Algorithmic Game Theory

Edited by
Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay Vazirani

Contents

3 Equilibrium Computation for Two-Player Games ***B. von Stengel*** *page* 4
3.1 Introduction 4
3.2 Bimatrix games and the best response condition 5
3.3 Equilibria via labeled polytopes 9
3.4 The Lemke–Howson algorithm 13
3.5 Integer pivoting 16
3.6 Degenerate games 19
3.7 Extensive games and their strategic form 21
3.8 Subgame perfect equilibria 23
3.9 Reduced strategic form 24
3.10 The sequence form 25
3.11 Computing equilibria with the sequence form 28
3.12 Further reading 31
3.13 Discussion and open problems 31

3

Equilibrium Computation for Two-Player Games in Strategic and Extensive Form

Bernhard von Stengel

Abstract

We explain algorithms for computing Nash equilibria of two-player games given in strategic form or extensive form. The strategic form is a table that lists the players' strategies and resulting payoffs. The "best response" condition states that in equilibrium, all pure strategies in the support of a mixed strategy must get maximal, and hence equal, payoff. The resulting equations and inequalities define polytopes, whose "completely labeled" vertex pairs are the Nash equilibria of the game. The Lemke–Howson algorithm follows a path of edges of the polytope pair that leads to one equilibrium. Extensive games are game trees, with information sets that model imperfect information of the players. Strategies in an extensive game are combinations of moves, so the strategic form has exponential size. In contrast, the linear-sized sequence form of the extensive game describes sequences of moves and how to randomize between them.

3.1 Introduction

A basic model in noncooperative game theory is the *strategic form* that defines a game by a set of strategies for each player and a payoff to each player for any strategy profile (which is a combination of strategies, one for each player). The central solution concept for such games is the *Nash equilibrium*, a strategy profile where each strategy is a *best response* to the fixed strategies of the other players. In general, equilibria exist only in *mixed* (randomized) strategies, with probabilities that fulfill certain equations and inequalities. Solving these constraints is an algorithmic problem. Its computational complexity is discussed in Chapter 2.

In this chapter, we describe methods for finding equilibria in sufficient detail to show how they could be implemented. We restrict ourselves to

games with *two players*. These can be studied using *polyhedra*, because a player's expected payoffs are linear in the mixed strategy probabilities of the other player. Nash equilibria of games with more than two players involve expected payoffs that are products of the other players' probabilities. The resulting polynomial equations and inequalities require different approaches.

For games in strategic form, we give the basic "best response condition" (Prop. 3.1, see Section 3.2), explain the use of polyhedra (Section 3.3), and describe the Lemke–Howson algorithm that finds one Nash equilibrium (Section 3.4). An implementation without numerical errors uses integer pivoting (Section 3.5). "Generic" games (that is, "almost all" games with real payoffs) are nondegenerate (see Def. 3.2); degenerate games are considered in Section 3.5.

An *extensive* game (defined in Section 3.7) is a fundamental model of dynamic interactions. A game tree models in detail the moves available to the players and their information over time. The nodes of the tree represent game states. An *information set* is a set of states in which a player has the same moves, and does not know which state he is in. A player's strategy in an extensive game specifies a move for each information set, so a player may have exponentially many strategies. This complexity can be reduced: *Subgames* (see Section 3.8) are subtrees so that all players know they are in the subgame. Finding equilibria inductively for subgames leads to *subgame perfect* equilibria, but this reduces the complexity only if players are sufficiently often (e.g., always) informed about the game state. The *reduced* strategic form applies to general games (see Section 3.9), but may still be exponential. A player has *perfect recall* if his information sets reflect that he remembers his earlier moves. Players can then randomize locally with *behavior strategies*. This classic theorem (Corollary 3.12) is turned into an algorithm with the *sequence form* (Sections 3.10 and 3.11) which is a strategic description that has the same size as the game tree.

We give in this chapter an exposition of the main ideas, not of all earliest or latest developments of the subject. Section 3.12 summarizes the main references. Further research is outlined in Section 3.13.

3.2 Bimatrix games and the best response condition

We use the following notation throughout. Let (A, B) be a bimatrix game, where A and B are $m \times n$ matrices of payoffs to the row player 1 and column player 2, respectively. This is a two-player game in strategic form (also called "normal form"), which is played by a simultaneous choice of a row i by player 1 and column j by player 2, who then receive payoff a_{ij} and

b_{ij}, respectively. The payoffs represent risk-neutral utilities, so when facing a probability distribution, the players want to maximize their expected payoff. These preferences do not depend on positive-affine transformations, so that A and B can be assumed to have nonnegative entries, which are rationals, or more simply integers, when A and B define the input to an algorithm.

All vectors are column vectors, so an m-vector x is treated as an $m \times 1$ matrix. A *mixed strategy* x for player 1 is a probability distribution on rows, written as an m-vector of probabilities. Similarly, a mixed strategy y for player 2 is an n-vector of probabilities for playing columns. The *support* of a mixed strategy is the set of pure strategies that have positive probability. A vector or matrix with all components zero is denoted by $\mathbf{0}$, a vector of all ones by $\mathbf{1}$. Inequalities like $x \geq \mathbf{0}$ between two vectors hold for all components. $B^\top$ is the matrix B transposed.

Let M be the set of the m pure strategies of player 1 and let N be the set of the n pure strategies of player 2. It is useful to assume that these sets are disjoint, as in

$$M = \{1, \ldots, m\}, \qquad N = \{m+1, \ldots, m+n\}. \tag{3.1}$$

Then $x \in \mathbb{R}^M$ and $y \in \mathbb{R}^N$, which means, in particular, that the components of y are y_j for $j \in N$. Similarly, the payoff matrices A and B belong to $\mathbb{R}^{M \times N}$.

A *best response* to the mixed strategy y of player 2 is a mixed strategy x of player 1 that maximizes his expected payoff $x^\top Ay$. Similarly, a best response y of player 2 to x maximizes her expected payoff $x^\top By$. A *Nash equilibrium* is a pair (x, y) of mixed strategies that are best responses to each other. The following proposition states that a mixed strategy x is a best response to an opponent strategy y if and only if all pure strategies in its support are pure best responses to y. The same holds with the roles of the players exchanged.

Proposition 3.1 (Best response condition) *Let x and y be mixed strategies of player 1 and 2, respectively. Then x is a best response to y if and only if for all $i \in M$,*

$$x_i > 0 \implies (Ay)_i = u = \max\{\, (Ay)_k \mid k \in M\}. \tag{3.2}$$

Proof $(Ay)_i$ is the ith component of Ay, which is the expected payoff to player 1 when playing row i. Then

$$x^\top Ay = \sum_{i \in M} x_i\, (Ay)_i = \sum_{i \in M} x_i\, (u - (u - (Ay)_i) = u - \sum_{i \in M} x_i\, (u - (Ay)_i).$$

So $x^\top Ay \le u$ because $x_i \ge 0$ and $u-(Ay)_i \ge 0$ for all $i \in M$, and $x^\top Ay = u$ if and only if $x_i > 0$ implies $(Ay)_i = u$, as claimed. □

Prop. 3.1 has the following intuition: Player 1's payoff $x^\top Ay$ is linear in x, so if it is maximized on a face of the simplex of mixed strategies of player 1, then it is also maximized on any vertex (that is, pure strategy) of that face, and if it is maximized on a set of vertices then it is also maximized on any convex combination of them. The proposition is useful because it states a finite condition, which is easily checked, about all pure strategies of the player, rather than about the infinite set of all mixed strategies. It can also be used algorithmically in order to find Nash equilibria, by trying out the different possible supports of mixed strategies. All pure strategies in the support must have maximum, and hence equal, expected payoff to that player. This leads to equations for the probabilities of the opponent's mixed strategy.

As an example, consider the 3×2 bimatrix game (A, B) with

$$A = \begin{bmatrix} 3 & 3 \\ 2 & 5 \\ 0 & 6 \end{bmatrix}, \qquad B = \begin{bmatrix} 3 & 2 \\ 2 & 6 \\ 3 & 1 \end{bmatrix}. \tag{3.3}$$

This game has only one pure-strategy Nash equilibrium, namely the top row (numbered 1 in the pure strategy set $M = \{1, 2, 3\}$ of player 1), together with the left column (which by (3.1) has number 4 in the pure strategy set $N = \{4, 5\}$ of player 2). A pure strategy equilibrium is given by mixed strategies of support size 1 each, so here it is the mixed strategy pair $((1, 0, 0)^\top, (1, 0)^\top)$.

The game in (3.3) has also some mixed equilibria. Any pure strategy of a player has a unique pure best response of the other player, so in any other equilibrium, each player must mix at least two pure strategies in order to fulfill condition (3.2). In particular, player 2 must be indifferent between her two columns. If the support of player 1's mixed strategy x is $\{1, 2\}$, then player 1 can make player 2 indifferent by $x_1 = 4/5$, $x_2 = 1/5$, which is the unique solution to the equations $x_1 + x_2 = 1$ and (for the two columns of B) $3x_1 + 2x_2 = 2x_1 + 6x_2$. In turn, (3.2) requires that player 2 plays with probabilities y_4 and y_5 so that player 1 is indifferent between rows 1 and 2, that is, $3y_4 + 3y_5 = 2y_4 + 5y_5$ or $(y_4, y_5) = (2/3, 1/3)$. The vector of expected payoffs to player 1 is then $Ay = (3, 3, 2)^\top$ so that (3.2) holds.

A second mixed equilibrium is $(x, y) = ((0, 1/3, 2/3)^\top, (1/3, 2/3)^\top)$ with expected payoff vectors $x^\top B = (8/3, 8/3)$ and $Ay = (3, 4, 4)^\top$. Again, the support of x only contains pure strategies i where the corresponding expected payoff $(Ay)_i$ is maximal.

A third support pair, $\{1,3\}$, for player 1, does not lead to an equilibrium, for two reasons. First, player 2 would have to play $y = (1/2, 1/2)^\top$ to make player 1 indifferent between row 1 and row 3. But then $Ay = (3, 7/2, 3)^\top$, so that rows 1 and 3 give the same payoff to player 1 but not the maximum payoff for all rows. Secondly, making player 2 indifferent via $3x_1 + 3x_3 = 2x_1 + x_3$ has the solution $x_1 = 2$, $x_3 = -1$ in order to have $x_1 + x_3 = 1$, so x is not a vector of probabilities.

In this "support testing" method, it normally suffices to consider supports of equal size for the two players. For example, in (3.3) it is not necessary to consider a mixed strategy x of player 1 where all three pure strategies have positive probability, because player 1 would then have to be indifferent between all these. However, a mixed strategy y of player 1 is already uniquely determined by equalizing the expected payoffs for two rows, and then the payoff for the remaining row is already different. This is the typical, "nondegenerate" case, according to the following definition.

Definition 3.2 A two-player game is called *nondegenerate* if no mixed strategy of support size k has more than k pure best responses.

In a *degenerate* game, Def. 3.2 is violated, for example if there is a pure strategy that has two pure best responses. For the moment, we only consider nondegenerate games, where the player's equilibrium strategies have equal sized support, which is immediate from Prop. 3.1:

Proposition 3.3 *In any Nash equilibrium (x, y) of a nondegenerate bimatrix game, x and y have supports of equal size.*

The "support testing" algorithm for finding equilibria of a nondegenerate bimatrix game then works as follows.

Algorithm 3.4 (Equilibria by support enumeration) *Input:* A nondegenerate bimatrix game. *Output:* All Nash equilibria of the game. *Method:* For each $k = 1, \ldots, \min\{m, n\}$ and each pair (I, J) of k-sized subsets of M and N, respectively, solve the equations $\sum_{i \in I} x_i b_{ij} = v$ for $j \in J$, $\sum_{i \in I} x_i = 1$, $\sum_{j \in J} a_{ij} y_j = u$ for $i \in I$, $\sum_{j \in J} y_j = 1$, and check that $x \geq \mathbf{0}$, $y \geq \mathbf{0}$, and that (3.2) holds for x and analogously y.

The linear equations considered in this algorithm may not have solutions, which then mean no equilibrium for that support pair. Nonunique solutions occur only for degenerate games, because a linear dependency allows to reduce the support of a mixed strategy. Degenerate games are discussed in Section 3.6 below.

3.3 Equilibria via labeled polytopes

In order to identify the possible supports of equilibrium strategies, one can use "best response polytopes" that express directly the inequalities of best responses and nonnegative probabilities.

We first recall some notions from the theory of (convex) polyhedra. An *affine combination* of points $z_1, \ldots, z_k$ in some Euclidean space is of the form $\sum_{i=1}^{k} z_i \lambda_i$ where $\lambda_1, \ldots, \lambda_k$ are reals with $\sum_{i=1}^{k} \lambda_i = 1$. It is called a *convex combination* if $\lambda_i \geq 0$ for all i. A set of points is *convex* if it is closed under forming convex combinations. Given points are *affinely independent* if none of these points is an affine combination of the others. A convex set has *dimension* d if and only if it has $d+1$, but no more, affinely independent points.

A *polyhedron* P in $\mathbb{R}^d$ is a set $\{z \in \mathbb{R}^d \mid Cz \leq q\}$ for some matrix C and vector q. It is called *full-dimensional* if it has dimension d. It is called a *polytope* if it is bounded. A *face* of P is a set $\{\, z \in P \mid c^\top z = q_0\}$ for some $c \in \mathbb{R}^d$, $q_0 \in \mathbb{R}$ so that the inequality $c^\top z \leq q_0$ holds for all z in P. A *vertex* of P is the unique element of a 0-dimensional face of P. An *edge* of P is a one-dimensional face of P. A *facet* of a d-dimensional polyhedron P is a face of dimension $d-1$. It can be shown that any nonempty face F of P can be obtained by turning some of the inequalities defining P into equalities, which are then called *binding* inequalities. That is, $F = \{\, z \in P \mid c_i z = q_i,\ i \in I\}$, where $c_i z \leq q_i$ for $i \in I$ are some of the rows in $Cz \leq q$. A facet is characterized by a single binding inequality which is *irredundant*, that is, the inequality cannot be omitted without changing the polyhedron. A d-dimensional polyhedron P is called *simple* if no point belongs to more than d facets of P, which is true if there are no special dependencies between the facet-defining inequalities.

The "best response polyhedron" of a player is the set of that player's mixed strategies together with the "upper envelope" of expected payoffs (and any larger payoffs) to the *other* player. For player 2 in the example (3.3), it is the set $\overline{Q}$ of triples (y_4, y_5, u) that fulfill $3y_4 + 3y_5 \leq u$, $2y_4 + 5y_5 \leq u$, $0y_4 + 6y_5 \leq u$, $y_4 \geq 0$, $y_5 \geq 0$, and $y_4 + y_5 = 1$. The first three inequalities, in matrix notation $Ay \leq \mathbf{1}u$, say that u is at least as large as the expected payoff for each pure strategy of player 1. The other constraints $y \geq \mathbf{0}$ and $\mathbf{1}^\top y = 1$ state that y is a vector of probabilities. The best response polyhedron $\overline{P}$ for player 1 is defined analogously. Generally,

$$\begin{aligned} \overline{P} &= \{(x, v) \in \mathbb{R}^M \times \mathbb{R} \mid x \geq \mathbf{0},\ \mathbf{1}^\top x = 1,\ B^\top x \leq \mathbf{1}v\,\}, \\ \overline{Q} &= \{(y, u) \in \mathbb{R}^N \times \mathbb{R} \mid Ay \leq \mathbf{1}u,\ y \geq \mathbf{0},\ \mathbf{1}^\top y = 1\,\}. \end{aligned} \tag{3.4}$$

The left picture in Fig. 3.1 shows $\overline{Q}$ for our example, for $0 \le y_4 \le 1$ which uniquely determines y_5 as $1 - y_4$. The circled numbers indicate the facets of $\overline{Q}$, which are either the strategies $i \in M$ of the other player 1 or the own strategies $j \in N$. Facets 1, 2, 3 of player 1 indicate his best responses together with his expected payoff u. For example, 1 is a best response when $y_4 \ge 2/3$. Facets 4 and 5 of player 2 tell when the respective own strategy has probability zero, namely $y_4 = 0$ or $y_5 = 0$.

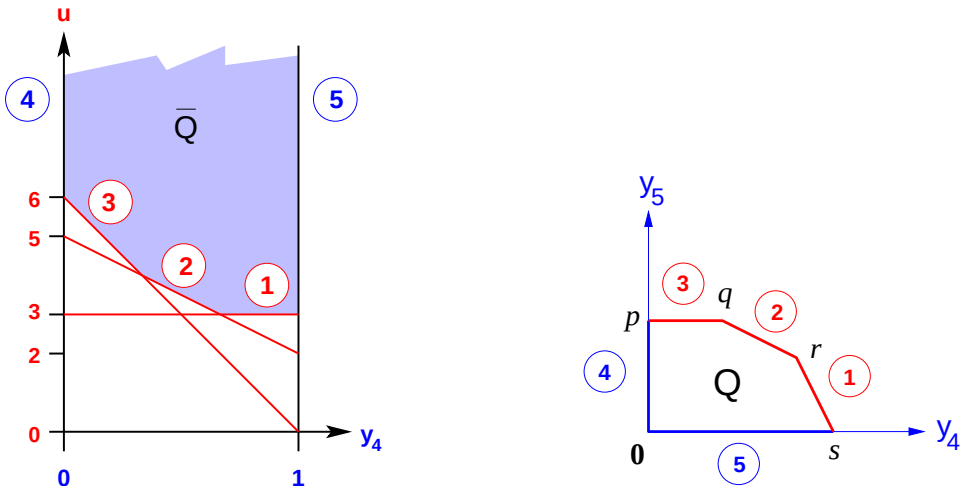


Fig. 3.1. Best reponse polyhedron $\overline{Q}$ for strategies of player 2, and corresponding polytope Q, which has vertices $\mathbf{0}, p, q, r, s$.

We say a point (y, u) of $\overline{Q}$ has *label* $k \in M \cup N$ if the kth inequality in the definition of $\overline{Q}$ is binding, which for $k = i \in M$ is the ith binding inequality $\sum_{j \in N} a_{ij} y_j = u$ (meaning i is a best response to y), or for $k = j \in N$ the binding inequality $y_j = 0$. In the example, $(y_4, y_5, u) = (2/3, 1/3, 3)$ has labels 1 and 2, so rows 1 and 2 are best responses to y with expected payoff 3 to player 1. The labels of a point (x, v) of $\overline{P}$ are defined correspondingly: It has label $i \in M$ if $x_i = 0$, and label $j \in N$ if $\sum_{i \in M} b_{ij} x_i = v$. With these labels, an equilibrium is a pair (x, y) of mixed strategies so that with the corresponding expected payoffs v and u, the pair $((x, v), (y, u))$ in $\overline{P} \times \overline{Q}$ is *completely labeled*, which means that every label $k \in M \cup N$ appears either as a label of (x, v) or of (y, u). This is equivalent to the best response condition (3.2): A missing label would mean a pure strategy of a player, for example i of player 1, that does not have probability zero, so $x_i > 0$, and is also not a best response, since $\sum_{j \in N} a_{ij} y_j < u$, because the respective inequality i is not binding in $\overline{P}$ or $\overline{Q}$. But this is exactly when the best response condition is violated. Conversely, if every label appears in $\overline{P}$ or $\overline{Q}$, then each pure

strategy is a best response or has probability zero, so x and y are mutual best responses.

The constraints (3.4) that define $\overline{P}$ and $\overline{Q}$ can be simplified by eliminating the payoff variables u and v, which works if these are always positive. For that purpose, assume that

$$A \text{ and } B^\top \text{ are nonnegative and have no zero column.} \tag{3.5}$$

We could simply assume $A > \mathbf{0}$ and $B > \mathbf{0}$, but it is useful to admit zero matrix entries (e.g. as in the identity matrix); even negative entries are possible as long as the upper envelope remains positive, for example for a_{34} (currently zero) in (3.3), as Fig. 3.1 shows.

For $\overline{P}$, we divide each inequality $\sum_{i\in M} b_{ij}x_i \le v$ by v, which gives $\sum_{i\in M} b_{ij}(x_i/v) \le 1$, treat x_i/v as a new variable that we call again x_i, and call the resulting polyhedron P. Similarly, $\overline{Q}$ is replaced by Q by dividing each inequality in $Ay \le \mathbf{1}u$ by u. Then

$$\begin{aligned} P &= \{\, x \in \mathbb{R}^M \mid \quad x \ge \mathbf{0},\ B^\top x \le \mathbf{1}\}, \\ Q &= \{\, y \in \mathbb{R}^N \mid Ay \le \mathbf{1}, \qquad y \ge \mathbf{0}\}\,. \end{aligned} \tag{3.6}$$

It is easy to see that (3.5) implies that P and Q are full-dimensional polytopes, unlike $\overline{P}$ and $\overline{Q}$. In effect, we have normalized the expected payoffs to be 1, and dropped the conditions $1^\top x = 1$ and $1^\top y = 1$. Nonzero vectors $x \in P$ and $y \in Q$ are multiplied by $v = 1/\mathbf{1}^\top x$ and $u = 1/\mathbf{1}^\top y$ to turn them into probability vectors. The scaling factors v and u are the expected payoffs to the other player.

The set $\overline{P}$ is in one-to-one correspondence with $P - \{\mathbf{0}\}$ with the map $(x, v) \mapsto x \cdot (1/v)$. Similarly, $(y, u) \mapsto y \cdot (1/u)$ defines a bijection $\overline{Q} \to Q - \{\mathbf{0}\}$. These bijections are not linear, but are known as "projective transformations" (for a visualization see [vS02, Fig. 2.5]). They preserve the face incidences since a binding inequality in $\overline{P}$ (respectively, $\overline{Q}$) corresponds to a binding inequality in P (respectively, Q) and vice versa. In particular, points have the same *labels* defined by the binding inequalities, which are some of the $m + n$ inequalities defining P and Q in (3.6). An equilibrium is then a completely labeled pair $(x, y) \in P \times Q - \{(\mathbf{0}, \mathbf{0})\}$ that has for each label $i \in M$ the respective binding inequality in $x \ge \mathbf{0}$ or $Ay \le \mathbf{1}$, and for each $j \in N$ the respective binding inequality in $B^\top x \le \mathbf{1}$ or $y \ge \mathbf{0}$.

For the example (3.3), the polytope Q is shown on the right in Fig. 3.1 and in Fig. 3.2. The vertices y of Q, written as $y^\top$, are $(0, 0)$ with labels $4, 5$, vertex $p = (0, 1/6)$ with labels $3, 4$, vertex $q = (1/12, 1/6)$ with labels $2, 3$, vertex $r = (1/6, 1/9)$ with labels $1, 2$, and $s = (1/3, 0)$ with labels $1, 5$. The polytope P is shown on the left in Fig. 3.2. Its vertices x are $\mathbf{0}$ with

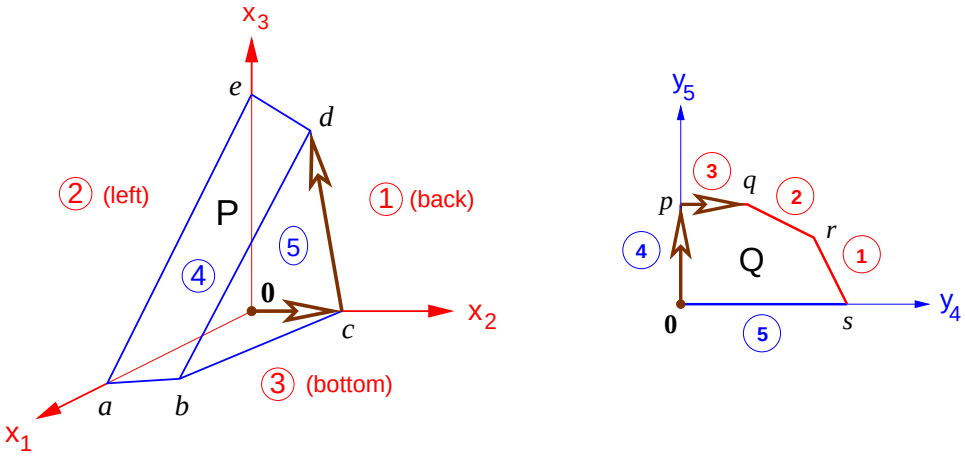


Fig. 3.2. The best response polytopes P (with vertices $\mathbf{0}, a, b, c, d, e$) and Q for the game in (3.3). The arrows describe the Lemke–Howson algorithm, see Section 3.4.

labels $1, 2, 3$, and (written as $x^\top$) vertex $a = (1/3, 0, 0)$ with labels $2, 3, 4$, vertex $b = (2/7, 1/14, 0)$ with labels $3, 4, 5$, vertex $c = (0, 1/6, 0)$ with labels $1, 3, 5$, vertex $d = (0, 1/8, 1/4)$ with labels $1, 4, 5$, and $e = (0, 0, 1/3)$ with labels $1, 2, 4$. Note that the vectors alone show only the "own" labels as the unplayed own strategies; the information about the other player's best responses is important as well. The following three completely labeled vertex pairs define the Nash equilibria of the game, which we already found earlier: the pure strategy equilibrium (a, s), and the mixed equilibria (b, r) and (d, q). The vertices c and e of P, and p of Q, are not part of an equilibrium.

Nondegeneracy of a bimatrix game (A, B) can be stated in terms of the polytopes P and Q in (3.6) as follows: no point in P has more than m labels, and no point in Q has more than n labels. (If $x \in P$ and x has support of size k and L is the set of labels of x, then $|L \cap M| = m - k$, so $|L| > m$ implies x has more than k best responses in $L \cap N$.) Then P and Q are simple polytopes, because a point of P, say, that is on more than m facets would have more than m labels. Even if P and Q are simple polytopes, the game can be degenerate if the *description* of a polytope is redundant in the sense that some inequality can be omitted, but nevertheless is sometimes binding. This occurs if a player has a pure strategy that is weakly dominated by or payoff equivalent to some other mixed strategy. Non-simple polytopes or redundant inequalities of this kind do not occur for "generic" payoffs; this illustrates the assumption of nondegeneracy from a geometric viewpoint. (A strictly dominated strategy may occur generically, but it defines a redundant inequality that is never binding, so this does not lead to a degenerate game.)

Because the game is nondegenerate, only vertices of P can have m labels, and only vertices of Q can have n labels. Otherwise, a point of P with m labels that is not a vertex would be on a higher-dimensional face, and a vertex of that face, which is a vertex of P, would have additional labels. Consequently, only vertices of P and Q have to be inspected as possible equilibrium strategies.

Algorithm 3.5 (Equilibria by vertex enumeration) *Input:* A nondegenerate bimatrix game. *Output:* All Nash equilibria of the game. *Method:* For each vertex x of $P - \{\mathbf{0}\}$, and each vertex y of $Q - \{\mathbf{0}\}$, if (x, y) is completely labeled, output the Nash equilibrium $(x \cdot 1/\mathbf{1}^\top x, y \cdot 1/\mathbf{1}^\top y)$.

Algorithm 3.5 is superior to the support enumeration algorithm 3.4 because there are more supports than vertices. For example, if $m = n$, then approximately 4^n possible support pairs have to be tested, but P and Q have less than 2.6^n many vertices, by the "upper bound theorem" for polytopes. This entails less work, assuming that complementary vertex pairs (x, y) are found efficiently.

Enumerating all vertices of a polytope P, say, is a standard problem in computional geometry. The elegant *lrs* (lexicographic reverse search) algorithm considers a known vertex, like $\mathbf{0}$ for P in (3.6), and a linear objective function that, over P, is maximized at that vertex, like the function $x \mapsto -\mathbf{1}^\top x$. For any vertex of P, the simplex algorithm with a unique pivoting rule (for example, Bland's least-index rule for choosing the entering and leaving variable) then generates a unique path to $\mathbf{0}$, defining a directed tree on the vertices of P with root $\mathbf{0}$. The algorithm explores that tree by a depth-first search from $\mathbf{0}$ which "reverts" the simplex steps by considering recursively for each vertex x of P the edges to vertices x' so that the simplex algorithm pivots from x' to x.

3.4 The Lemke–Howson algorithm

Algorithms 3.4 and 3.5 find all Nash equilibria of a nondegenerate bimatrix game (A, B). In contrast, the Lemke–Howson (for short LH) algorithm finds one Nash equilibrium, and provides an elementary proof that Nash equilibria exist. The LH algorithm follows a path (called LH path) of vertex pairs (x, y) of $P \times Q$, for the polytopes P and Q defined in (3.6), that starts at $(\mathbf{0}, \mathbf{0})$ and ends at a Nash equilibrium.

An LH path alternately follows edges of P and Q, keeping the vertex in the other polytope fixed. Because the game is nondegenerate, a vertex of P

is given by m labels, and a vertex of Q is given by n labels. An edge of P is defined by $m-1$ labels. For example, in Fig. 3.2 the edge defined by labels 1 and 3 joins the vertices $\mathbf{0}$ and c. *Dropping* a label l of a vertex x of P, say, means traversing the unique edge that has all the labels of x except for l. For example, dropping label 2 of the vertex $\mathbf{0}$ of P in Fig. 3.2 gives the edge, defined by labels 1 and 3, that joins $\mathbf{0}$ to vertex c. The endpoint of the edge has a new label, which is said to be *picked up*, so in the example label 5 is picked up at vertex c.

The LH algorithm starts from $(\mathbf{0}, \mathbf{0})$ in $P \times Q$. This is called the *artificial equilibrium*, which is a completely labeled vertex pair because every pure strategy has probability zero. It does not represent a Nash equilibrium of the game because the zero vector cannot be rescaled to a mixed strategy vector. An initial free choice of the LH algorithm is a pure strategy k of a player (any label in $M \cup N$), called the *missing label*. Starting with $(x, y) = (\mathbf{0}, \mathbf{0})$, label k is dropped. At the endpoint of the corresponding edge (of P if $k \in M$, of Q if $k \in N$), the new label that is picked up is *duplicate* because it was present in the other polytope. That duplicate label is then dropped in the other polytope, picking up a new label. If the newly picked label is the missing label, the algorithm terminates and has found a Nash equilibrium. Otherwise, the algorithm repeats by dropping the duplicate label in the other polytope, and continues in this fashion.

In the example (3.3), suppose the missing label is $k = 2$. The polytopes P and Q are shown in Fig. 3.2. Starting from $\mathbf{0}$ in P, label 2 is dropped, traversing the edge from $\mathbf{0}$ to vertex c, which is set of points x of P that have labels 1 and 3, shown by an arrow in Fig. 3.2. The endpoint c of that edge has label 5 which is picked up. At the vertex pair $(c, \mathbf{0})$ of $P \times Q$, all labels except for the missing label 2 are present, so label 5 is now duplicate because it is both a label of c and of $\mathbf{0}$. The next step is therefore to drop the duplicate label 5 in Q, traversing the edge from $\mathbf{0}$ to vertex p while keeping c in P fixed. The label that is picked up at vertex p is 3, which is now duplicate. Dropping label 3 in P defines the unique edge defined by labels 1 and 5, which joins vertex c to vertex d. At vertex d, label 4 is picked up. Dropping label 4 in Q means traversing the edge of Q from p to q. At vertex q, label 2 is picked up. Because 2 is the missing label, the current vertex pair (d, q) is completely labeled, and it is the Nash equilibrium found by the algorithm.

In terms of the game, the first two LH steps amount to taking a pure strategy (given by the missing label k, say of player 1) and considering its best response, say j, which defines a pure strategy pair (k, j). If this is not already an equilibrium, the best response i to j is not k, so that i is a

duplicate label, and is now given positive probability in addition to k. In general, a duplicate label is either a new best response which in the next step gets positive probability, as in this case. Alternatively, the duplicate label is a pure strategy whose probability has just become zero, so that it no longer needs to be maintained as a best response in the other polytope and the path moves away from the best response facet.

Algorithm 3.6 (Lemke–Howson) *Input:* Nondegenerate bimatrix game. *Output:* One Nash equilibrium of the game. *Method:* Choose $k \in M \cup N$, called the *missing label*. Let $(x, y) = (\mathbf{0}, \mathbf{0}) \in P \times Q$. Drop label k (from x in P if $k \in M$, from y in Q if $k \in N$). **Loop:** Call the new vertex pair (x, y). Let l be the label that is picked up. If $l = k$, terminate with Nash equilibrium (x, y) (rescaled as mixed strategy pair). Otherwise, drop l in the other polytope and repeat.

The LH algorithm terminates, and finds a Nash equilibrium, because $P \times Q$ has only finitely many vertex pairs. The next vertex pair on the path is always unique. Hence, a given vertex pair cannot be revisited because that would provide an additional possibility to proceed in the first place.

We have described the LH path for missing label k by means of alternating edges between two polytopes. In fact, it is a path on the product polytope $P \times Q$, given by the set of pairs (x, y) of $P \times Q$ that are *k-almost completely labeled*, meaning every label in $M \cup N - \{k\}$ appears as a label of either x or y. In Fig. 3.2 for $k = 2$, the vertex pairs on the path are $(\mathbf{0}, \mathbf{0})$, $(c, \mathbf{0})$, (c, p), (d, p), (d, q).

For a fixed missing label k, the k-almost completely labeled vertices and edges of the product polytope $P \times Q$ form a graph of degree 1 or 2. Clearly, such a graph consists of disjoints paths and cycles. The endpoints of the paths are completely labeled. They are the Nash equilibria of the game and the artificial equilibrium $(\mathbf{0}, \mathbf{0})$. Since the number of endpoints of the paths is even, we obtain the following.

Corollary 3.7 *A nondegenerate bimatrix game has an odd number of Nash equilibria.*

The LH algorithm can start at any Nash equilibrium, not just the artificial equilibrium. In Fig. 3.2 with missing label 2, starting the algorithm at the Nash equilibrium (d, q) would just generate the known LH path backwards to $(\mathbf{0}, \mathbf{0})$. When started at the Nash equilibrium (a, s), the LH path for the missing label 2 gives the vertex pair (b, s), where label 5 is duplicate, and then the equilibrium (b, r). This path cannot go back to $(\mathbf{0}, \mathbf{0})$ because the

path leading to $(\mathbf{0}, \mathbf{0})$ starts at (d, q). This gives the three Nash equilibria of the game as endpoints of the two LH paths for missing label 2.

These three equilibria can also be found by the LH algorithm by varying the missing label. For example, the LH path for missing label 1 in Fig. 3.2 leads to (a, s), from which (b, r) is subsequently found via missing label 2. taking a pure strategy k and its best response.)

However, some Nash equilibria can remain elusive to the LH algorithm. An example is the following symmetric 3×3 game with

$$A = B^\top = \begin{bmatrix} 3 & 3 & 0 \\ 4 & 0 & 1 \\ 0 & 4 & 5 \end{bmatrix}. \tag{3.7}$$

Every Nash equilibrium (x, y) of this game is symmetric, that is, $x = y$, where $x^\top$ is either $(0, 0, 1)$, $(1/2, 1/4, 1/4)$, or $(3/4, 1/4, 0)$. Only the first of these is found by the LH algorithm, for any missing label; because the game is symmetric, it suffices to consider the missing labels $1, 2, 3$. (A symmetric game remains unchanged when the players are exchanged; a symmetric game has always a symmetric equilibrium, but may also have nonsymmetric equilibria, which obviously come in pairs.)

3.5 Integer pivoting

The LH algorithm follows the edges of a polyhedron, which is implemented algebraically by *pivoting* as used by the simplex algorithm for solving a linear program. We describe an efficient implementation that has no numerical errors by storing *integers* of arbitrary precision. The constraints defining the polyhedron are thereby represented as linear equations with nonnegative *slack* variables. For the polytopes P and Q in (3.6), these slack variables are nonnegative vectors $s \in \mathbb{R}^N$ and $r \in \mathbb{R}^M$ so that $x \in P$ and $y \in Q$ if and only if

$$B^\top x + s = \mathbf{1}, \qquad r + Ay = \mathbf{1} \tag{3.8}$$

and

$$x \geq \mathbf{0}, \quad s \geq \mathbf{0}, \quad r \geq \mathbf{0}, \quad y \geq \mathbf{0}. \tag{3.9}$$

A binding inequality corresponds to a zero slack variable. The pair (x, y) is completely labeled if and only if $x_i r_i = 0$ for all $i \in M$ and $y_j s_j = 0$ for all $j \in N$, which by (3.9) can be written as the orthogonality condition

$$x^\top r = 0, \qquad y^\top s = 0. \tag{3.10}$$

A *basic* solution to (3.8) is given by n *basic* (linearly independent) columns of $B^\top x + s = \mathbf{1}$ and m basic columns of $r + Ay = \mathbf{1}$ where the *nonbasic* variables that correspond to the m respectively n other (nonbasic) columns are set to zero, so that the basic variables are uniquely determined. A basic *feasible* solution also fulfills (3.9), and defines a vertex x of P and y of Q. The labels of such a vertex are given by the respective nonbasic columns.

Pivoting is a change of the basis where a nonbasic variable *enters* and a basic variable *leaves* the set of basic variables, while preserving feasibility (3.9). We illustrate this for the edges of the polytope P in Fig. 3.2 shown as arrows, which are the edges that connect $\mathbf{0}$ to vertex c, and c to d. The system $B^\top x + s = \mathbf{1}$ is here

$$\begin{aligned} 3x_1 + \phantom{\boxed{6}}2x_2 + 3x_3 + s_4 &= 1 \\ 2x_1 + \boxed{6}x_2 + x_3 + s_5 &= 1 \end{aligned} \tag{3.11}$$

and the basic variables in (3.11) are s_4 and s_5, defining the basic feasible solution $s_4 = 1$ and $s_5 = 1$ which is simply the right hand side of (3.11) because the basic columns form the identity matrix. Dropping label 2 means x_2 is no longer a nonbasic variable, so x_2 enters the basis. Increasing x_2 while maintaining (3.11) changes the current basic variables as $s_4 = 1 - 2x_2$, $s_5 = 1 - 6x_2$, and these stay nonnegative as long as $x_2 \leq 1/6$. The term $1/6$ is the *minimum ratio*, over all rows in (3.11) with positive coefficients of the entering variable x_2, of the right hand side divided by the coefficient. (Only positive coefficients bound the increase of x_2, which applies to at least one row since the polyhedron P is bounded.) The minimum ratio test determines uniquely s_5 as the variable that leaves the basis, giving the label 5 that is picked up in that step. The respective coefficient 6 of x_2 is indicated by a box in (3.11), and is called the *pivot element*; its row is the pivot row and its column is the pivot column.

Algebraically, pivoting is done by applying row operations to (3.11) so that the new basic variable x_2 has a unit column, so that the basic solution is again given by the right hand side. *Integer* pivoting is a way to achieve this while keeping all coefficients of the system as integers; the basic columns then form an identity matrix multiplied by an integer. To that end, all rows (which in (3.11) is only the first row) except for the pivot row are multiplied with the pivot element, giving the intermediate system

$$\begin{aligned} 18x_1 + 12x_2 + 18x_3 + 6s_4 &= 6 \\ 2x_1 + 6x_2 + x_3 + s_5 &= 1 \end{aligned} \tag{3.12}$$

Then, suitable multiples of the pivot row are subtracted from the other rows

to obtain zero entries in the pivot column, giving the new system

$$\begin{array}{rrrrrrrrr} 14x_1 & & & + & \boxed{16}x_3 & + & 6s_4 & - & 2s_5 = 4 \\ 2x_1 & + & 6x_2 & + & x_3 & & + & s_5 & = 1 \end{array} \tag{3.13}$$

In (3.13), the basic columns for the basic variables s_4 and x_2 form the identity matrix, multiplied by 6 (which is pivot element that has just been used). Clearly, all matrix entries are integers. The next step of the LH algorithm in the example is to let y_5 be the entering variable in the system $r + Ay = \mathbf{1}$, which we do not show. There, the leaving variable is r_3 (giving the duplicate label 3), so that the next entering variable in (3.13) is x_3. The minimum ratio test (which can be performed using only multiplications, not divisions) shows that among the nonnegativity constraints $6s_4 = 4 - 16x_3 \geq 0$ and $6x_2 = 1 - x_3 \geq 0$, the former is tighter, so that s_4 is the leaving variable. The pivot element, shown by a box in (3.13), is 16, with the first row as pivot row.

The integer pivoting step is to multiply the other rows with the pivot element, giving

$$\begin{array}{rrrrrrrrr} 14x_1 & & & + & 16x_3 & + & 6s_4 & - & 2s_5 = 4 \\ 32x_1 & + & 96x_2 & + & 16x_3 & & & + & 16s_5 = 16 \end{array} \tag{3.14}$$

Subsequently, a suitable multiple of the pivot row is subtracted from each other row, giving the new system

$$\begin{array}{rrrrrrrrr} 14x_1 & & & + & 16x_3 & + & 6s_4 & - & 2s_5 = 4 \\ 18x_1 & + & 96x_2 & & & - & 6s_4 & + & 18s_5 = 12 \end{array} \tag{3.15}$$

with x_3 and x_2 as basic variables. However, except for the pivot row, the unchanged basic variables have larger coefficients than before, because they have been multiplied with the new pivot element 16. The second row in (3.15) can now be divided by the previous pivot element 6, and this division is integral for *all* coefficients in that row; this is the key feature of integer pivoting, explained shortly. The new system is

$$\begin{array}{rrrrrrrrr} 14x_1 & & & + & 16x_3 & + & 6s_4 & - & 2s_5 = 4 \\ 3x_1 & + & 16x_2 & & & - & s_4 & + & 3s_5 = 2 \end{array} \tag{3.16}$$

This is the final system because the duplicate label 4 (given by the variable s_4 that has just left) is dropped in Q, where the missing label 2 is picked up. The basic solution in (3.16) is vertex d of P with $x_3 = 4/16$, $x_2 = 2/16$ and labels (given by the nonbasic columns) 1, 4, and 5.

Integer pivoting, as illustrated in this example, always maintains an integer matrix (or "tableau") of coefficients of a system of linear equations that

is equivalent to the original system $B^\top x + s = \mathbf{1}$, in the form

$$CB^\top x + Cs = C\mathbf{1}. \tag{3.17}$$

In (3.17), C is the inverse of the *basis matrix* given by the basic columns of the original system, multiplied by the *determinant* of the basis matrix (which is 6 in (3.13), and 16 in (3.16)). The matrix C is given by the (integer) cofactors of the basis matrix; the cofactor of a matrix entry is the determinant of the matrix when the row and column of that element are deleted. Each entry in (3.17) has a bounded number of digits (by at most a factor of $n \log n$ compared to the original matrix entries), so integer pivoting is a polynomial-time algorithm. It is also superior to using fractions of integers (rational numbers) because their cancelation requires greatest common divisor computations that take the bulk of computation time. Only the final fractions defining the solution, like $x_3 = 4/16$ and $x_2 = 2/16$ in (3.16), may have to be canceled.

3.6 Degenerate games

The uniqueness of a LH path requires a nondegenerate game. In a degenerate game, a vertex of P, for example, may have more than m labels. When that vertex is represented as a basic feasible solution as in (3.17) this means that not only the m nonbasic variables are zero, but also at least one basic variable. Such a degenerate basic feasible solution results from a pivoting step where the leaving variable (representing the label that is picked up) is not unique.

As an example, consider the 3×2 game

$$A = \begin{bmatrix} 3 & 3 \\ 2 & 5 \\ 0 & 6 \end{bmatrix}, \qquad B = \begin{bmatrix} 3 & 3 \\ 2 & 6 \\ 3 & 1 \end{bmatrix}, \tag{3.18}$$

which agrees with (3.3) except that $b_{15} = 3$. The polytope Q for this game is the same as before, shown on the right in Fig. 3.2. The polytope P is the convex hull of the original vertices $\mathbf{0}, a, c, d, e$ shown on the left in Fig. 3.2, so vertex b has merged with a. The new facets of P with labels 4 and 5 are triangles with vertices a, d, e and a, c, d, respectively.

In this example (3.18), the first step of the LH path for missing label 1 would be from $(\mathbf{0}, \mathbf{0})$ to $(a, \mathbf{0})$, where the two labels 4 and 5 are picked up, because vertex a has the four labels $2, 3, 4, 5$ due to the degeneracy. If then label 4 is dropped in Q, the algorithm finds the equilibrium (a, s) and no problem occurs. However, dropping label 5 in Q would mean a move

to (a, p) where label 3 is picked up, and none of the two edges of P that move away from the facet with label 3 (which are the edges from a to d and from a to e) would, together with p, be 1-almost completely labeled, so the algorithm fails at this point.

Degeneracy can be resolved by *perturbing* the linear system *lexicographically*, which is well known from linear programming. Assume that the system $B^\top x + s = \mathbf{1}$, say, is changed to the perturbed system $B^\top x + s = \mathbf{1} + (\varepsilon^1, \ldots, \varepsilon^n)^\top$. After any number of pivoting steps, this system has the form

$$CB^\top x + Cs = C\mathbf{1} + C(\varepsilon^1, \ldots, \varepsilon^n)^\top \tag{3.19}$$

for some invertible matrix C. The corresponding unperturbed basic feasible solution may have a zero basic variable, which is a row of $C\mathbf{1}$, but for sufficiently small $\varepsilon > 0$ it is positive if and only if in that row the first nonzero entry of the matrix C is positive; this is the invariant maintained by the algorithm, using a more general "lexico-minimum" ratio test. No actual perturbance is required, and C is already stored in the system as the matrix of coefficients of s, as seen from (3.19).

Degenerate games may have infinite sets of equilibria. In the example (3.18), vertex a of P, which represents the pure strategy $(1, 0, 0)^\top$ of player 1, together with the entire edge that joins vertices r and s of C, defines a *component* of Nash equilibria, where player 2 plays some mixed strategy $(y_4, 1 - y_4)$ for $2/3 \leq y_4 \leq 1$. However, this equilibrium component is a convex combination of the "extreme" equilibria (a, r) and (a, s). In general, even in a degenerate game, the Nash equilibria can be described in terms of pairs of vertices of P and Q. We write $\operatorname{conv} U$ for the convex hull of a set U.

Proposition 3.8 *Let (A, B) be a bimatrix game, and $(x, y) \in P \times Q$. Then (x, y) (rescaled) is a Nash equilibrium if and only if there is a set U of vertices of $P - \{\mathbf{0}\}$ and a set V of vertices of $Q - \{\mathbf{0}\}$ so that $x \in \operatorname{conv} U$ and $y \in \operatorname{conv} V$, and every $(u, v) \in U \times V$ is completely labeled.*

Prop. 3.8 holds because labels are preserved under convex combinations, and because every face of P or Q has the labels of its vertices, which are vertices of the entire polytope; for details see [vS02, Thm. 2.14].

The following algorithm, which extends Algorithm 3.5, outputs a complete description of all Nash equilibria of a bimatrix game: Define a bipartite graph on the vertices of $P - \{\mathbf{0}\}$ and $Q - \{\mathbf{0}\}$ whose edges are the completely labeled vertex pairs (x, y). The "cliques" (maximal complete bipartite subgraphs) of this graph of the form $U \times V$ then define sets of Nash equilibria $\operatorname{conv} U \times \operatorname{conv} V$ whose union is the set of all Nash equilibria. These sets are

called "maximal Nash subsets". They may be nondisjoint, if they contain common points (x, y). The connected unions of these sets are usually called the (topological) *components* of Nash equilibria.

3.7 Extensive games and their strategic form

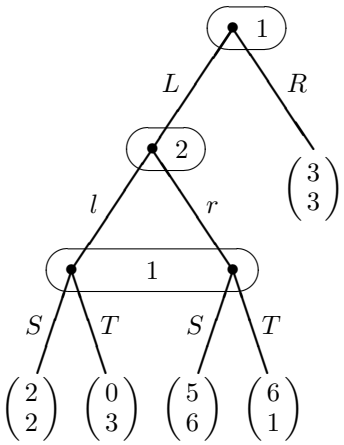
A game in strategic form is a "static" description of an interactive situation, where players act simultaneously. A detailed "dynamic" description is an *extensive* game where players act sequentially, where some moves can be made by a *chance* player, and where each player's *information* about earlier moves is modeled in detail. Extensive games are a fundamental representation of dynamic interactions which generalizes other models like repeated and multistage games, or games with incomplete information.

The basic structure of an extensive game is a directed *tree*. The nodes of the tree represent game *states*. Trees (rather than general graphs) are used because then a game state encodes the full history of play. Only one player moves at any one state along a tree edge. The game starts at the root (initial node) of the tree and ends at a leaf (terminal node), where each player receives a *payoff*. The nonterminal nodes are called *decision nodes*. A player's possible *moves* are assigned to the outgoing edges of the decision node.

The decision nodes are partitioned into *information sets*. All nodes in an information set belong to the same player, and have the same moves. The interpretation is that when a player makes a move, he only knows the information set but not the particular node he is at. In a game with *perfect information*, all information sets are singletons (and can therefore be omitted). We denote the set of information sets of player i by H_i, information sets by h, and the set of moves at h by C_h.

Fig. 3.3 shows an example of an extensive game. Moves are marked by upper case letters for player 1 and by lower case letters for player 2. Information sets are indicated by ovals. The two information sets of player 1 have move sets $\{L, R\}$ and $\{S, T\}$, and the information set of player 2 has move set $\{l, r\}$. A play of the game may proceed by player 1 choosing L, player 2 choosing r, and player 1 choosing S, after which the game terminates with payoffs 5 and 6 to player 1 and 2. By definition, move S of player 1 is the same no matter whether player 2 has chosen l or r, because player 1 does not know the game state in his second information set.

At some decision nodes, the next move may be a *chance* move. Chance is here treated as an additional player 0 who receives no payoff and who plays according to a known *behavior strategy*. A behavior strategy of player i is



$$A = \begin{array}{c|cc|c} & l & r & \\ \hline & 2 & 5 & \langle L,S\rangle \\ & 0 & 6 & \langle L,T\rangle \\ & 3 & 3 & \langle R,S\rangle \\ & 3 & 3 & \langle R,T\rangle \end{array} \qquad B = \begin{array}{c|cc|c} & l & r & \\ \hline & 2 & 6 & \langle L,S\rangle \\ & 3 & 1 & \langle L,T\rangle \\ & 3 & 3 & \langle R,S\rangle \\ & 3 & 3 & \langle R,T\rangle \end{array}$$

$$A = \begin{array}{c|ccc|c} & \emptyset & l & r & \\ \hline & & & & \emptyset \\ & & & & L \\ & 3 & & & R \\ & & 2 & 5 & LS \\ & & 0 & 6 & LT \end{array} \qquad B = \begin{array}{c|ccc|c} & \emptyset & l & r & \\ \hline & & & & \emptyset \\ & & & & L \\ & 3 & & & R \\ & & 2 & 6 & LS \\ & & 3 & 1 & LT \end{array}$$

Fig. 3.3. Left: A game in extensive form. Top right: Its *strategic form* payoff matrices A and B. Bottom right: Its *sequence form* payoff matrices A and B, where rows and columns correspond to the sequences of the players which are marked at the side. Any sequence pair not leading to a leaf has matrix entry zero, which is left blank.

given by a probability distribution on C_h for all h in H_i. (The information sets belonging to the chance player are singletons.) A *pure strategy* is a behavior strategy where each move is picked deterministically. A pure strategy of player i can be regarded as an element $\langle c_h\rangle_{h\in H_i}$ of $\prod_{h\in H_i} C_h$, that is, as a tuple of moves, like $\langle L,S\rangle$ for player 1 in Fig. 3.3.

Tabulating all pure strategies of the players and recording the resulting expected payoffs defines the *strategic form* of the game. In Fig. 3.3, the strategic form of the extensive game is shown at the top right, with payoff matrices A and B to player 1 and player 2.

Given the strategic form, a player can play according to a *mixed strategy*, which is a probability distribution on pure strategies. The player chooses a pure strategy, which is a complete plan of action, according to this distribution, and plays it in the game. In contrast, a behavior strategy can be played by "delaying" the random move until the player reaches the respective information set. It can be considered as a special mixed strategy since it defines a probability for every pure strategy, where the moves at information sets are chosen independently.

We consider algorithms for finding Nash equilibria of an extensive game, with the tree together with the described game data as input. The strategic form is bad for this purpose because it is typically exponentially large in the

game tree. As described in the subsequent sections, this complexity can be reduced, in some cases by considering *subgames* and corresponding *subgame perfect* equilibria. The *reduced* strategic form of the game it smaller but may still be exponentially large. A reduction from exponential to linear size is provided by the *sequence form*, which allows one to compute directly behavior strategies rather than mixed strategies.

3.8 Subgame perfect equilibria

A *subgame* of an extensive game is a subtree of the game tree that includes all information sets containing a node of the subtree. Fig. 3.3 has a subgame starting at the decision node of player 2; the nodes in the second information set of player 1 are not roots of subgames because player 1 does not know that he is in the respective subtree. In the subgame, player 2 moves first, but player 1 does not get to know that move. So this subgame is equivalent to a 2×2 game in strategic form where the players act simultaneously. (In this way, every game in strategic form can be represented as a game in extensive form.)

The subgame in Fig. 3.3 has a unique mixed equilibrium with probability $2/3$ for the moves T and r, respectively, and expected payoff 4 to player 1 and $8/3$ to player 2. Replacing the subgame by the payoff pair $(4, 8/3)$, one obtains a very simple game with moves L and R for player 1, where L is optimal. So player 1's mixed strategy with probabilities $1/3$ and $2/3$ for $\langle L, S\rangle$ and $\langle L, T\rangle$ and player 2's mixed strategy $(1/3, 2/3)$ for l, r define a Nash equilibrium of the game. This is the, here unique, *subgame perfect* equilibrium of the game, defined by the property that it induces a Nash equilibrium in every subgame.

Algorithm 3.9 (Subgame perfect equilibrium) *Input:* An extensive game. *Output:* A subgame perfect Nash equilibrium of the game. *Method:* Consider, in increasing order of inclusion, each subgame of the game, find a Nash equilibrium of the subgame, and replace the subgame by a new terminal node that has the equilibrium payoffs.

In a game with perfect information, every node is the root of a subgame. Then Algorithm 3.9 is the well known, linear time *backward induction* method, also sometimes known as "Zermelo's algorithm". Because the subgame involves only one player in each iteration, a deterministic move is optimal, which shows that any game with perfect information has a (subgame perfect) Nash equilibrium where every player uses a pure strategy.

In games with imperfect information, a subgame perfect equilibrium may require mixed strategies, as Fig. 3.3 demonstrates.

3.9 Reduced strategic form

Not all extensive games have nontrivial subgames, and one may also be interested in equilibria that are not subgame perfect. In Fig. 3.3, such an equilibrium is the pure strategy pair $(\langle R, S\rangle, l)$. Here, player 2 is *indifferent* between her moves l and r because the initial move R of player 1 means that player 2 never has to make move l or r, so player 2 receives the constant payoff 3 after move R. If play actually reached player 2's information set, move l would not be optimal against S, which is why this is not a subgame perfect equilibrium. Player 2 can in fact randomize between l and r, and as long as l is played with probability at least $2/3$, $\langle R, S\rangle$ remains a best response of player 1, as required in equilibrium.

In this game, the pure strategies $\langle R, S\rangle$ and $\langle R, T\rangle$ of player 1 are overspecific as "plans of action": the initial move R of player 1 makes the subsequent choice of S or T *irrelevant* since player 1's second information set cannot be reached after move R. Consequently, the two payoff rows for $\langle R, S\rangle$ and $\langle R, T\rangle$ are identical for both players. In the *reduced strategic form*, moves at information sets that cannot be reached due to an earlier own move are identified. In Fig. 3.3, this reduction yields the pure strategy (more precisely, equivalence class of pure strategies) $\langle R, *\rangle$, where $*$ denotes an arbitrary move. The two (reduced as well as unreduced) pure strategies of player 2 are her moves l and r.

The reduced strategic form of Fig. 3.3 corresponds to the bimatrix game (3.18) if $\langle R, *\rangle$ is taken as the first strategy (top row) of player 1. This game is *degenerate* even if the payoffs in the extensive game are generic, because player 2, irrespective of her own move, receives constant payoff 3 when player 1 chooses $\langle R, *\rangle$.

Once a two-player extensive game has been converted to its reduced strategic form, it can be considered as a bimatrix game, where we refer to its rows and columns as the "pure strategies" of player 1 and 2, even if they leave moves at unreachable information sets unspecified.

The concept of subgame perfect equilibrium requires fully specified strategies, rather than reduced strategies. For example, it is not possible to say whether the Nash equilibrium $(\langle R, *\rangle, l)$ of the reduced strategic form of the game in Fig. 3.3 is subgame perfect or not, because player 1's behavior at his second information set is unspecified. This could be said for a Nash equilibrium of the full strategic form with two rows $\langle R, S\rangle$ and $\langle R, T\rangle$.

However, these identical two rows are indistinguishable computationally, so there is no point in applying an algorithm to the full rather than the reduced strategic form, because any splitting of probabilities between payoff-identical strategies would be arbitrary. If one is interested in finding subgame perfect equilibria, one should use Algorithm 3.9. At each stage of that algorithm, the considered games have by definition no further subgames, and equilibria of these games can be found using the reduced strategic form or the sequence form.

A player may have *parallel* information sets that are not distinguished by own earlier moves. These arise when a player receives information about an earlier move by another player. Combinations of moves at parallel information sets cannot be reduced, which causes a multiplicative growth of the number of reduced strategies. In general, the reduced strategic form can therefore still be exponential in the size of the game tree.

3.10 The sequence form

In the reduced strategic form, pure strategies are only partially specified, by omitting moves at information sets that cannot be reached due to an own earlier move. In the *sequence form*, pure strategies are replaced by an even more partial description of *sequences* which specify a player's moves only along a *path* in the game tree. The number of these paths, and therefore of these sequences, is bounded by the number of nodes of the tree. However, randomizing between such sequences can no longer be described by a single probability distribution, but requires a system of linear equations.

A *sequence* of moves of player i is the sequence of his moves (disregarding the moves of other players) on the unique path from the root to some node t of the tree, and is denoted $\sigma_i(t)$. For example, for the leftmost leaf t in Fig. 3.3 this sequence is LS for player 1 and l for player 2. The empty sequence is denoted $\emptyset$. Player i has *perfect recall* if and only if $\sigma_i(s) = \sigma_i(t)$ for any nodes $s, t \in h$ and $h \in H_i$. Then the unique sequence $\sigma_i(t)$ leading to any node t in h will be denoted σ_h. Perfect recall means that the player cannot get additional information about his position in an information set by remembering his earlier moves. We assume all players have perfect recall.

Let β_i be a behavior strategy of player i. The move probabilities $\beta_i(c)$ fulfill

$$\sum_{c \in C_h} \beta_i(c) = 1, \qquad \beta_i(c) \geq 0 \quad \text{for } h \in H_i\,, \quad c \in C_h\,. \tag{3.20}$$

The *realization probability* of a sequence σ of player i under β_i is

$$\beta_i[\sigma] = \prod_{c \text{ in } \sigma} \beta_i(c). \tag{3.21}$$

An information set h in H_i is called *relevant* under β_i if $\beta_i[\sigma_h] > 0$, otherwise *irrelevant*, in agreement with irrelevant information sets as considered in the reduced strategic form.

Let S_i be the set of sequences of moves for player i. Then any σ in S_i is either the empty sequence $\emptyset$ or uniquely given by its last move c at the information set h in H_i, that is, $\sigma = \sigma_h c$. Hence,

$$S_i = \{\, \emptyset \,\} \;\cup\; \{\, \sigma_h c \mid h \in H_i,\; c \in C_h \,\}.$$

This implies that the number of sequences of player i, apart from the empty sequence, is equal to his total number of moves, that is, $|S_i| = 1 + \sum_{h \in H_i} |C_h|$. This number is linear in the size of the game tree.

Let β_1 and β_2 denote behavior strategies of the two players, and let β_0 be the known behavior of the chance player. Let $a(t)$ and $b(t)$ denote the payoffs to player 1 and player 2, respectively, at a leaf t of the tree. The probability of reaching t is the product of move probabilities on the path to t. The expected payoff to player 1 is therefore

$$\sum_{\text{leaves } t} a(t)\, \beta_0[\sigma_0(t)]\, \beta_1[\sigma_1(t)]\, \beta_2[\sigma_2(t)]\,, \tag{3.22}$$

and the expected payoff to player 2 is the same expression with $b(t)$ instead of $a(t)$. However, the expected payoff is nonlinear in terms of behavior strategy probabilities $\beta_i(c)$ since the terms $\beta_i[\sigma_i(t)]$ are products by (3.21).

Therefore, we consider directly the realization probabilities $\beta_i[\sigma]$ as functions of *sequences* σ in S_i. They can also be defined for mixed strategies μ_i of player i, which choose each pure strategy π_i of player i with probability $\mu_i(\pi_i)$. Under π_i, the realization probability of σ in S_i is $\pi_i[\sigma]$, which is equal to one if π_i prescribes all the moves in σ and zero otherwise. Under μ_i, the realization probability of σ is

$$\mu_i[\sigma] = \sum_{\pi_i} \mu_i(\pi_i)\pi_i[\sigma]. \tag{3.23}$$

For player 1, this defines a map x from S_1 to $\mathbb{R}$ by $x(\sigma) = \mu_1[\sigma]$ for $\sigma \in S_1$. We call x the *realization plan* of μ_1 or a realization plan for player 1. A realization plan for player 2, similarly defined on S_2 by a mixed strategy μ_2, is denoted y. Realization plans have two important properties:

Proposition 3.10 *A realization plan x of a mixed strategy of player 1 fulfills $x(\sigma) \geq 0$ for all $\sigma \in S_1$ and*

$$x(\emptyset) = 1, \qquad \sum_{c \in C_h} x(\sigma_h c) = x(\sigma_h) \quad \text{for all } h \in H_1. \tag{3.24}$$

Conversely, any $x\colon S_1 \to \mathbb{R}$ with these properties is the realization plan of a behavior strategy of player 1, which is unique except at irrelevant information sets. A realization plan y of player 2 is characterized analogously.

For the second property, two mixed strategies are called *realization equivalent* if they reach any node of the tree with the same probabilities, given any strategy of the other player. We can assume that all chance probabilities $\beta_0(c)$ are positive, by pruning any tree branches that are unreached by chance.

Proposition 3.11 *Two mixed strategies μ_i and μ_i' of player i are realization equivalent if and only if they have the same realization plan, that is, $\mu_i[\sigma] = \mu_i'[\sigma]$ for all $\sigma \in S_i$.*

These two propositions (to be proved in Exercise 3.13) imply the well-known result by Kuhn [Ku53] that behavior strategies are strategically as expressive as mixed strategies.

Corollary 3.12 (Kuhn's theorem) *For a player with perfect recall, any mixed strategy is realization equivalent to a behavior strategy.*

Prop. 3.10 characterizes realization plans by nonnegativity and the equations (3.11). A realization plan describes a behavior strategy uniquely except for the moves at irrelevant information sets. In particular, the realization plan of a *pure* strategy (that is, a realization plan with values zero or one) is as specific as a reduced pure strategy.

A realization plan represents all the relevant strategic information of a mixed strategy by Prop. 3.11. This compact information is obtained with the linear map in (3.23). This map assigns to any mixed strategy μ_i, regarded as a tuple of mixed strategy probabilities $\mu_i(\pi_i)$, its realization plan, regarded as a tuple of realization probabilities $\mu_i[\sigma]$ for σ in S_i. The simplex of mixed strategies is thereby mapped to the polytope of realization plans defined by the linear constraints in Prop. 3.10. The vertices of this polytope are the realization plans of pure strategies. The number of these vertices may be exponential. However, the number of defining inequalities and the dimension of the polytope is linear in the tree size. For player i, this dimension is the

number $|S_i|$ of variables minus the number $1+|H_i|$ of equations (3.24) (which are linearly independent), so it is $\sum_{h\in H_i}(|C_h|-1)$.

We consider realization plans as vectors in $x \in \mathbb{R}^{|S_1|}$ and $y \in \mathbb{R}^{|S_2|}$, that is, $x = (x_\sigma)_{\sigma\in S_1}$ where $x_\sigma = x(\sigma)$, and similarly $y = (y_\tau)_{\tau\in S_2}$. The linear constraints in Prop. 3.10 are denoted by

$$Ex = e, \quad x \geq \mathbf{0} \qquad \text{and} \qquad Fy = f, \quad y \geq \mathbf{0}, \tag{3.25}$$

using the *constraint* matrices E and F and vectors e and f. The matrix E and right hand side e have $1+|H_1|$ rows, and E has $|S_1|$ columns. The first row denotes the equation $x(\emptyset) = 1$ in (3.24). The other rows for $h \in H_1$ are the equations $-x(\sigma_h) + \sum_{c\in C_h} x(\sigma_h c) = 0$.

In Fig. 3.3, the sets of sequences are $S_1 = \{\emptyset, L, R, LS, LT\}$ and $S_2 = \{\emptyset, l, r\}$, and in (3.25),

$$E = \begin{bmatrix} 1 & & & & \\ -1 & 1 & 1 & & \\ & -1 & & 1 & 1 \end{bmatrix}, \quad e = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad F = \begin{bmatrix} 1 & & \\ -1 & 1 & 1 \end{bmatrix}, \quad f = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Each sequence appears exactly once on the left hand side of the equations (3.24), accounting for the entry 1 in each column of E and F. The number of information sets and therefore the number of rows of E and F is at most linear in the size of the game tree.

Define the *sequence form payoff* matrices A and B, each of dimension $|S_1| \times |S_2|$, as follows. For $\sigma \in S_1$ and $\tau \in S_2$, let the matrix entry $a_{\sigma\tau}$ of A be defined by

$$a_{\sigma\tau} = \sum_{\text{leaves } t \,:\, \sigma_1(t)=\sigma,\ \sigma_2(t)=\tau} a(t)\,\beta_0[\sigma_0(t)]\,. \tag{3.26}$$

The matrix entry of B is this term with b instead of a. An example is shown on the bottom right in Fig. 3.3. These two matrices are *sparse*, since the matrix entry for a pair σ, τ of sequences is zero (the empty sum) whenever these sequences do not lead to a leaf. If they do, the matrix entry is the payoff at the leaf (or leaves, weighted with chance probabilities of reaching the leaves, if there are chance moves). Then by (3.22), the expected payoffs to player 1 and player 2 are $x^\top Ay$ and $x^\top By$, respectively, which is just another way of writing the weighted sum over all leaves. The constraint and payoff matrices define the *sequence form* of the game.

3.11 Computing equilibria with the sequence form

Realization plans in the sequence form take the role of mixed strategies in the strategic form. In fact, mixed strategies x and y are a special case, by

letting E and F in (3.25) be single rows $\mathbf{1}^\top$ and $e = f = 1$. The computation of equilibria with the sequence form uses linear programming duality, which is also of interest for the strategic form.

Consider a fixed realization plan y of player 2. A best response x of player 1 is a realization plan that maximizes his expected payoff $x^\top(Ay)$. That is, x is a solution to the linear program (LP)

$$\text{maximize } x^\top(Ay) \text{ subject to } Ex = e, \quad x \geq \mathbf{0}. \tag{3.27}$$

This LP has a *dual* LP with a vector u of unconstrained variables whose dimension is $1 + |H_1|$, the number of rows of E. This dual LP states

$$\text{minimize } e^\top u \text{ subject to } E^\top u \geq Ay. \tag{3.28}$$

Both LPs have feasible solutions, so by the strong duality theorem of linear programming, they have the same optimal value.

Consider now a *zero-sum game*, where $B = -A$. Player 2, when choosing y, has to assume that her opponent plays rationally and maximizes $x^\top Ay$. This maximum payoff to player 1 is the optimal value of the LP (3.27), which is equal to the optimal value $e^\top u$ of the dual LP (3.28). Player 2 is interested in minimizing $e^\top u$ by her choice of y. The constraints of (3.28) are linear in u and y even if y is treated as a variable. So a *minmax* realization plan y of player 2 (minimizing the maximum amount she has to pay) is a solution to the LP

$$\underset{u,\,y}{\text{minimize}} \; e^\top u \text{ subject to } Fy = f, \quad E^\top u - Ay \geq \mathbf{0}, \quad y \geq \mathbf{0}. \tag{3.29}$$

The dual of this LP has variables v and x corresponding to the primal constraints $Fy = f$ and $E^\top u - Ay \geq \mathbf{0}$, respectively. It has the form

$$\underset{v,\,x}{\text{maximize}} \; f^\top v \text{ subject to } Ex = e, \quad F^\top v - A^\top x \leq \mathbf{0}, \quad x \geq \mathbf{0}. \tag{3.30}$$

It is easy to verify that this LP describes the problem of finding a *maxmin* realization plan x (with maxmin payoff $f^\top v$) for player 1.

This implies, first, that any zero-sum game has an equilibrium (x, y). More importantly, given an extensive game, the number of nonzero entries in the sparse matrices E, F, A, and the number of variables, is linear in the size of the game tree. Hence, we have shown the following.

Theorem 3.13 *The equilibria of a two-person zero-sum game in extensive form with perfect recall are the solutions to the LP* (3.29) *with sparse payoff matrix A in* (3.26) *and constraint matrices E and F in* (3.25) *defined by Prop.* 3.10. *The size of this LP is linear in the size of the game tree.*

A best response x of player 1 against the mixed strategy y of player 2 is a solution to the LP (3.27). This is also useful for games that are not zero-sum. By strong duality, a feasible solution x is optimal if and only if there is a dual solution u fulfilling $E^\top u \geq Ay$ and $x^\top(Ay) = e^\top u$, that is, $x^\top(Ay) = (x^\top E^\top)u$ or equivalently

$$x^\top(E^\top u - Ay) = 0\,. \tag{3.31}$$

Because the vectors x and $E^\top u - Ay$ are nonnegative, (3.31) states that they are *complementary* in the sense that they cannot both have positive components in the same position. This characterization of an optimal primal-dual pair of feasible solutions is known as *complementary slackness* in linear programming. For the strategic form, this condition is equivalent to the best response condition (3.2).

For player 2, the realization plan y is a best response to x if and only if it maximizes $(x^\top B)y$ subject to $Fy = f$, $y \geq \mathbf{0}$. The dual of this LP has the vector v of variables and says: minimize $f^\top v$ subject to $F^\top v \geq B^\top x$. Here, a primal-dual pair y, v of feasible solutions is optimal if and only if, analogous to (3.31),

$$y^\top(F^\top v - B^\top x) = 0\,. \tag{3.32}$$

Considering these conditions for both players, this shows the following.

Theorem 3.14 *Consider the two-person extensive game with sequence form payoff matrices A, B and constraint matrices E, F. Then the pair (x, y) of realization plans defines an equilibrium if and only if there are vectors u, v so that*

$$\begin{array}{rlrl} Ex = e, & x \geq \mathbf{0}, & Fy = f, & y \geq \mathbf{0}, \\ E^\top u - Ay \geq \mathbf{0}, & & F^\top v - B^\top x \geq \mathbf{0} & \end{array} \tag{3.33}$$

and (3.31), (3.32) *hold. The size of the matrices E, F, A, B is linear in the size of the game tree.*

The conditions (3.33) define a *linear complementarity problem* (LCP). For a game in strategic from, (3.8), (3.9), and (3.10) define also an LCP, to which the LH algorithm finds one solution. For a general extensive game, the LH algorithm cannot be applied to the LCP in Theorem 3.14, because u and v are not scalar dual variables that are easily eliminated from the system. Instead, it is possible to use a variant called *Lemke's algorithm*. Similar to the LH algorithm, it introduces a degree of freedom to the system, by considering an additional column for the linear equations and a corresponding variable z_0 which is initially nonzero, and which allows for an

initial feasible solution where $x = \mathbf{0}$ and $y = \mathbf{0}$. Then a binding inequality in $r = E^\top u - Ay \geq \mathbf{0}$ (or $s = F^\top v - B^\top x \geq \mathbf{0}$) means that a basic slack variable r_σ (or s_τ) can leave the basis, with x_σ (respectively, y_τ) entering, while keeping (3.10). Like in the LH algorithm, this "complementary pivoting rule" continues until an equilibrium is found, here when the auxiliary variable z_0 leaves the basis.

3.12 Further reading

A scholarly and more comprehensive account of the results of this chapter is [vS02]. The best response condition (Prop. 3.1) is due to Nash [Na]. Algorithm 3.4 is folklore, and has been used by [DK]. Polyhedra are explained in [Zi]. Shapley [Sh] introduced distinct labels as in (3.1) to visualize the LH algorithm. He labels subdivisions of the mixed strategy simplices, ignoring the payoff components in $\overline{P}$ and $\overline{Q}$ in (3.4). We prefer the polytope view using P and Q in (3.6), which simplifies the LH algorithm. Moreover, this view is useful for constructing games with many equilibria [vS99] that come close to the upper bound theorem for polytopes [Mc][Ke], and for games with exponentially long LH paths [SvS].

Algorithm 3.5 is suggested in [Vo][Ku61][Ma]. The *lrs* method for vertex enumeration is due to [AF][Av]. An equilibrium enumeration that (implicitly) alternates between $\overline{P}$ and $\overline{Q}$ is [AHJS]. It has been implemented with integer pivoting (like *lrs*) by [Ros].

The LH algorithm is due to [LH]. Shapley [Sh] also shows that the endpoints of an LH path are equilibria of different *index*, which is an orientation defined by determinants, explored further in [vSch]. A recent account of integer pivoting is [AP]. Prop. 3.8 is due to [Wi] and [Ja].

Extensive games with information sets are due to [Ku53]. Subgame perfection [Se75] is one of many *refinements* of Nash equilibria [vD]. Main ideas of the sequence form have been discovered independently by [Rom][KM][vS96]. Lemke's algorithm [Le] is applied to the sequence form in [KMvS][vSET].

A recent paper, with further references, on algorithms for finding equilibria of games with more than two players is [Da].

3.13 Discussion and open problems

We have described the basic mathematical structure of Nash equilibria for two-player games, namely polyhedra and the complementarity condition of best responses. The resulting algorithms should simplify the analysis of larger games as used by applied game theorists. At present, existing software

packages [Av][Ca][MMT] are prototypes that are not easy to use. Improved implementations should lead to more widespread use of the algorithms, and reveal which kinds of games practitioners are interested in. If the games are discretized versions of games in economic settings, enumerating all equilibria will soon hit the size barriers of these exponential algorithms. Then the LH algorithm may possibly be used to give an indication if the game has only one Nash equilibrium, or Lemke's method with varying starting point as in [vSET]. This should give practical evidence if these algorithms have usually good running times, as is widely believed, in contrast to the extremal examples in [SvS]. An open theoretical question is if LH, or Lemke's algorithm, has expected polynomial running time, as it is known for the simplex method, for suitable probabilistic assumptions on the instance data.

The computational complexity of finding one Nash equilibrium of a two-player game, as discussed in Chapter 2, is open in the sense that not even a sub-exponential algorithm is known. Incremental or divide-and-conquer approaches, perhaps using the polyhedral structure, require a generalization of the equilibrium condition, because equilibria typically do not result from equilibria of games with fewer strategies. At the same time, such an approach must not maintain the entire set of Nash equilibria, because questions about that set (such as uniqueness, see Theorem 2.1) are typically NP-hard.

Extensive games are a general model of dynamic games. The condition of perfect recall leads to canonical representations and algorithms, as described. Special types of extensive games, like repeated games and Bayesian games, are widely used in applied game theory. Finding equilibria of these models – where that task is difficult – should give a focus for further research.

Bibliography

[AHJS] C. Audet, P. Hansen, B. Jaumard, and G. Savard (2001), Enumeration of all extreme equilibria of bimatrix games. *SIAM J. Sci. Computing* 23, 323–338.

[AF] D. Avis and K. Fukuda (1992), A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Disc. Comp. Geometry* 8, 295–313.

[Av] D. Avis (2005), User's Guide for lrs. http://cgm.cs.mcgill.ca/∼avis

[AP] D.-O. Azulay and J.-F. Pique (2001), A revised simplex method with integer Q-matrices. *ACM Trans. Math. Software* 27, 350–360.

[BK] C. Bron and J. Kerbosch (1973), Finding all cliques of an undirectred graph. *Comm. ACM* 16, 575–577.

[Ca] M. J. Canty (2003), *Resolving Conflict with Mathematica: Algorithms for Two-Person Games*. Academic Press, Amsterdam.

[Da] R. S. Datta (2003), Using computer algebra to compute Nash equilibria. *Proc. 2003 Int. Symp. Symbolic and Algebraic Computation*, ACM, 74–79.

[DK] J. Dickhaut and T. Kaplan (1991), A program for finding Nash equilibria. *The Mathematica Journal* 1:4, 87–93.

[Ja] M. J. M. Jansen (1981), Maximal Nash subsets for bimatrix games. *Naval Research Logistics Quarterly* 28, 147–152.

[Ke] H. Keiding (1997), On the maximal number of Nash equilibria in an $n \times n$ bimatrix game. *Games Econ. Behav.* 21, 148–160.

[KM] D. Koller and N. Megiddo (1992), The complexity of two-person zero-sum games in extensive form. *Games Econ. Behav.* 4, 528–552.

[KMvS] D. Koller, N. Megiddo, and B. von Stengel (1996), Efficient computation of equilibria for extensive two-person games. *Games Econ. Behav.* 14, 247–259.

[Ku53] H. W. Kuhn (1953), Extensive games and the problem of information. In: *Contributions to the Theory of Games II*, eds. H. W. Kuhn and A. W. Tucker, Ann. Math. Studies 28, Princeton Univ. Press, Princeton, 193–216.

[Ku61] H. W. Kuhn (1961), An algorithm for equilibrium points in bimatrix games. *Proc. National Academy of Sciences of the U.S.A.* 47, 1657–1662.

[Le] C. E. Lemke (1965), Bimatrix equilibrium points and mathematical programming. *Management Science* 11, 681–689.

[LH] C. E. Lemke and J. T. Howson, Jr. (1964), Equilibrium points of bimatrix games. *J. SIAM* 12, 413–423.

[Ma] O. L. Mangasarian (1964), Equilibrium points in bimatrix games. *J. SIAM* 12, 778–780.

[MMT] R. D. McKelvey, A. McLennan, and T. L. Turocy (2006), Gambit: Software Tools for Game Theory. http://econweb.tamu.edu/gambit

[Mc] P. McMullen (1970), The maximum number of faces of a convex polytope. *Mathematika* 17, 179–184.

[Na] J. F. Nash (1951), Non-cooperative games. *Ann. Math.* 54, 286–295.

[Rom] I. V. Romanovskii (1962), Reduction of a game with complete memory to a matrix game. *Soviet Mathematics* 3, 678–681.

[Ros] G. D. Rosenberg (2004), Enumeration of all extreme equilibria of bimatrix games with integer pivoting and improved degeneracy check. CDAM Res. Rep. LSE-CDAM-2005-18, London School of Economics.

[SvS] R. Savani and B. von Stengel (2006), Hard-to-solve bimatrix games. *Econometrica* 74, 397–429.

[Se75] R. Selten (1975), Reexamination of the perfectness concept for equilibrium points in extensive games. *Int. J. Game Theory* 4, 22–55.

[Sh] L. S. Shapley (1974), A note on the Lemke–Howson algorithm. *Mathematical Programming Study 1: Pivoting and Extensions*, 175–189.

[vD] E. van Damme (1987), *Stability and Perfection of Nash Equilibria.* Springer, Berlin.

[vSch] A. von Schemde (2005), *Index and Stability in Bimatrix Games.* Springer, Berlin.

[vS96] B. von Stengel (1996), Efficient computation of behavior strategies. *Games Econ. Behav.* 14, 220–246.

[vS99] B. von Stengel (1999), New maximal numbers of equilibria in bimatrix games. *Disc. Comp. Geometry* 21, 557–568.

[vS02] B. von Stengel (2002), Computing equilibria for two-person games. In: *Handbook of Game Theory with Economic Applications*, Vol. 3, eds. R. J. Aumann and S. Hart, Elsevier, Amsterdam, 1723–1759.

[vSET] B. von Stengel, A. H. van den Elzen, and A. J. J. Talman (2002), Computing normal form perfect equilibria for extensive two-person games. *Econometrica*

70, 693–715.
[Vo] N. N. Vorob'ev (1958), Equilibrium points in bimatrix games. *Theory of Probability and its Applications* 3, 297–309.
[Wi] H.-M. Winkels (1979), An algorithm to determine all equilibrium points of a bimatrix game. In: *Game Theory and Related Topics*, eds. O. Moeschlin and D. Pallaschke, North-Holland, Amsterdam, 137–148.
[Zi] G. M. Ziegler (1995), *Lectures on Polytopes*. Springer, New York.

Exercises

3.1 Prove the claim made after Algorithm 3.4 that nonunique solutions to the equations in that algorithm occur only for degenerate games.

3.2 Show that in an equilibrium of a nondegenerate game, all pure best responses are played with positive probability.

3.3 Give further details of the argument made after Algorithm 3.6 that LH terminates. A duplicate label of a vertex pair (x, y) can be dropped in either polytope. Interpret these two possibilities.

3.4 Why is every pure strategy equilibrium found by LH for a suitable missing label?

3.5 Show that the "projection" to polytope P, say, of a LH path from (x, y) to (x', y') in $P \times Q$ is also a path in P from x to x'. Hence, if (x, y) is an equilibrium, where can x be on that projected path?

3.6 Verify the LH paths for the example (3.7).

3.7 Apply integer pivoting to the system $r + Ay = \mathbf{1}$ in the example, omitted after (3.13).

3.8 After (3.14), what is the multiplier in the "suitable multiple of the pivot row"? Give formulas for the update rules of the tableau.

3.9 Draw the polytope P for the game (3.18), and verify that the described naive use of LH fails.

3.10 Implement the lexico-minimum ratio test for the system (3.19) using the data in (3.17); you need a suitable array to identify the order of the basic variables.

3.11 Adapt a clique enumeration algorithm for graphs such as [BK] to find all maximal Nash subsets (see end of Section 3.6).

3.12 Consider an extensive game with a binary game tree of depth L (and thus 2^L leaves), where the two players alternate and are informed about all past moves except for the last move of the other player (see [vSET]). How many reduced strategies do the players have?

3.13 Prove Prop. 3.10, using (3.20), (3.21), and (3.23). Prove Prop. 3.11.

3.14 Write down the LCP of Theorem 3.14 for the game in Fig. 3.3. Find all its solutions, for example with a variant of Algorithm 3.4.