On the Complexity of Two-Player Win-Lose Games

Tim Abbott
MIT
tabbott@mit.edu

Daniel Kane
MIT
dankane@mit.edu

Paul Valiant*
MIT
Computer Science and Artificial Intelligence Laboratory
The Stata Center, Building 32
32 Vassar Street, Cambridge, MA 02139
pvaliant@mit.edu

Abstract

The efficient computation of Nash equilibria is one of the most formidable challenges in computational complexity today. The problem remains open for two-player games.

We show that the complexity of two-player Nash equilibria is unchanged when all outcomes are restricted to be 0 or 1. That is, win-or-lose games are as complex as the general case for two-player games.

1 Game Theory

Game theory asks the question: given a set of players playing a certain game, what happens? Computational game theory asks the question: given a representation of a game and some fixed criteria for reasonable play, how may we efficiently compute properties of the possible outcomes?

Needless to say, there are many possible ways to define a game, and many more ways to efficiently represent these games. Since the computational complexity of an algorithm is defined as a function of the length of its input representation, different game representations may have significantly different algorithmic consequences. Much work is being done to investigate how to take advantage of some of the more exotic representations of games (see [4, 7, 8, 10] and the references therein). Nevertheless, for two player games, computational game theorists almost exclusively work with the representation known as a *rational bimatrix game*, which we define as follows.

Definition 1 *A rational bimatrix game is a game representation that consists of a matrix of pairs of rational numbers represented in binary (or equivalently a pair of identically sized rational matrices). The game has two players, known as the* row *and* column *players respectively. The matrix is interpreted to represent the following interaction: the row and column players simultaneously pick a row and column respectively of the matrix; these choices specify an entry—a pair—at the intersection of this row and column, and the row and column players receive* payoffs *proportional respectively, to the first and second components of the pair.*

In this model, a *strategy* for the row or column player consists of a probability distribution on the rows or columns respectively, and is represented as a vector r or c.

To motivate the definition of a *Nash equilibrium*, we define the notion of a *best response*. Given a strategy r for the row player, we may ask which strategies c give the column player his maximal payoff. Such a strategy c is said to be a *best response* to the strategy r. Game theorists model "reasonable play" in a bimatrix game with the following criterion:

Definition 2 *A pair of strategies* (r, c) *is said to be a* Nash equilibrium *if* r *is a best response to* c *and* c *is simultaneously a best response to* r.

2 Complexity of Nash equilibria

A fundamental property of Nash equilibria is that *they always exist*. It is far from obvious that this should be the case—equilibria for constant-sum two-player games were first shown to exist by von Neumann. This result was later generalized by Nash to general multi-player games using the Kakutani fixed point theorem.

A purely combinatorial existence proof for Nash equilibria in two-player games was found by Lemke and Howson

*Supported by an NDSEG Fellowship.

that has the additional advantage of being *constructive* [5]. Unfortunately, the Lemke-Howson algorithm has exponential worst-case running time [9].

An alternate algorithm for finding Nash equilibria for two-player games is suggested by the following observation: if we know the *support* of the strategies in a Nash equilibria, namely the set of rows and columns that are played with positive probability, we can reconstruct the set of Nash equilibria with that support by solving a linear program. This suggests the *support enumeration* algorithm, wherein we nondeterministically guess supports and check their feasibility. This algorithm has the important consequence of placing the Nash equilibrium search problem in the complexity class FNP, the search problem version of NP. This linear programming formulation also has the consequence that if the payoffs of the game are rational, then every support set that has a Nash equilibrium has a Nash equilibrium with *rational* weights.

THE DIFFICULTY OF THE NASH PROBLEM. It is natural to ask whether the problem of finding a Nash equilibrium is in fact in P, the class of problems with polynomial-time algorithms. Quite recently there have been significant results on the complexity of several related problems, which have been shown to be NP- or #P-hard [1, 3]. Specifically, counting the number of Nash equilibria is #P-hard, while determining if there exist Nash equilibria with certain properties—such as having specific payoffs or having specific strategies in their support—is NP-complete. However, the original problem of finding a single Nash equilibrium remains open and, as Christos Papidimitriou has famously stated, "Together with factoring, the complexity of finding a Nash equilibrium is in my opinion the most important concrete open question on the boundary of P today" [6].

SOURCE OF COMPLEXITY FOR THE NASH PROBLEM. There are many aspects of games that might make the Nash problem hard to solve. Specifically, considering multi-player games as multi-dimensional arrays of numbers, it is natural to ask which parameters of these arrays make finding Nash equilibria hard. Is it:

1. the number of dimensions of the array?
2. the number of options available to each player?
3. the complexity of the individual numbers involved?

The first question remains unresolved, as the Nash problem is wide open even for two-player games. We consider two-player games exclusively for the remainder of this paper.

The second question appears to have a positive answer: there exist fixed-parameter tractable algorithms with parameter the size of the strategy space available to one player of a two-player game. Thus games where one of the players has a restricted strategy space are easy, while general games appear much harder.

The third question—asking whether having complicated payoffs makes the Nash problem hard—is the subject of this paper. We answer this question in the negative. The first results of this kind were shown in [2]: determining whether there is more than one Nash equilibrium is NP-complete even in a $\{0,1\}$-game, and determining if there exists a Nash equilibrium with 0-payoff for one player is NP-complete for $\{0,1\}$-games. These results led them to raise the question of whether $\{0,1\}$-games are as hard as general games.

OUR CONTRIBUTION. We give a strong positive answer to the above question, exhibiting a specific mapping from rational-payoff bimatrix games into $\{0,1\}$-payoff bimatrix games that preserves the Nash equilibria in an efficiently recoverable form. We make this statement more precise in the next section by introducing the notion of a *Nash homomorphism*.

3 Nash homomorphisms

Our goal is to reduce the problem of finding a Nash equilibrium of a rational-payoff game to that of finding a Nash equilibrium of a $\{0,1\}$-payoff game. The notion of *reduction* suitable to our purposes is a kind of one-query Cook reduction which we call a *Nash homomorphism*. Specifically, we have the following:

Definition 3 *A* Nash homomorphism *is a map h, from a set of two-player games $\mathfrak{A}$ into a set of two-player games $\mathfrak{B}$, such that there exists a polynomial-time function f that when given a Nash equilibrium of a game in $\mathfrak{B}$ returns a Nash equilibrium of the game's pre-image under h.*

Intuitively, if a Nash homomorphism h maps a game A to a game B, then finding a Nash equilibrium of A reduces to finding a Nash equilibrium of B.

The main result of this paper is an explicit Nash homomorphism that takes rational-payoff games to $\{0,1\}$-payoff games. Specifically, we exhibit a *sequence* of Nash homomorphisms. Note that Nash homomorphisms compose: the forward mappings h compose, and the backward mappings f compose. The homomorphisms we construct will change the game incrementally into a $\{0,1\}$-game, while leaving a trail of backward mappings which relate each equilibrium of the final $\{0,1\}$-game to an equilibrium of the original game.

OUR GENERAL STRATEGY.

We construct a Nash homomorphism that translates a single column of the row player's payoffs from rational to binary, without significantly increasing the size of the game. Applying this homomorphism once on each column, and then a corresponding homomorphism to each row of the column player's payoffs yields our desired result.

This Nash homomorphism results from a combination of three main ideas, which we outline briefly before discussing in detail below.

Expressing entries in binary, for us, consists of the following three steps: first, find powers of two in $\{0,1\}$-games; second, find out how to take linear combinations of these powers of two; and third, find a way to restrict the structure of Nash equilibria of the resulting game so that we get no extraneous equilibria.

For the first task, that of simply finding powers of two in the Nash equilibria of a $\{0,1\}$-game, we note a well-known special case of the Nash equilibrium problem that reduces to solving a system of linear equations. For a $\{0,1\}$-game, the corresponding linear equations have $\{0,1\}$-coefficients. Through straightforward arithmetic, we show how to construct powers of two as solutions to these equations, and thereby as Nash equilibria of a $\{0,1\}$-game. We call the game thus constructed a *generator* game G, because it generates the powers of two that are fundamental to the rest of the binary translation process.

The second task—of taking suitable linear combinations of the powers of two we have just constructed—is accomplished by embedding the generator game G inside a larger matrix. Specifically, we take advantage of the following fact: the only thing that matters to either of the players of a game is his *expected* payoff. That is, a player is completely indifferent between getting a payoff of (say) $\frac{1}{3}$ up front, and getting a payoff of 1 exactly a third of the time. In the case of a two player game, the natural way to randomly choose a payoff for (say) the row player is to let the column player effect the randomization. Thus to simulate a payoff that is a sum of powers of two, we place the generator game G across certain columns of a larger game, fill in the row player's payoffs in these columns with appropriate $\{0,1\}$-values, and trust the row player to correctly "interpret" these entries as representing a single entry that is a linear combination of powers of 2.

For the third task, of binding the pieces of the game together to make sure that every Nash equilibrium of the modified game corresponds to one from the original game, we use a technique based on the notion of a *mimicking game*. A mimicking game (sometimes called an *imitation* game) is a game whose payoffs satisfy a simple set of inequalities that results in severely constraining the structure of the game's Nash equilibria. We use this structure to "program" a $\{0,1\}$-game to have an equilibrium of the right form, correctly integrating the above two techniques with the rest of the game.

4 Finding 2s in a $\{0,1\}$-game

In this section we provide the construction mentioned above of how to find powers of two in a $\{0,1\}$ game. Specifically, for any positive integer j, we construct a game G_j that has a *unique* Nash equilibrium wherein the actions of the row and column player are played with probabilities proportional to the first j powers of 2.

As mentioned above, the Nash equilibrium problem is equivalent to solving linear equations in certain cases. Here we consider the case of *full support* Nash equilibria.

Suppose we have a game (R,C), where R is the payoff matrix for the row player and C is the payoff matrix for the column player. Suppose further, that we have a *full support* Nash equilibrium (r,c) of this game, namely a Nash equilibrium where *every* row and column is played with nonzero weight.

Suppose the expected payoff to the row player in this equilibrium is p. Then for the row player to play each row with nonzero probability, the expected payoff of him playing in each row must equal p. These expected payoffs are called the *incentives* to play in each row. Formally, this becomes the constraint

$$Rc = p.$$

We note that since c represents a probability distribution, we have the additional constraints that

$$\sum c = 1 \quad \text{and} \quad c > 0.$$

Expressing c in homogenous coordinates, we may equivalently solve

$$Rc = 1,$$

and then check that $c > 0$. Our goal now is to find a $\{0,1\}$-matrix R such that the unique solution to $Rc = 1$ has elements of c proportional to powers of 2.

Define matrices A, B as

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}.$$

For $k = 3j$ define the $k \times k$ matrix S_j to have the following $j \times j$ block form:

$$S_j = \begin{pmatrix} A & A & \cdots & A & B \\ A & A & \cdots & B & 0 \\ \vdots & \vdots & \iddots & \vdots & \vdots \\ A & B & \cdots & 0 & 0 \\ B & 0 & \cdots & 0 & 0 \end{pmatrix}.$$

Explicitly, S_j has block B on the minor diagonal, block A above, and 0 below.

We claim that $R = S_j$ has the desired properties.

Claim 4 *The equation $S_j c = 1$ has a unique solution of*

$$c = \frac{1}{2^j}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T.$$

Proof: We prove this claim by induction. Suppose as our induction hypothesis that the first $3i$ entries of c equal the corresponding elements of this vector. As a base case, we note that the bottom three rows produce the equations

$$B(c_1, c_2, c_3)^T = 1,$$

which implies

$$c_1 = c_2 = c_3 = \frac{1}{2},$$

as desired.

To prove the induction, consider the $i + 1$st block row from the bottom of S_j. These three rows consist of i blocks of A followed by one block B, followed by zeros.

Consider the contribution to the sums of these rows provided by the i blocks of A. By the induction hypothesis, the first $3i$ components of c are

$$(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, ..., \frac{1}{2^i}, \frac{1}{2^i}, \frac{1}{2^i}),$$

which makes these i blocks of A have sum

$$1 - 2^i.$$

Since the total sum of each row is 1, and the only other nonzero coefficients in these rows are in the following B block, we have that

$$c_{3i+1} + c_{3i+2} = c_{3i+1} + c_{3i+3} = c_{3i+2} + c_{3i+3} = \frac{1}{2^i}.$$

This implies that

$$c_{3i+1} = c_{3i+2} = c_{3i+3} = \frac{1}{2^{i+1}},$$

which proves the desired induction.
▮

We note similarly the following claim, with proof analagous to the above:

Claim 5 *The equation $(1-S_j^T)r = 1$ has a unique solution of*

$$r = \frac{1}{2 \cdot 2^j - 3}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T.$$

We now have the following immediate corollary:

Corollary 6 *The game $(S_j, 1 - S_j)$ has exactly one* full support *Nash equilibrium (r, c), where*

$$r = c = \frac{1}{3(2^j-1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T.$$

Proof: Rescaling the solutions from the above two claims, we find that the unique solutions to

$$S_j c = p, \quad \sum c = 1, \quad c > 0$$

and

$$(1 - S_j^T)r = p', \quad \sum r = 1, \quad r > 0$$

are

$$r = c = \frac{1}{3(2^j-1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T.$$

Thus these are the only full support equilibria of the game $(S_j, 1 - S_j)$. ▮

It remains to be shown that the game $(S_j, 1 - S_j)$ has no other Nash equilibria, i.e. without full support. The key observation here is that, by construction, the game $(S_j, 1-S_j)$ is a *constant-sum* game, namely that the sum of the row and column payoffs in each entry is a constant—here 1. The significance of this fact is that the Nash equilibria of constant-sum games may be expressed as the solutions of a linear program, and thus this set is convex. This implies uniqueness by a simple topology argument.

Corollary 7 *The equilibrium*

$$r = c = \frac{1}{3(2^j-1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T$$

of the game $(S_j, 1-S_j)$ is the only *Nash equilibrium of this game.*

Proof: The set of full-support Nash equilibria of $(S_j, 1 - S_j)$ is the intersection of the set of Nash equilibria of $(S_j, 1 - S_j)$ and the set of vectors (r, c) with strictly positive elements. Namely, a single point is the intersection of a convex set with an open set. This implies that the convex set consists only of this point. Explicitly, the set of Nash equilibria of $(S_j, 1 - S_j)$ consists only of the full support equilibrium we have already found. ▮

We have thus exhibited a $\{0, 1\}$-game $G_j = (S_j, 1-S_j)$ which represents powers of 2 via its Nash equilibrium, our desired goal.

5 Subgames and linearity of expectation

In this next section we consider the ramifications of embedding a generator game inside a larger game. As a motivating example, consider the game "rock–paper–scissors", defined by the following payoff matrix:

0,0	0,1	1,0
1,0	0,0	0,1
0,1	1,0	0,0

It is easy to check that this game has a unique Nash equilibrium where each row and column is played with probability $\frac{1}{3}$. Suppose we embed the rock–paper–scissors game into a larger game. Specifically, suppose we take the rock–paper–scissors game and add a number of rows to it, filling in the new entries somehow. Suppose further that we know for a fact that, despite our modifications, the column player will still play his three strategies with probabilities $\frac{1}{3} : \frac{1}{3} : \frac{1}{3}$.

Consider this game from the row player's perspective: if he sees a row with two ones in it, he should instead view it as a row with the single entry $\frac{2}{3}$, since the column player's random actions make this row worth exactly $\frac{2}{3}$ to the row player. In this manner, we can express any payoff in $\{0, \frac{1}{3}, \frac{2}{3}, 1\}$ as a triple of $\{0, 1\}$-payoffs.

We note that if instead of using rock–paper–scissors as our subgame we use one of the generator games G_j, then instead of only being able to express payoffs that are sums of subsets of

$$\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$$

we could now represent any number expressible as a sum of a subset of

$$\frac{1}{3(2^j - 1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1),$$

which is our desired binary representation.

All this, however, rests on the supposition that the original Nash equilibrium of the generator game will be preserved despite the game being embedded in a larger game. To ameliorate this situation, we show how to set up an embedding so that this situation will arise from a more limited hypothesis. We have the following lemma.

Lemma 8 *Suppose a game G appears embedded in a larger game H. Specifically, if H has row set R and column set C, let the game G appears at the intersection of rows $r \subset R$ and columns $c \subset C$. Further, suppose the row player gets columnwise-uniform payoff at the intersection of rows r and columns $C \backslash c$, and the column player gets rowwise-uniform payoff at the intersection of columns c and rows $R \backslash r$. Then in any Nash equilibrium of H where some row of r and some column of c are played with positive probability, the restriction of this Nash equilibrium to rows r and columns c will be a scaled version of a Nash equilibrium of G.*

This lemma states that if we embed the subgame in the right way, then we are guaranteed that the columns of the subgame are played in the right ratio provided, only, that some row of G is played. (If none the columns of G are played then we are still fine since the 0 vector is proportional to anything.) In the next section we will see how to guarantee nonzero weights for these rows. We end this section with a proof of the lemma.

Proof: We first prove the lemma in the restricted case where the row player's payoffs at the intersection of rows r and columns $C \backslash c$ are 0 (instead of columnwise uniform) and the column player's payoffs at the intersection of columns c and rows $R \backslash r$ are 0:

$$H = \left(\begin{array}{c|c|c} ?,? & ?,0 & ?,? \\ \hline 0,? & \mathbf{G} & 0,? \\ \hline ?,? & ?,0 & ?,? \end{array} \right)$$

Let (x, y) be a Nash equilibrium of game H. Thus x is a best response to y, and y is a best response to x. Specifically, every row of x that is played with positive probability is a best response to y. Since the row player's payoffs in rows r are potentially nonzero only in columns c, we further note that every row of x in r that is played is a best response to the restriction of y to c. By symmetry, the complementary statement holds, that every column of y in c ever played is a best response to the restriction of x to r. Thus y restricted to c and x restricted to r are *mutual best responses*. Since these two restrictions have nonzero weight by hypothesis, we can scale them to have total weight 1, and have thus reconstructed the condition that these two restrictions are scaled versions of a Nash equilibrium of G, as desired.

For the general case, we note that it is straightforward to prove that adding a constant to any column of the row player's payoff, or to any row of the column player's payoff does not affect the Nash equilibria of a game. (In fact this is a simple example of a Nash homomorphism.) Thus given a game H with columnwise uniform payoffs for the row player at the intersection of rows r and columns $C \backslash c$, we could subtract off the appropriate constants for these columns, apply the special case of this lemma, and then add in the constants to derive the desired result. ∎

6 Binding with mimicking games

Suppose at this point we try as a gedanken experiment to come up with a construction.

We start with a game H that we wish to transform into binary, a column of row player payoffs at a time. Since games are scale invariant, we may as well start by clearing denominators of any fractional entries until all the payoffs are integer. We then create a generator game G_j, where j is at least as large as the number of bits in each integer. To transform a column of H, we rewrite this column as j columns, expressing each row player payoff in binary. Then we add enough extra rows to the game so that we may place the generator G_j at the head of these j columns to randomize appropriately. From Lemma 8, these binary entries will be interpreted correctly provided we can "bind" some row of G_j to be played with positive probability whenever its columns are. We must also decide what to do with the column player's entries in this column: copying each of these entries j times could greatly increase the number of non-$\{0, 1\}$ payoffs; leaving only one copy would violate the rowwise-uniform condition of Lemma 8, and moving these entries anywhere else would create a second set of entries that have to be properly "bound" back to this column. As it turns out, the right solution is in fact this third option, and we will solve both "binding" problems at once, via a construction we call a *mimicking game*.

Consider a 2×2 game where the row player's payoffs are a matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

satisfying

$$a > c \quad \text{and} \quad \mathrm{b} < \mathrm{d}.$$

Consider a Nash equilibrium of this game. Despite the fact that we know nothing about the column player's payoffs, and very little about the row player's payoffs, we can still reveal that this game has a significant "binding" structure. Specifically, suppose we have a Nash equilibrium of the game, and further suppose that in this Nash equilibrium, the row player sometimes plays the first row. Since $b < d$, the row player would not play the first row if the column player exclusively played the second column. Thus whenever the row player plays the first row with positive weight, we can conclude that the column player plays the first column with positive weight. Similarly, since $a > c$ we conclude that whenever the row player plays the second row with positive weight, the column player must play the second column with positive weight. Thus whatever strategies the first player plays, the second player must also play. As it turns out, this is exactly the form of binding we need.

To take this methodology one step further, suppose we give the second player a payoff matrix

$$\begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

where the payoffs are bound as

$$e < f \quad \text{and} \quad \mathrm{g} > \mathrm{h}.$$

Since these inequalities have the opposite direction from those in the first case, the binding is flipped: every time the column player plays column 1, the row player must play row 2 with nonzero weight, and every time the column player plays column 2, the row player must play row 1 with nonzero weight.

Putting the above two constructions together, we have a game where the row player sometimes playing row 1 implies the column player sometimes plays column 1, which implies the row player sometimes plays row 2, which implies the column player sometimes plays column 2, which implies the row player sometimes plays row 1 etc. In other words, all four strategies must have nonzero weight.

The import of this motivating example is that a few simple inequalities can bind various components of a game together so that they have a specific Nash equilibrium structure. Our final construction, of course, is not a 2×2 matrix; however, it uses the same inequalities. Specifically, we construct a 2×2 *block* matrix, with some additional columns not subject to these inequalities. Nevertheless, the robustness of these inequalities enables us to carry out our program.

7 The Construction

Using the above three components, we now exhibit the Nash homomorphism that translates a column of row-player payoffs to binary. As a preliminary, we compute the set of numbers expressible in terms of the Nash equilibrium of the generator game G_j. Recall that it is exactly these numbers that we can expect to express in binary.

Claim 9 *The set of numbers expressible as the product of a $\{0, 1\}$-vector with the Nash equilibrium strategy of the column player in the game G_j are just those number expressible as*

$$\frac{r}{3(2^j - 1)},$$

where r is an integer, $0 \leq r \leq 3(2^j - 1)$.

Proof: From Corollary 7, the column player's strategy in the unique Nash equilibrium of G_j is to play his $3j$ actions with probabilities

$$c = \frac{1}{3(2^j - 1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)$$

respectively. Thus the set of numbers expressible as the sum of a subset of these probabilities is exactly those numbers expressible as

$$\frac{r}{3(2^j - 1)},$$

where r is an integer, $0 \leq r \leq 3(2^j - 1)$, as desired. ▌

As we can see from the above claim, we must first translate all the row player's payoffs to be in this set before we can apply the binary translation. Towards this end, we note that the Nash equilibria of a game are completely unaffected when we apply any linear transform

$$f(x) = ax + b, \quad a > 0$$

to either player's payoffs. Thus, as a preconditioning step before we translate any columns, we transform all the row player's payoffs to lie in the set expressible by G_j. It turns out that we further need each of the transformed payoffs to be strictly greater than $\frac{2^j}{3(2^j-1)}$ in order for our construction to act as a "mimicking game" in the style of the previous section. We thus have the following preconditioning step:

Construction 10 (Preconditioning) *Given a matrix of (rational) row player payoffs, multiply the matrix by the common denominator of its entries so as to make the matrix integral. Next, find a j such that the difference between the largest and smallest integer is at most 2^j. Next, add or subtract a constant to all the entries so that the smallest entry becomes 2^j+1, and finally multiply all the entries by $\frac{1}{3(2^j-1)}$.*

We note that will be important to our construction for each entry of the column player's payoffs to be positive, and each row of this matrix to contain a strictly *positive entry; if this is not the case initially, we add a constant to each column player payoff to make it so here.*

Thus, without changing the Nash equilibria of the game, we have made all the entries strictly greater than $\frac{2^j}{3(2^j-1)}$, and expressible as $\frac{r}{3(2^j-1)}$, for r an integer $0 \leq r \leq 3(2^j-1)$.

We now introduce the Nash homomorphism that will translate a column of these row player payoffs to binary.

Construction 11 (Column translation) *Given an $m \times n$ game H, and a chosen column of H each of whose row player payoffs are expressible as $\frac{r}{3(2^j-1)}$ for r an integer, $2^j < r \leq 3(2^j-1)$, we exhibit a Nash homomorphism that transforms it into a new game H' such that all the row player's payoffs in this column become $\{0,1\}$, and the remaining non-$\{0,1\}$ entries of H are unchanged. Let H' be structured as a 2×3 block matrix as follows: let the first block of rows have size $3j$—enough to fit a copy of G_j, the generator for j-bit integers—and the second block have size m; let the first block of columns in H' have size 1, the second block have size $3j$, and the third block of columns have size $n-1$.*

For notational convenience, denote the matrix of row player payoffs of H by R, and the column player payoffs by C; let the ith column of H be the one we are translating; denote the ith column of R and C by R_i and C_i respectively; denote the remaining columns as R_{-i} and C_{-i} respectively. We fill in the blocks of H' as follows:

- *Block $(1,1)$, of size $3j \times 1$ receives a 1s vector for its row player payoffs, and a 0s vector for its column player payoffs.*
- *Block $(2,1)$, of size $m \times 1$ receives a 0s vector for its row player payoffs, and C_i for its column player payoffs.*
- *Block $(1,2)$, of size $3j \times 3j$, receives the game G_j.*
- *Block $(2,2)$, of size $m \times 3j$ receives for its row player payoffs the $\{0,1\}$ matrix obtained by taking the $m \times 1$ vector R_i and expressing each entry by the $1 \times 3j$ $\{0,1\}$-vector whose product with*
$$\frac{1}{3(2^j-1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1)^T$$
equals the original entry; the column player payoffs are 0.
- *Block $(3,1)$, of size $3j \times n-1$ receives 0s in both components.*
- *Block $(3,2)$, of size $m \times n-1$ receives the unaltered payoffs R_{-i} and C_{-i} as its row and column player payoffs respectively.* ▮

Pictorially, the matrix H' looks like:

$$\left(\begin{array}{c|c|c} 1,0 & G_j^1, G_j^2 & 0,0 \\ \hline 0, C_i & \mathrm{tr(R_i)}, 0 & R_{-i}, C_{-i} \end{array} \right).$$

We claim that the above construction is in fact a Nash homomorphism, i.e., that there exists an efficient way of recovering a Nash equilibrium of H given a Nash equilibrium of H'.

Theorem 12 *Construction 11 is a Nash homomorphism.*

We first describe the proposed map from Nash equilibria of H' to Nash equilibria of H, and then prove it actually maps equilibria to equilibria. The proof is a fairly straightforward application of the techniques that we have considered so far.

Construction 13 (Recovering an equilibrium) *Consider a Nash equilibrium (r', c') of the transformed game H'. Considering H' as a 2×3 block game as above, we may consider r' as having 2 blocks and c' as having 3 blocks. To transform (r', c') into a Nash equilibrium of H, apply the following steps:*

1. *Discard the first block from both r' and c'.*
2. *Replace the second block of c' with the sum of its $3j$ entries.*
3. *Reorder the n resulting entries from c' so that this sum appears in the ith place.*
4. *Scale the resulting vectors so that they each have sum 1, i.e. are proper probability distributions. Let this result be (r, c).* ▮

As a first step to showing that (r, c) is a Nash equilibrium of H, we examine the structure of (r', c'), the Nash equilibrium of H'. The main idea here is that the 2×2 block portion of H' obtained by ignoring the third column-block can be viewed as a mimicking game, that is, its payoffs can be viewed as satisfying the inequalities discussed in the previous section. Thus we can easily find a lot of structure in (r', c') that is analogous to the mimicking properties we found above.

Corresponding to the four binding properties of the 2×2 mimicking game, we present the following four claims.

Claim 14 *If a column from column-block 2 is played with positive probability then a row from row-block 1 must be played with positive probability.*

Proof: Suppose for the sake of contradiction that this were not the case. Thus since the intersection of the second column-block and the second row-block contains only 0 payoffs for the column player, the *incentive* for the column player to play in column-block 2 is 0. However, since no row from row-block 1 is played, some row from block 2 must be played; further, each row of block 2 contains a strictly positive payoff for the column player since each row of C contains a positive payoff. Thus some column would give the column player positive profit, which contradicts our assumption that the column player plays a column with 0 payoff. Thus the column player playing in block 2 with positive probability implies the row player plays in block 1 with positive probability. ▮

Corollary 15 *If any column from column-block 2 is played with positive probability then the probabilities of playing columns in column-block 2 are proportional to the weights of the Nash equilibrium of the generator game G_j.*

Proof: From the previous claim, we have that some row of row-block 1 is played with positive probability. Thus we can apply Lemma 8 to conclude that both the rows of row-block 1 and the columns of column-block 2 are played with weights proportional to the Nash equilibrium of G_j. ▮

This enables us to continue the mimicking argument.

Claim 16 *If the the actions in row-block 1 and column-block 2 are played with (strictly positive) probabilities proportional to the Nash equilibrium of G_j then column 1 must be played with positive probability.*

Proof: Assume for the sake of contradiction that this is not the case. Denote the total probability of the column player playing in column-block 2 by c'_2. Examining G_j we see that the incentive for the row player to play in any row in row-block 1 is $\frac{2^j}{3(2^j-1)} c'_2$. However, since each entry of A_i is strictly greater than $\frac{2^j}{3(2^j-1)}$, and each of these entries has been properly "translated" to $\{0, 1\}$ by construction, the incentives for the row player to play in the rows of row-block 2 are $A_i c'_2$, each of which is strictly greater than the incentive for the row player to play in the row he is playing—a contradiction. Thus a column from column-block 1 must be played with positive probability. ▮

Thus whenever a column from block 2 is played with positive probability, column 1 must also be played with positive probability.

Claim 17 *If column 1 is played with positive probability then some row from row-block 2 is played with positive probability.*

Proof: Assume otherwise, that only rows from row-block 1 are played. Then the column player receives 0 payoff in column 1. However, since each column of the column player's payoffs in G_j contains a 1, the column player could play a column from block 2 and receive positive payoff—a contradiction. Thus a row from block 2 must be played whenever column 1 is played. ▮

The fourth implication we would expect from the mimicking game methodology is that any time row-block 2 is played with positive probability column-block 2 must also be played with positive probability. However, this is not necessarily the case, and only the following weaker implication holds:

Claim 18 *If a row from row-block 2 is played with positive probability, then some column from* either *column-block 2 or 3 must be played with positive probability.*

Proof: Assume for the sake of contradiction that only column 1 is played. Then the row player receives 0 payoff. However, he has an incentive of 1 to play in the first row-block, contradicting the fact that we are in Nash equilibrium. Thus anytime the second row-block is played with positive probability, some column other than the first must sometimes be played. ▮

This above cycle of implications lets us relate the Nash equilibria of H' to those of H.

As a first step, let us show that the proposed map of Construction 13 from (r', c') to (r, c) is in fact well-defined. The issue here is that the final rescaling step might involve rescaling a zeros vector. To see that this will never happen, we show the following:

Claim 19 *In any Nash equilibrium (r', c') of H' some column other than the first must have positive weight, and some row outside the first block must have positive weight.*

Proof: For the first part, we note that from Claim 17, if the first column is played with nonzero probability then the second row-block must be played with nonzero probability, and thus by Claim 18 either the second or third column-block must be played with nonzero probability, as desired.

For the second part, that some row outside the first block must be played, we assume for the sake of contradiction that only rows from the first row-block are played. Note that if the first column is played, then by Claim 17 the second row-block must be played. Further, if the second column-block is played, then sequentially applying Claim 14, its corollary, and Claim 16, we see that the first column must also be played, in which case we are done as above. Thus if any column from the first or second block is played, we are done. Otherwise, if only columns from the third block are played, then we note that both players receive 0 payoff, while the row player could receive payoff greater than $\frac{2^j}{3(2^j-1)}$ by playing in the second row-block, a contradiction. Thus some row outside the first block must sometimes be played, and some column outside the first must sometimes be played, as desired. ▌

Thus, the transformation of Construction 13 is well-defined on Nash equilibria of H'. We now complete the proof of Theorem 12, that Construction 11 is in fact a Nash homomorphism.

Proof: We show that if (r', c') is a Nash equilibrium of H' then (r, c) is a Nash equilibrium of H.

We have two cases. For the first case, suppose no column from column-block 2 is played in c'. Thus in (r, c), column i is not played. Thus the row player payoffs in the second row-block of H' will be the matrix-vector product $R_{-i}c'_3$, and the column player payoffs in the third column-block will be $C^T_{-i}r'$ which, up to scaling (and adding a 0 for column i), are identical to the payoffs in game H when strategies (r, c) are played. Thus since (r', c') is an equilibrium of H', (r, c) is a Nash equilibrium of game H.

The second case, where some column from column-block 2 is played, is slightly more involved. We note that from Claim 14 and its corollary, the columns of column-block 2 are indeed played in proportion to the vector

$$\frac{1}{3(2^j-1)}(2^{j-1}, 2^{j-1}, 2^{j-1}, ..., 4, 4, 4, 2, 2, 2, 1, 1, 1).$$

Thus, since the sum of the weights in column-block 2 becomes the weight on the ith component of c under Construction 13, the payoffs for the row player in game H when the column player plays c are indeed proportional to the payoffs in game H' when the column player plays c'. As above, we can also see that the column player payoffs in the third block of H' are proportional to those in H outside of the ith column. We will now show that the payoff of the ith column in H is at least as large as any other payoff.

Note that, by the mimicking claims, since the second column-block of H' is played, the first column must also be played. Thus because (r', c') is a Nash equilibrium of H', the payoff of the first column must be at least as high as that of any other column. Further, note that the column player's payoffs in the first column of H' consist of zeros followed by the payoffs of the ith column of H. Thus since the first column of H' must receive at least as much payoff as any column in the third block, the ith column of H must receive at least as much payoff as any other column.

Thus the ith column has high enough incentive that it is a best response to c in H. Further, all the other columns and all the rows receive incentives exactly proportional to those in H'. Thus r and c are mutual best responses, and we conclude that the mapping from (r', c') to (r, c) maps Nash equilibria of H' to Nash equilibria of H, as desired. ▌

We now state and prove our main result.

Theorem 20 *There is a polynomial-time Nash homomorphism from the set of $m \times n$ rational payoff games that are expressible using j total bits in binary representation into the set of $\{0, 1\}$-games of size at most $(3j + 1)(m + n) \times (3j + 1)(m + n)$.*

Proof: Given a game G in this set, we follow the strategy outlined above. We first apply Construction 10 to precondition the row player payoffs. Then, for each of the n columns of the original game, we transform its entries into binary via an application of Construction 11. Note that each application of this construction increases the number of rows and columns by $3j$. Then we precondition the column player's payoffs, and apply the analogous translation homomorphism to each of the m original rows of the game. We note that since each of the constructions is a Nash homomorphism, we can clearly compose them like this. Further, since Construction 11 does not move or duplicate any of the non-$\{0, 1\}$ entries, the above $m + n$ applications of the construction will completely remove all non-$\{0, 1\}$ entries from the game. These transformations increase the number of rows and columns by $3j(m + n)$, as desired, and each transformation can clearly be done in polynomial time. Thus we have the desired Nash homomorphism. ▌

8 Conclusion

We have exhibited a polynomial-time Nash homomorphism from two-player rational-payoff games of k bits to $\{0, 1\}$-games of size polynomial in k. Thus the complexity of finding Nash equilibria of these two classes of games is polynomially related.

Very recently this result has been extended to the multi-player case, showing that n-player $\{0,1\}$-games are no harder than n-player general games[12]. It may be hoped that $\{0,1\}$-games could offer algorithmic insights into the general Nash problem.

9 Acknowledgement

We would like to thank Silvio Micali for many invaluable discussions during the course of writing this paper.

References

[1] V. Conitzer and T. Sandholm. Complexity results about Nash equilibria. *IJCAI* 2003, pp. 765–771.

[2] B. Codenotti and D. Stefankovic. On the computational complexity of Nash equilibria for (0,1)-bimatrix games. *Information Processing Letters*, 94(3):145–150, May 2005.

[3] I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1989.

[4] D. Koller and N. Megiddo. Finding mixed strategies with small supports in extensive games. *International Journal of Game Theory*, 1995.

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

[6] C. H. Papadimitriou. Algorithms, Games, and the Internet. *STOC* 2001, pp. 749–753.

[7] C. H. Papadimitriou. Computing Correlated Equilibria in Multi-Player Games. *STOC* 2005, pp. 49–56.

[8] C. H. Papadimitriou and T. Roughgarden. Computing Equilibria in Multi-Player Games. *SODA* 2005.

[9] R. Savani and B. von Stengel. Exponentially Many Steps for Finding a Nash Equilibrium in a Bimatrix Game. *FOCS* 2004, pp. 258-267.

[10] G. Schoenebeck and S. Vadhan. The Computational Complexity of Nash Equilibria in Concisely Represented Games. *ECCC* Report TR05-052.

[11] B. von Stengel. Computing equilibria for two-person games. *Handbook of Game Theory, Vol. 3,* eds. R. J. Aumann and S. Hart, pages 1723–1759, 2002.

[12] P. Valiant. Manuscript, July 2005.