the intuition underlying subgame-perfection in more sophisticated ways. One of the more influential notions has been that of *sequential equilibrium* (SE). It shares some features with the notion of trembling-hand perfection, discussed in Section 3.4.6. Note that indeed trembling-hand perfection, which was defined for normal-form games, applies here just as well; just think of the normal form induced by the extensive-form game. However, this notion makes no reference to the tree structure of the game. SE does, but at the expense of additional complexity.

sequential equilibrium

Sequential equilibrium is defined for games of perfect recall. As we have seen, in such games we can restrict our attention to behavioral strategies. Consider for the moment a fully mixed-strategy profile.[7] Such a strategy profile induces a positive probability on every node in the game tree. This means in particular

7. Again, recall that a strategy is fully mixed if, at every information set, each action is given some positive probability.

that every information set is given a positive probability. Therefore, for a given fully mixed-strategy profile, one can meaningfully speak of i's expected utility, given that he finds himself in any particular information set. (The expected utility of starting at any node is well defined, and since each node is given positive probability, one can apply Bayes' rule to aggregate the expected utilities of the different nodes in the information set.) If the fully mixed-strategy profile constitutes an equilibrium, it must be that each agent's strategy maximizes his expected utility in each of his information sets, holding the strategies of the other agents fixed.

All of the preceding discussion is for a fully mixed-strategy profile. The problem is that equilibria are rarely fully mixed, and strategy profiles that are not fully mixed do *not* induce a positive probability on every information set. The expected utility of starting in information sets whose probability is zero under the given strategy profile is simply not well defined. This is where the ingenious device of SE comes in. Given any strategy profile S (not necessarily fully mixed), imagine a probability distribution $\mu(h)$ over each information set. μ has to be *consistent* with S, in the sense that for sets whose probability is nonzero under their parents' conditional distribution S, this distribution is precisely the one defined by Bayes' rule. However, for other information sets, it can be any distribution. Intuitively, one can think of these distributions as the new beliefs of the agents, if they are surprised and find themselves in a situation they thought would not occur.[8] This means that each agent's expected utility is now well defined in any information set, including those having measure zero. For information set h belonging to agent i, with the associated probability distribution $\mu(h)$, the expected utility under strategy profile S is denoted by $u_i(S \mid h, \mu(h))$.

With this, the precise definition of SE is as follows.

Definition 5.2.10 (Sequential equilibrium) *A strategy profile S is a* sequential equilibrium *of an extensive-form game G if there exist probability distributions $\mu(h)$ for each information set h in G, such that the following two conditions hold:*

1. *$(S, \mu) = \lim_{n\to\infty}(S^n, \mu^n)$ for some sequence $(S^1, \mu^1), (S^2, \mu^2), \ldots$, where S^n is fully mixed, and μ^n is consistent with S^n (in fact, since S^n is fully mixed, μ^n is uniquely determined by S^n); and*
2. *For any information set h belonging to agent i, and any alternative strategy S_i' of i, we have that*

$$u_i(S \mid h, \mu(h)) \geq u_i((S', S_{-i}) \mid h, \mu(h)).$$

Analogous to subgame perfect equilibria in games of perfect information, sequential equilibria are guaranteed to always exist.

Theorem 5.2.11 *Every finite game of perfect recall has a sequential equilibrium.*

Finally, while sequential equilibria are defined for games of imperfect information, they are obviously also well defined for the special case of games of

8. This construction is essentially that of an LPS, discussed in Chapter 13.

perfect information. This raises the question of whether, in the context of games of perfect information, the two solution concepts coincide. The answer is that they almost do, but not quite.

Theorem 5.2.12 *Every subgame-perfect equilibrium is a sequential equilibrium, but the converse is not true in general.*[9]

5.3 History and references

As in Chapter 3, much of the material in this chapter is covered in modern game theory textbooks. Some of the historical references are as follows. The earliest game-theoretic publication is arguably that of Zermelo, who in 1913 introduced the notions of a game tree and backward induction and argued that in principle chess admits a trivial solution [Zermelo, 1913]. It was already mentioned in Chapter 3 that extensive-form games were discussed explicitly in von Neumann and Morgenstern [1944], as was backward induction. Subgame perfection was introduced by Selten [1965], who received a Nobel Prize in 1994. The material on computing all subgame-perfect equilibria is based on Littman et al. [2006]. The Centipede game was introduced by Rosenthal [1981]; many other papers discuss the rationality of backward induction in such games [Aumann, 1995, 1996; Binmore, 1996].

In 1953 Kuhn introduced extensive-form games of imperfect information, including the distinction and connection between mixed and behavioral strategies [Kuhn, 1953]. The sequence form, and its application to computing the equilibria of zero-sum games of imperfect information with perfect recall, is due to von Stengel [1996]. Many of the same ideas were developed earlier by Koller and Megiddo [1992]; see [von Stengel, 1996, pp. 242–243] for the distinctions. The use of the sequence form for computing the equilibria of general-sum two-player games of imperfect information is explained by Koller et al. [1996]. Sequential equilibria were introduced by Kreps and Wilson [1982]. Here, as in normal-form games, the full list of alternative solution concepts and connection among them is long, and the interested reader is referred to Hillas and Kohlberg [2002] and Govindan and Wilson [2005b] for a more extensive survey than is possible here.

9. For the record, the converse *is* true in so-called *generic games*, but we do not discuss those here. We do discuss genericity in *normal-form* games in Chapter 7, though there too only briefly.

6

Richer Representations: Beyond the Normal and Extensive Forms

In this chapter we will go beyond the normal and extensive forms by considering a variety of richer game representations. These further representations are important because the normal and extensive forms are not always suitable for modeling large or realistic game-theoretic settings.

First, we may be interested in games that are not finite and that therefore cannot be represented in normal or extensive form. For example, we may want to consider what happens when a simple normal-form game such as the Prisoner's Dilemma is repeated infinitely. We might want to consider a game played by an uncountably infinite set of agents. Or we may want to use an interval of the real numbers as each player's action space.[1]

Second, both of the representations we have studied so far presume that agents have perfect knowledge of everyone's payoffs. This seems like a poor model of many realistic situations, where, for example, agents might have private information that affects their own payoffs and other agents might have only probabilistic information about each others' private information. An elaboration like this can have a big impact, because one agent's actions can depend on what he knows about another agent's payoffs.

Finally, as the numbers of players and actions in a game grow—even if they remain finite—games can quickly become far too large to reason about or even to write down using the representations we have studied so far. Luckily, we are not usually interested in studying arbitrary strategic situations. The sorts of noncooperative settings that are most interesting in practice tend to involve highly structured payoffs. This can occur because of constraints imposed by the fact that the play of a game actually unfolds over time (e.g., because a large game actually corresponds to finitely repeated play of a small game). It can also occur because of the nature of the problem domain (e.g., while the world may involve many agents, the number of agents who are able to directly affect any given agent's payoff is small). If we understand the way in which agents' payoffs are structured, we can represent them much more compactly than we would be able to do using

1. We will explore the first example in detail in this chapter. A thorough treatment of infinite sets of players or action spaces is beyond the scope of this book; nevertheless, we will consider certain games with infinite sets of players in Section 6.4.4 and with infinite action spaces in Chapters 10 and 11.

the normal or extensive forms. Often, these compact representations also allow us to reason more efficiently about the games they describe (e.g., the computation of Nash equilibria can be provably faster, or pure-strategy Nash equilibria can be proved to always exist).

In this chapter we will present various different representations that address these limitations of the normal and extensive forms. In Section 6.1 we will begin by considering the special case of extensive-form games that are constructed by repeatedly playing a normal-form game and then we will extend our consideration to the case where the normal form is repeated infinitely. This will lead us to stochastic games in Section 6.2, which are like repeated games but do not require that the same normal-form game is played in each time step. In Section 6.3 we will consider structure of a different kind: instead of considering time, we will consider games involving uncertainty. Specifically, in Bayesian games agents face uncertainty—and hold private information—about the game's payoffs. Section 6.4 describes congestion games, which model situations in which agents contend for scarce resources. Finally, in Section 6.5 we will consider representations that are motivated primarily by compactness and by their usefulness for permitting efficient computation (e.g., of Nash equilibria). Such compact representations can extend any other existing representation, such as normal-form games, extensive-form games, or Bayesian games.

6.1 Repeated games

In repeated games, a given game (often thought of in normal form) is played multiple times by the same set of players. The game being repeated is called the stage game *stage game*. For example, Figure 6.1 depicts two players playing the Prisoner's Dilemma exactly twice in a row.

	C	D
C	$-1, -1$	$-4, 0$
D	$0, -4$	$-3, -3$

$\Rightarrow$

	C	D
C	$-1, -1$	$-4, 0$
D	$0, -4$	$-3, -3$

Figure 6.1 Twice-played Prisoner's Dilemma.

This representation of the repeated game, while intuitive, obscures some key factors. Do agents see what the other agents played earlier? Do they remember what they knew? And, while the utility of each stage game is specified, what is the utility of the entire repeated game?

We answer these questions in two steps. We first consider the case in which the game is repeated a finite and commonly-known number of times. Then we

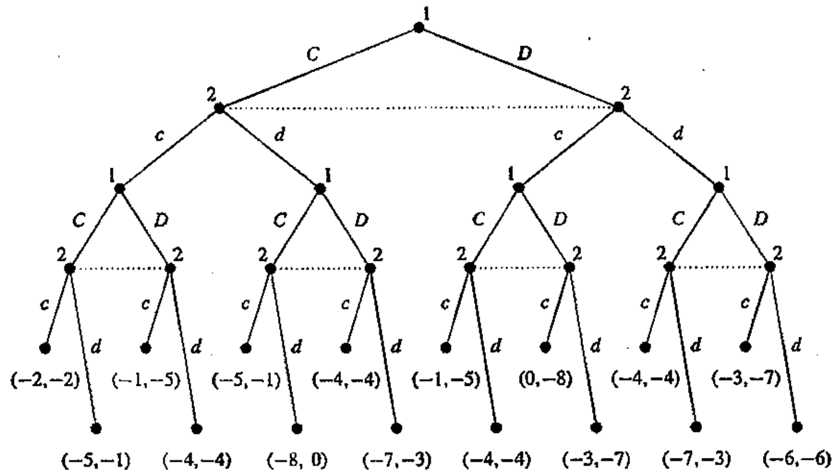

Figure 6.2 Twice-played Prisoner's Dilemma in extensive form.

consider the case in which the game is repeated infinitely often, or a finite but unknown number of times.

6.1.1 *Finitely repeated games*

One way to completely disambiguate the semantics of a finitely repeated game is to specify it as an imperfect-information game in extensive form. Figure 6.2 describes the twice-played Prisoner's Dilemma game in extensive form. Note that it captures the assumption that at each iteration the players do not know what the other player is playing, but afterward they do. Also note that the payoff function of each agent is additive; that is, it is the sum of payoffs in the two-stage games.

The extensive form also makes it clear that the strategy space of the repeated game is much richer than the strategy space in the stage game. Certainly one strategy in the repeated game is to adopt the same strategy in each stage game; clearly, this memory less strategy, called a *stationary strategy*, is a behavioral strategy in the extensive-form representation of the game. But in general, the action (or mixture of actions) played at a stage game can depend on the history of play thus far. Since this fact plays a particularly important role in infinitely repeated games, we postpone further discussion of it to the next section. Indeed, in the finite, known repetition case, we encounter again the phenomenon of backward induction, which we first encountered when we introduced subgame-perfect equilibria. Recall that in the Centipede game, discussed in Section 5.1.3, the unique SPE was to go down and terminate the game at every node. Now consider a finitely repeated Prisoner's Dilemma game. Again, it can be argued, in the last round it is a dominant strategy to defect, no matter what happened so far. This is common knowledge, and no choice of action in the preceding rounds will impact the play in the last round. Thus in the second-to-last round too it is a dominant strategy to defect. Similarly, by induction, it can be argued that the only equilibrium in this case is to always defect. However, as in the case of the

stationary strategy

Centipede game, this argument is vulnerable to both empirical and theoretical criticisms.

6.1.2 *Infinitely repeated games*

When the infinitely repeated game is transformed into extensive form, the result is an infinite tree. So the payoffs cannot be attached to any terminal nodes, nor can they be defined as the sum of the payoffs in the stage games (which in general will be infinite). There are two common ways of defining a player's payoff in an infinitely repeated game to get around this problem. The first is the average payoff of the stage game in the limit.[2]

average reward

Definition 6.1.1 (Average reward) *Given an infinite sequence of payoffs* $r_i^{(1)}$, $r_i^{(2)}, \ldots$ *for player* i, *the* average reward *of* i *is*

$$\lim_{k\to\infty} \frac{\sum_{j=1}^{k} r_i^{(j)}}{k}.$$

The *future discounted reward* to a player at a certain point of the game is the sum of his payoff in the immediate stage game, plus the sum of future rewards discounted by a constant factor. This is a recursive definition, since the future rewards again give a higher weight to early payoffs than to later ones.

future discounted reward

Definition 6.1.2 (Discounted reward) *Given an infinite sequence of payoffs* $r_i^{(1)}, r_i^{(2)}, \ldots$ *for player* i, *and a discount factor* β *with* $0 \leq \beta \leq 1$, *the* future discounted reward *of* i *is* $\sum_{j=1}^{\infty} \beta^j r_i^{(j)}$.

The discount factor can be interpreted in two ways. First, it can be taken to represent the fact that the agent cares more about his well-being in the near term than in the long term. Alternatively, it can be assumed that the agent cares about the future just as much as he cares about the present, but with some probability the game will be stopped any given round; $1 - \beta$ represents that probability. The analysis of the game is not affected by which perspective is adopted.

Now let us consider strategy spaces in an infinitely repeated game. In particular, consider the infinitely repeated Prisoner's Dilemma game. As we discussed, there are many strategies other than stationary ones. One of the most famous is *Tit-for-Tat*. TfT is the strategy in which the player starts by cooperating and thereafter chooses in round $j+1$ the action chosen by the other player in round j. Beside being both simple and easy to compute, this strategy is notoriously hard to beat; it was the winner in several repeated Prisoner's Dilemma competitions for computer programs.

Tit-for-Tat (TfT)

Since the space of strategies is so large, a natural question is whether we can characterize all the Nash equilibria of the repeated game. For example, if the discount factor is large enough, both players playing TfT is a Nash equilibrium. But there is an infinite number of others. For example, consider the *trigger strategy*. This is a draconian version of TfT; in the trigger strategy, a player starts

trigger strategy

2. The observant reader will notice a potential difficulty in this definition, since the limit may not exist. One can extend the definition to cover these cases by using the lim sup operator in Definition 6.1.1 rather than lim.

by cooperating, but if ever the other player defects then the first defects forever. Again, for sufficiently large discount factor, the trigger strategy forms a Nash equilibrium not only with itself but also with TfT.

The folk theorem—so-called because it was part of the common lore before it was formally written down—helps us understand the space of all Nash equilibria of an infinitely repeated game, by answering a related question. It does not characterize the equilibrium strategy profiles, but rather the payoffs obtained in them. Roughly speaking, it states that in an infinitely repeated game the set of average rewards attainable in equilibrium are precisely those pairs attainable under mixed strategies in a single-stage game, with the constraint on the mixed strategies that each player's payoff is at least the amount he would receive if the other players adopted minmax strategies against him.

More formally, consider any n-player game $G = (N, A, u)$ and any payoff profile $r = (r_1, r_2, \ldots, r_n)$. Let

$$v_i = \min_{s_{-i} \in S_{-i}} \max_{s_i \in S_i} u_i(s_{-i}, s_i).$$

In words, v_i is player i's minmax value: his utility when the other players play minmax strategies against him, and he plays his best response.

Before giving the theorem, we provide some more definitions.

Definition 6.1.3 (Enforceable) *A payoff profile $r = (r_1, r_2, \ldots, r_n)$ is* enforceable *if $\forall i \in N$, $r_i \geq v_i$.*

Definition 6.1.4 (Feasible) *A payoff profile $r = (r_1, r_2, \ldots, r_n)$ is* feasible *if there exist rational, nonnegative values α_a such that for all i, we can express r_i as $\sum_{a \in A} \alpha_a u_i(a)$, with $\sum_{a \in A} \alpha_a = 1$.*

In other words, a payoff profile is feasible if it is a convex, rational combination of the outcomes in G.

folk theorem

Theorem 6.1.5 (Folk Theorem) *Consider any n-player normal-form game G and any payoff profile $r = (r_1, r_2, \ldots, r_n)$.*

1. *If r is the payoff profile for any Nash equilibrium s of the infinitely repeated G with average rewards, then for each player i, r_i is enforceable.*
2. *If r is both feasible and enforceable, then r is the payoff profile for some Nash equilibrium of the infinitely repeated G with average rewards.*

This proof is both instructive and intuitive. The first part uses the definition of minmax and best response to show that an agent can never receive less than his minmax value in any equilibrium. The second part shows how to construct an equilibrium that yields each agent the average payoffs given in any feasible and enforceable payoff profile r. This equilibrium has the agents cycle in perfect lock-step through a sequence of game outcomes that achieve the desired average payoffs. If any agent deviates, the others punish him forever by playing their minmax strategies against him.

Proof. Part 1: Suppose r is not enforceable, that is, $r_i < v_i$ for some i. Then consider an alternative strategy for i: playing $BR(s_{-i}(h))$, where $s_{-i}(h)$ is the equilibrium strategy of other players given the current history h and $BR(s_{-i}(h))$ is a function that returns a best response for i to a given strategy profile s_{-i} in the (unrepeated) stage game G. By definition of a minmax strategy, player i receives a payoff of at least v_i in every stage game if he plays $BR(s_{-i}(h))$, and so i's average reward is also at least v_i. Thus, if $r_i < v_i$ then s cannot be a Nash equilibrium.

Part 2: Since r is a feasible enforceable payoff profile, we can write it as $r_i = \sum_{a\in A}(\frac{\beta_a}{\gamma})u_i(a)$, where β_a and γ are nonnegative integers. (Recall that α_a were required to be rational. So we can take γ to be their common denominator.) Since the combination was convex, we have $\gamma = \sum_{a\in A}\beta_a$.

We are going to construct a strategy profile that will cycle through all outcomes $a \in A$ of G with cycles of length γ, each cycle repeating action a exactly β_a times. Let (a^t) be such a sequence of outcomes. Let us define a strategy s_i of player i to be a trigger version of playing (a^t): if nobody deviates, then s_i plays a_i^t in period t. However, if there was a period t' in which some player $j \neq i$ deviated, then s_i will play $(p_{-j})_i$, where (p_{-j}) is a solution to the minimization problem in the definition of v_j.

First observe that if everybody plays according to s_i, then, by construction, player i receives average payoff of r_i (look at averages over periods of length γ). Second, this strategy profile is a Nash equilibrium. Suppose everybody plays according to s_i, and player j deviates at some point. Then, forever after, player j will receive his min max payoff $v_j \leq r_j$, rendering the deviation unprofitable. ∎

The reader might wonder why this proof appeals to i's minmax value rather than his maxmin value. First, notice that the trigger strategies in Part 2 of the proof use minmax strategies to punish agent i. This makes sense because even in cases where i's minmax value is strictly greater than his maxmin value,[3] i's minmax value is the smallest amount that the other agents can guarantee that i will receive. When i best responds to a minmax strategy played against him by $-i$, he receives exactly his minmax value; this is the deviation considered in Part 1.

Theorem 6.1.5 is actually an instance of a large family of folk theorems. As stated, Theorem 6.1.5 is restricted to infinitely repeated games, to average reward, to the Nash equilibrium, and to games of complete information. However, there are folk theorems that hold for other versions of each of these conditions, as well as other conditions not mentioned here. In particular, there are folk theorems for infinitely repeated games with discounted reward (for a large enough discount factor), for finitely repeated games, for subgame-perfect equilibria (i.e., where agents only administer finite punishments to deviators), and for games of incomplete information. We do not review them here, but the message of each of them

3. This can happen in games with more than two players, as discussed in Section 3.4.1.

	C	D
C	$3, 3$	$0, 4$
D	$4, 0$	$1, 1$

Figure 6.3 Prisoner's Dilemma game.

is fundamentally the same: the payoffs in the equilibria of a repeated game are essentially constrained only by enforceability and feasibility.

6.1.3 *"Bounded rationality": repeated games played by automata*

Until now we have assumed that players can engage in arbitrarily deep reasoning and mutual modeling, regardless of their complexity. In particular, consider the fact that we have tended to rely on equilibrium concepts as predictions of—or prescriptions for—behavior. Even in the relatively uncontroversial case of two-player zero-sum games, this is a questionable stance in practice; otherwise, for example, there would be no point in chess competitions. While we will continue to make this questionable assumption in much of the remainder of the book, we pause here to revisit it. We ask what happens when agents are not perfectly rational expected-utility maximizers. In particular, we ask what happens when we impose specific computational limitations on them.

Consider (yet again) an instance of the Prisoner's Dilemma, which is reproduced in Figure 6.3. In the finitely repeated version of this game, we know that each player's dominant strategy (and thus the only Nash equilibrium) is to choose the strategy D in each iteration of the game. In reality, when people actually play the game, we typically observe a significant amount of cooperation, especially in the earlier iterations of the game. While much of game theory is open to the criticism that it does not match well with human behavior, this is a particularly stark example of this divergence. What models might explain this fact?

One early proposal in the literature is based on the notion of an ϵ-equilibrium, defined in Section 3.4.7. Recall that this is a strategy profile in which no agent can gain more than ϵ by changing his strategy; a Nash equilibrium is thus the special case of a 0-equilibrium. This equilibrium concept is motivated by the idea that agents' rationality may be bounded in the sense that they are willing to settle for payoffs that are slightly below their best response payoffs. In the finitely repeated Prisoner's Dilemma game, as the number of repetitions increases, the corresponding sets of ϵ-equilibria include outcomes with longer and longer sequences of the "cooperate" strategy.

Various other models of bounded rationality exist, but we will focus on what has proved to be the richest source of results so far, namely, restricting agents' strategies to those implemented by automata of the sort investigated in computer science.

games, an alternative is to use an algorithm developed by Shapley that is related to value iteration, a commonly-used method for solving MDPs (see Appendix C).

Moving on to the average reward case, we have to impose more restrictions in order to use a linear program than we did for the discounted reward case. Specifically, for the class of two-player, general-sum, average-reward stochastic games, the single-controller assumption no longer suffices—we also need the game to be zero sum.

Even when we cannot use a linear program, irreducibility allows us to use an algorithm that is guaranteed to converge. This algorithm is a combination of policy iteration (another method used for solving MDPs) and successive approximation.

6.3 Bayesian games

All of the game forms discussed so far assumed that all players know what game is being played. Specifically, the number of players, the actions available to each player, and the payoff associated with each action vector have all been assumed to be common knowledge among the players. Note that this is true even of imperfect-information games; the actual moves of agents are not common knowledge, but the game itself is. In contrast, *Bayesian games*, or games of incomplete information, allow us to represent players' uncertainties about the very game being played.[4] This uncertainty is represented as a probability distribution over a set of possible games. We make two assumptions.

Bayesian game

1. All possible games have the same number of agents and the same strategy space for each agent; they differ only in their payoffs.
2. The beliefs of the different agents are posteriors, obtained by conditioning a common prior on individual private signals.

The second assumption is substantive, and we return to it shortly. The first is not particularly restrictive, although at first it might seem to be. One can imagine many other potential types of uncertainty that players might have about the game—how many players are involved, what actions are available to each player, and perhaps other aspects of the situation. It might seem that we have severely limited the discussion by ruling these out. However, it turns out that these other types of uncertainty can be reduced to uncertainty only about payoffs via problem reformulation.

For example, imagine that we want to model a situation in which one player is uncertain about the number of actions available to the other players. We can reduce this uncertainty to uncertainty about payoffs by padding the game with irrelevant actions. For example, consider the following two-player game, in which the row player does not know whether his opponent has only the two strategies L and R or also the third one C:

4. It is easy to confuse the term "incomplete information" with "imperfect information"; don't...

	L	R
U	1, 1	1, 3
D	0, 5	1, 13

	L	C	R
U	1, 1	0, 2	1, 3
D	0, 5	2, 8	1, 13

Now consider replacing the leftmost, smaller game by a padded version, in which we add a new C column.

	L	C	R
U	1, 1	0, −100	1, 3
D	0, 5	2, −100	1, 13

Clearly the newly added column is dominated by the others and will not participate in any Nash equilibrium (or any other reasonable solution concept). Indeed, there is an isomorphism between Nash equilibria of the original game and the padded one. Thus the uncertainty about the strategy space can be reduced to uncertainty about payoffs.

Using similar tactics, it can be shown that it is also possible to reduce uncertainty about other aspects of the game to uncertainty about payoffs only. This is not a mathematical claim, since we have given no mathematical characterization of all the possible forms of uncertainty, but it is the case that such reductions have been shown for all the common forms of uncertainty.

common-prior assumption

The second assumption about Bayesian games is the *common-prior assumption*, addressed in more detail in our discussion of multiagent probabilities and KP-structures in Chapter 13. As discussed there, a Bayesian game thus defines not only the uncertainties of agents about the game being played, but also their beliefs about the beliefs of other agents about the game being played, and indeed an entire infinite hierarchy of nested beliefs (the so-called epistemic type space). As also discussed in Chapter 13, the common-prior assumption is a substantive assumption that limits the scope of applicability. We nonetheless make this assumption since it allows us to formulate the main ideas in Bayesian games, and without the assumption the subject matter becomes much more involved than is appropriate for this text. Indeed, most (but not all) work in game theory makes this assumption.

6.3.1 *Definition*

There are several ways of presenting Bayesian games; we will offer three different definitions. All three are equivalent, modulo some subtleties that lie outside the

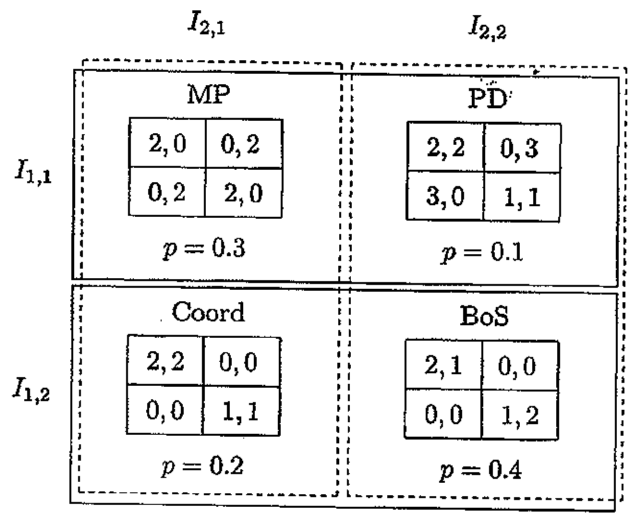

Figure 6.7 A Bayesian game.

scope of this book. We include all three since each formulation is useful in different settings and offers different intuition about the underlying structure of this family of games.

Information sets

First, we present a definition that is based on information sets. Under this definition, a Bayesian game consists of a set of games that differ only in their payoffs, a common prior defined over them, and a partition structure over the games for each agent.[5]

Bayesian game

Definition 6.3.1 (Bayesian game: information sets) *A* Bayesian game *is a tuple* (N, G, P, I) *where:*

- *N is a set of agents;*
- *G is a set of games with N agents each such that if $g, g' \in G$ then for each agent $i \in N$ the strategy space in g is identical to the strategy space in g';*
- *$P \in \Pi(G)$ is a common prior over games, where $\Pi(G)$ is the set of all probability distributions over G; and*
- *$I = (I_1, ..., I_N)$ is a tuple of partitions of G, one for each agent.*

Figure 6.7 gives an example of a Bayesian game. It consists of four 2×2 games (Matching Pennies, Prisoner's Dilemma, Coordination and Battle of the Sexes), and each agent's partition consists of two equivalence classes.

Extensive form with chance moves

A second way of capturing the common prior is to hypothesize a special agent called Nature who makes probabilistic choices. While we could have Nature's

5. This combination of a common prior and a set of partitions over states of the world turns out to correspond to a KP-structure, defined in Chapter 13.

choice be interspersed arbitrarily with the agents' moves, without loss of generality we assume that Nature makes all its choices at the outset. Nature does not have a utility function (or, alternatively, can be viewed as having a constant one), and has the unique strategy of randomizing in a commonly known way. The agents receive individual signals about Nature's choice, and these are captured by their information sets in a standard way. The agents have no additional information; in particular, the information sets capture the fact that agents make their choices without knowing the choices of others. Thus, we have reduced games of incomplete information to games of imperfect information, albeit ones with chance moves. These chance moves of Nature require minor adjustments of existing definitions, replacing payoffs by their expectations given Nature's moves.[6]

For example, the Bayesian game of Figure 6.7 can be represented in extensive form as depicted in Figure 6.8.

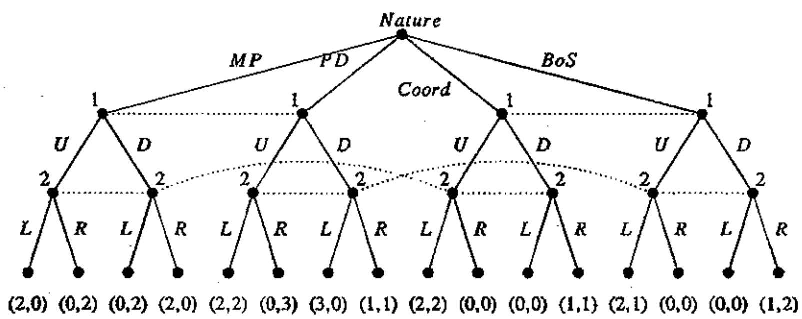

Figure 6.8 The Bayesian game from Figure 6.7 in extensive form.

Although this second definition of Bayesian games can be initially more intuitive than our first definition, it can also be more cumbersome to work with. This is because we use an extensive-form representation in a setting where players are unable to observe each others' moves. (Indeed, for the same reason we do not routinely use extensive-form games of imperfect information to model simultaneous interactions such as the Prisoner's Dilemma, though we could do so if we wished.) For this reason, we will not make further use of this definition. We close by noting one advantage that it does have, however: it extends very naturally to Bayesian games in which players move sequentially and do (at least sometimes) learn about previous players' moves.

Epistemic types

Recall that a game may be defined by a set of players, actions, and utility functions. In our first definition agents are uncertain about which game they are playing; however, each possible game has the same sets of actions and players, and so agents are really only uncertain about the game's utility function.

6. Note that the special structure of this extensive-form game means that we do not have to agonize over the refinements of Nash equilibrium; since agents have no information about prior choices made other than by Nature, all Nash equilibria are also sequential equilibria.

a_1	a_2	θ_1	θ_2	u_1	u_2
U	L	$\theta_{1,1}$	$\theta_{2,1}$	2	0
U	L	$\theta_{1,1}$	$\theta_{2,2}$	2	2
U	L	$\theta_{1,2}$	$\theta_{2,1}$	2	2
U	L	$\theta_{1,2}$	$\theta_{2,2}$	2	1
U	R	$\theta_{1,1}$	$\theta_{2,1}$	0	2
U	R	$\theta_{1,1}$	$\theta_{2,2}$	0	3
U	R	$\theta_{1,2}$	$\theta_{2,1}$	0	0
U	R	$\theta_{1,2}$	$\theta_{2,2}$	0	0

a_1	a_2	θ_1	θ_2	u_1	u_2
D	L	$\theta_{1,1}$	$\theta_{2,1}$	0	2
D	L	$\theta_{1,1}$	$\theta_{2,2}$	3	0
D	L	$\theta_{1,2}$	$\theta_{2,1}$	0	0
D	L	$\theta_{1,2}$	$\theta_{2,2}$	0	0
D	R	$\theta_{1,1}$	$\theta_{2,1}$	2	0
D	R	$\theta_{1,1}$	$\theta_{2,2}$	1	1
D	R	$\theta_{1,2}$	$\theta_{2,1}$	1	1
D	R	$\theta_{1,2}$	$\theta_{2,2}$	1	2

Figure 6.9 Utility functions u_1 and u_2 for the Bayesian game from Figure 6.7.

epistemic type

Our third definition uses the notion of an *epistemic type*, or simply a *type* as a way of defining uncertainty directly over a game's utility function.

Bayesian game

Definition 6.3.2 (Bayesian game: types) *A* Bayesian game *is a tuple* (N, A, Θ, p, u) *where:*

- *N is a set of agents;*
- *$A = A_1 \times \cdots \times A_n$, where A_i is the set of actions available to player i;*
- *$\Theta = \Theta_1 \times \ldots \times \Theta_n$, where Θ_i is the type space of player i;*
- *$p : \Theta \mapsto [0, 1]$ is a common prior over types; and*
- *$u = (u_1, \ldots, u_n)$, where $u_i : A \times \Theta \mapsto \mathbb{R}$ is the utility function for player i.*

The assumption is that all of the above is common knowledge among the players, and that each agent knows his own type. This definition can seem mysterious, because the notion of type can be rather opaque. In general, the type of agent encapsulates all the information possessed by the agent that is not common knowledge. This is often quite simple (e.g., the agent's knowledge of his private payoff function), but can also include his beliefs about other agents' payoffs, about their beliefs about his own payoff, and any other higher-order beliefs.

We can get further insight into the notion of a type by relating it to the formulation at the beginning of this section. Consider again the Bayesian game in Figure 6.7. For each of the agents we have two types, corresponding to his two information sets. Denote player 1's actions as U and D, player 2's actions as L and R. Call the types of the first agent $\theta_{1,1}$ and $\theta_{1,2}$, and those of the second agent $\theta_{2,1}$ and $\theta_{2,2}$. The joint distribution on these types is as follows: $p(\theta_{1,1}, \theta_{2,1}) = .3$, $p(\theta_{1,1}, \theta_{2,2}) = .1$, $p(\theta_{1,2}, \theta_{2,1}) = .2$, $p(\theta_{1,2}, \theta_{2,2}) = .4$. The conditional probabilities for the first player are $p(\theta_{2,1} \mid \theta_{1,1}) = 3/4$, $p(\theta_{2,2} \mid \theta_{1,1}) = 1/4$, $p(\theta_{2,1} \mid \theta_{1,2}) = 1/3$, and $p(\theta_{2,2} \mid \theta_{1,2}) = 2/3$. Both players' utility functions are given in Figure 6.9.

6.3.2 *Strategies and equilibria*

Now that we have defined Bayesian games, we must explain how to reason about them. We will do this using the epistemic type definition, because that is the definition most commonly used in mechanism design (discussed in Chapter 10), one of the main applications of Bayesian games. All of the concepts defined below can also be expressed in terms of the first two Bayesian game definitions as well.

The first task is to define an agent's strategy space in a Bayesian game. Recall that in an imperfect-information extensive-form game a pure strategy is a mapping from information sets to actions. The definition is similar in Bayesian games: a pure strategy $\alpha_i : \Theta_i \mapsto A_i$ is a mapping from every type agent i could have to the action he would play if he had that type. We can then define mixed strategies in the natural way as probability distributions over pure strategies. As before, we denote a mixed strategy for i as $s_i \in S_i$, where S_i is the set of all i's mixed strategies. Furthermore, we use the notation $s_j(a_j|\theta_j)$ to denote the probability under mixed strategy s_j that agent j plays action a_j, given that j's type is θ_j.

Next, since we have defined an environment with multiple sources of uncertainty, we will pause to reconsider the definition of an agent's expected utility. In a Bayesian game setting, there are three meaningful notions of expected utility: *ex post*, *ex interim* and *ex ante*. The first is computed based on all agents' actual types, the second considers the setting in which an agent knows his own type but not the types of the other agents, and in the third case the agent does not know anybody's type.

ex post expected utility

Definition 6.3.3 (*Ex post* expected utility) *Agent i's* ex post *expected utility in a Bayesian game (N, A, Θ, p, u), where the agents' strategies are given by s and the agent' types are given by θ, is defined as*

$$EU_i(s, \theta) = \sum_{a \in A} \left(\prod_{j \in N} s_j(a_j|\theta_j) \right) u_i(a, \theta). \tag{6.1}$$

In this case, the only uncertainty concerns the other agents' mixed strategies, since agent i's *ex post* expected utility is computed based on the other agents' actual types. Of course, in a Bayesian game no agent *will* know the others' types; while that does not prevent us from offering the definition given, it might make the reader question its usefulness. We will see that this notion of expected utility is useful both for defining the other two and also for defining a specialized equilibrium concept.

Definition 6.3.4 (*Ex interim* expected utility) *Agent i's* ex interim *expected utility in a Bayesian game (N, A, Θ, p, u), where i's type is θ_i and where the agents' strategies are given by the mixed-strategy profile s, is defined as*

$$EU_i(s, \theta_i) = \sum_{\theta_{-i} \in \Theta_{-i}} p(\theta_{-i}|\theta_i) \sum_{a \in A} \left(\prod_{j \in N} s_j(a_j|\theta_j) \right) u_i(a, \theta_{-i}, \theta_i), \tag{6.2}$$

or equivalently as

$$EU_i(s, \theta_i) = \sum_{\theta_{-i} \in \Theta_{-i}} p(\theta_{-i}|\theta_i) EU_i(s, (\theta_i, \theta_{-i})). \tag{6.3}$$

Thus, i must consider every assignment of types to the other agents θ_{-i} and every pure action profile a in order to evaluate his utility function $u_i(a, \theta_i, \theta_{-i})$. He must weight this utility value by two amounts: the probability that the other players' types would be θ_{-i} given that his own type is θ_i, and the probability that the pure action profile a would be realized given all players' mixed strategies and types. (Observe that agents' types may be correlated.) Because uncertainty over mixed strategies was already handled in the *ex post* case, we can also write *ex interim* expected utility as a weighted sum of $EU_i(s, \theta)$ terms.

Finally, there is the *ex ante* case, where we compute i's expected utility under the joint mixed strategy s without observing any agents' types.

ex ante expected utility

Definition 6.3.5 (*Ex ante* expected utility) *Agent i's* ex ante *expected utility in a Bayesian game (N, A, Θ, p, u), where the agents' strategies are given by the mixed-strategy profile s, is defined as*

$$EU_i(s) = \sum_{\theta \in \Theta} p(\theta) \sum_{a \in A} \left(\prod_{j \in N} s_j(a_j|\theta_j) \right) u_i(a, \theta), \tag{6.4}$$

or equivalently as

$$EU_i(s) = \sum_{\theta \in \Theta} p(\theta) EU_i(s, \theta), \tag{6.5}$$

or again equivalently as

$$EU_i(s) = \sum_{\theta_i \in \Theta_i} p(\theta_i) EU_i(s, \theta_i). \tag{6.6}$$

Next, we define best response.

best response in a Bayesian game

Definition 6.3.6 (Best response in a Bayesian game) *The set of agent i's* best responses *to mixed-strategy profile s_{-i} are given by*

$$BR_i(s_{-i}) = \arg\max_{s_i' \in S_i} EU_i(s_i', s_{-i}). \tag{6.7}$$

Note that BR_i is a set because there may be many strategies for i that yield the same expected utility. It may seem odd that BR is calculated based on i's *ex ante* expected utility. However, write $EU_i(s)$ as $\sum_{\theta_i \in \Theta_i} p(\theta_i) EU_i(s, \theta_i)$ and observe that $EU_i(s_i', s_{-i}, \theta_i)$ does not depend on strategies that i would play if his type were not θ_i. Thus, we are in fact performing independent maximization of i's *ex interim* expected utilities conditioned on each type that he could have. Intuitively speaking, if a certain action is best after the signal is received, it is also the best conditional plan devised ahead of time for what to do should that signal be received.

We are now able to define the Bayes–Nash equilibrium.

Bayes–Nash equilibrium

Definition 6.3.7 (Bayes–Nash equilibrium) *A* Bayes–Nash equilibrium *is a mixed-strategy profile s that satisfies $\forall i \;\; s_i \in BR_i(s_{-i})$.*

This is exactly the definition we gave for the Nash equilibrium in Definition 3.3.4: each agent plays a best response to the strategies of the other players. The difference from Nash equilibrium, of course, is that the definition of Bayes–Nash equilibrium is built on top of the Bayesian game definitions of best response and expected utility. Observe that we would not be able to define equilibrium in this way if an agent's strategies were not defined for every possible type. In order for a given agent i to play a best response to the other agents $-i$, i must know what strategy each agent would play for each of his possible types. Without this information, it would be impossible to evaluate the term $EU_i(s_i', s_{-i})$ in Equation (6.7).

6.3.3 *Computing equilibria*

Despite its similarity to the Nash equilibrium, the Bayes–Nash equilibrium may seem more conceptually complicated. However, as we did with extensive-form games, we can construct a normal-form representation that corresponds to a given Bayesian game.

As with games in extensive form, the induced normal form for Bayesian games has an action for every pure strategy. That is, the actions for an agent i are the distinct mappings from Θ_i to A_i. Each agent i's payoff given a pure-strategy profile s is his *ex ante* expected utility under s. Then, as it turns out, the Bayes–Nash equilibria of a Bayesian game are precisely the Nash equilibria of its induced normal form. This fact allows us to note that Nash's theorem applies directly to Bayesian games, and hence that Bayes–Nash equilibria always exist.

An example will help. Consider the Bayesian game from Figure 6.9. Note that in this game each agent has four possible pure strategies (two types and two actions). Then player 1's four strategies in the Bayesian game can be labeled UU, UD, DU, and DD: UU means that 1 chooses U regardless of his type, UD that he chooses U when he has type $\theta_{1,1}$ and D when he has type $\theta_{1,2}$, and so forth. Similarly, we can denote the strategies of player 2 in the Bayesian game by RR, RL, LR, and LL.

We now define a 4×4 normal-form game in which these are the four strategies of the two agents, and the payoffs are the expected payoffs in the individual games, given the agents' common prior beliefs. For example, player 2's *ex ante* expected utility under the strategy profile (UU, LL) is calculated as follows:

$$
\begin{aligned}
&u_2(UU, LL)\\
&\quad = \sum_{\theta \in \Theta} p(\theta) u_2(U, L, \theta)\\
&\quad = p(\theta_{1,1}, \theta_{2,1}) u_2(U, L, \theta_{1,1}, \theta_{2,1}) + p(\theta_{1,1}, \theta_{2,2}) u_2(U, L, \theta_{1,1}, \theta_{2,2})\\
&\qquad + p(\theta_{1,2}, \theta_{2,1}) u_2(U, L, \theta_{1,2}, \theta_{2,1}) + p(\theta_{1,2}, \theta_{2,2}) u_2(U, L, \theta_{1,2}, \theta_{2,2})\\
&\quad = 0.3(0) + 0.1(2) + 0.2(2) + 0.4(1) = 1.
\end{aligned}
$$

Continuing in this manner, the complete payoff matrix can be constructed as shown in Figure 6.10.

	LL	LR	RL	RR
UU	2, 1	1, 0.7	1, 1.2	0, 0.9
UD	0.8, 0.2	1, 1.1	0.4, 1	0.6, 1.9
DU	1.5, 1.4	0.5, 1.1	1.7, 0.4	0.7, 0.1
DD	0.3, 0.6	0.5, 1.5	1.1, 0.2	1.3, 1.1

Figure 6.10 Induced normal form of the game from Figure 6.9.

Now the game may be analyzed straightforwardly. For example, we can determine that player 1's best response to RL is DU.

Given a particular signal, the agent can compute the posterior probabilities and recompute the expected utility of any given strategy vector. Thus in the previous example once the row agent gets the signal $\theta_{1,1}$ he can update the expected payoffs and compute the new game shown in Figure 6.11.

	LL	LR	RL	RR
UU	2, 0.5	1.5, 0.75	0.5, 2	0, 2.25
UD	2, 0.5	1.5, 0.75	0.5, 2	0, 2.25
DU	0.75, 1.5	0.25, 1.75	2.25, 0	1.75, 0.25
DD	0.75, 1.5	0.25, 1.75	2.25, 0	1.75, 0.25

Figure 6.11 *Ex interim* induced normal-form game, where player 1 observes type $\theta_{1,1}$.

Note that for the row player, DU is still a best response to RL; what has changed is how much better it is compared to the other three strategies. In particular, the row player's payoffs are now independent of his choice of which action to take upon observing type $\theta_{1,2}$; in effect, conditional on observing type $\theta_{1,1}$ the player needs only to select a single action U or D. (Thus, we could have written the *ex interim* induced normal form in Figure 6.11 as a table with four columns but only two rows.)

Although we can use this matrix to find best responses for player 1, it turns out to be meaningless to analyze the Nash equilibria in this payoff matrix. This is because these expected payoffs are not common knowledge; if the column player were to condition on his signal, he would arrive at a different set of numbers (though, again, for him best responses would be preserved). Ironically, it is only in the induced normal form, in which the payoffs do not correspond to any *ex interim* assessment of any agent, that the Nash equilibria are meaningful.

Other computational techniques exist for Bayesian games that also have temporal structure—that is, for Bayesian games written using the "extensive form with chance moves" formulation, for which the game tree is smaller than its induced normal form. First, there is an algorithm for Bayesian games of perfect information that generalizes backward induction (defined in Section 5.1.4), is called *expectimax*. Intuitively, this algorithm is very much like the standard backward induction algorithm given in Figure 5.6. Like that algorithm, expectimax recursively explores a game tree, labeling each non-leaf node h with a payoff vector by examining the labels of each of h's child nodes—the actual payoffs when these child nodes are leaf nodes—and keeping the payoff vector in which the agent who moves at h achieves maximal utility. The new wrinkle is that chance nodes must also receive labels. Expectimax labels a chance node h with a weighted sum of the labels of its child nodes, where the weights are the probabilities that each child node will be selected. The same idea of labeling chance nodes with the expected value of the next node's label can also be applied to extend the minimax algorithm (from which expectimax gets its name) and alpha-beta pruning (see Figure 5.7) in order to solve zero-sum games. This is a popular algorithmic framework for building computer players for perfect-information games of chance such as Backgammon.

expectimax algorithm

There are also efficient computational techniques for computing sample equilibria of imperfect-information extensive-form games with chance nodes. In particular, all the computational results for computing with the sequence form that we discussed in Section 5.2.3 still hold when chance nodes are added. Intuitively, the only change we need to make is to replace our definition of the payoff function (Definition 5.2.7) with an expected payoff that supplies the expected value, ranging over Nature's possible actions, of the payoff the agent would achieve by following a given sequence. This means that we can sometimes achieve a substantial computational savings by working with the extensive-form representation of a Bayesian game, rather than considering the game's induced normal form.

6.3.4 Ex post *equilibrium*

Finally, working with *ex post* utilities allows us to define an equilibrium concept that is stronger than the Bayes–Nash equilibrium.

ex post equilibrium

Definition 6.3.8 (*Ex post* equilibrium) *An* ex post *equilibrium is a mixed-strategy profile s that satisfies $\forall\theta$, $\forall i$, $s_i \in \arg\max_{s_i' \in S_i} EU_i(s_i', s_{-i}, \theta)$.*

Observe that this definition does not presume that each agent actually *does* know the others' types; instead, it says that no agent would ever want to deviate from his mixed strategy *even if* he knew the complete type vector θ. This form of equilibrium is appealing because it is unaffected by perturbations in the type distribution $p(\theta)$. Said another way, an *ex post* equilibrium does not ever require any agent to believe that the others have accurate beliefs about his own type distribution. (Note that a standard Bayes–Nash equilibrium *can* imply this requirement.) The *ex post* equilibrium is thus similar in flavor to equilibria in dominant strategies, which do not require agents to believe that other agents act rationally.

Indeed, many dominant strategy equilibria are also *ex post* equilibria, making it easy to believe that this relationship always holds. In fact, it does not, as the following example shows. Consider a two-player Bayesian game where each agent has two actions and two corresponding types ($\forall_{i \in N}, A_i = \Theta_i = \{H, L\}$) distributed uniformly ($\forall_{i \in N}, P(\theta_i = H) = 0.5$), and with the same utility function for each agent i:

$$u_i(a, \theta) = \begin{cases} 10 & a_i = \theta_{-i} = \theta_i; \\ 2 & a_i = \theta_{-i} \neq \theta_i; \\ 0 & \text{otherwise.} \end{cases}$$

In this game, each agent has a dominant strategy of choosing the action that corresponds to his type, $a_i = \theta_i$. An equilibrium in these dominant strategies is not *ex post* because if either agent knew the other's type, he would prefer to deviate to playing the strategy that corresponds to the other agent's type, $a_i = \theta_{-i}$.

Unfortunately, another sense in which *ex post* equilibria are in fact similar to equilibria in dominant strategies is that neither kind of equilibrium is guaranteed to exist.

Finally, we note that the term "*ex post* equilibrium" has been used in several different ways in the literature. One alternate usage requires that each agent's strategy constitute a best response not only to every possible *type* of the others, but also to every *pure strategy profile* that can be realized given the others' mixed strategies. (Indeed, this solution concept has also been applied in settings where there is no uncertainty about agents' types.) A third usage even more stringently requires that no agent ever play a mixed strategy. Both of these definitions can be useful, e.g., in the context of mechanism design (see Chapter 10). However, the advantage of Definition 6.3.8 is that of the three, it describes the most general prior-free equilibrium concept for Bayesian games.

6.4 Congestion games

Congestion games are a restricted class of games that are useful for modeling some important real-world settings and that also have attractive theoretical properties. Intuitively, they simplify the representation of a game by imposing constraints on the effects that a single agent's action can have on other agents' utilities.

6.4.1 *Definition*

Intuitively, in a congestion game each player chooses some subset from a set of resources, and the cost of each resource depends on the number of other agents who select it. Formally, a congestion game is single-shot n-player game, defined as follows.

congestion game

Definition 6.4.1 (Congestion game) *A* congestion game *is a tuple* (N, R, A, c), *where*

- N *is a set of* n *agents;*
- R *is a set of* r resources;