On the Complexity of Nash Equilibria and Other Fixed Points (Extended Abstract)

Kousha Etessami
LFCS, School of Informatics
University of Edinburgh

Mihalis Yannakakis
Department of Computer Science
Columbia University

Abstract

We reexamine what it means to compute Nash equilibria and, more generally, what it means to compute a fixed point of a given Brouwer function, and we investigate the complexity of the associated problems. Specifically, we study the complexity of the following problem: given a finite game, Γ, with 3 or more players, and given $\epsilon > 0$, compute a vector x' (a mixed strategy profile) that is within distance ϵ (say, in l_∞) of some (exact) Nash equilibrium.

We show that approximation of an (actual) Nash equilibrium for games with 3 players, even to within any non-trivial constant additive factor $\epsilon < 1/2$ in just one desired coordinate, is at least as hard as the long standing square-root sum *problem, as well as more general arithmetic circuit decision problems, and thus that even placing the approximation problem in NP would resolve a major open problem in the complexity of numerical computation. Furthermore, we show that the (exact or approximate) computation of Nash equilibria for 3 or more players is complete for the class of search problems, which we call FIXP, that can be cast as fixed point computation problems for functions represented by algebraic circuits (straight line programs) over basis $\{+, *, -, /, max, min\}$, with rational constants. We show that the linear fragment of FIXP equals PPAD.*

Many problems in game theory, economics, and probability theory, can be cast as fixed point problems for such algebraic functions. We discuss several important such problems: computing the value of Shapley's stochastic games, and the simpler games of Condon, extinction probabilities of branching processes, termination probabilities of stochastic context-free grammars, and of Recursive Markov Chains. We show that for some of them, the approximation, or even exact computation, problem can be placed in PPAD, while for others, they are at least as hard as the square-root sum and arithmetic circuit decision problems.

1 Introduction

A wide variety of problems from many fields (economics, game theory, probability, etc.) can be cast in the form of finding a solution to a fixed point equation $x = F(x)$. Computing a Nash equilibrium is one prominent such problem that has attracted a lot of attention in economics, and more recently in the computer science community. Nash's theorem says that every (finite) game has an equilibrium, i.e., a set of mixed strategies for the players such that no player can improve its payoff by changing its strategy unilaterally [33]. Nash proved his theorem using Brouwer's fixed point theorem: every continuous function F from a compact convex body to itself has a fixed point. There are many other applications of Brouwer's theorem (and related fixed point theorems, e.g., Banach, Kakutani) such as price equilibria, values of games, probabilities of events in stochastic models and others. The problem is that the proof of Brouwer's theorem is nonconstructive, i.e., it establishes the existence of one or more fixed points without showing how to compute one.

The problem of computing Nash Equilibria, and more generally computing fixed points of Brouwer functions, has a long and rich history, dating back at least to the fundamental algorithm of Scarf [37]. Given a continuous function F and $\epsilon > 0$, Scarf partitions the domain into simplices of sufficiently small diameter δ (depending on ϵ and the modulus of continuity of the function F) and navigates through the simplices to produce a point x' such that $||F(x') - x'||_\infty < \epsilon$. The point x' is *almost* fixed by F, but it may be far from the actual fixed points. Let us call such a point x' a *weak* ϵ-fixed point (weak ϵ-FP), to distinguish it from a point x that is *near* a fixed point x^* (i.e., $||x^* - x'||_\infty < \epsilon$) which we will call a *strong* ϵ-fixed point. (The names are due to the fact that for the kind of 'well-behaved' functions that are encountered in most applications, weak approximation reduces to strong; see section 2 for a formal statement.) To obtain (the existence of) an actual fixed point, the simplicial partition can be refined more and more so that the diameter, δ, of the simplices tends to 0; then the sequence of weakly approximate fixed points must have (by compactness) a subsequence that converges to a point, which must be an actual fixed point x^*. However, as Scarf pointed out ([37]), this part is nonconstructive in general. A number of other algorithms along related lines have been proposed both for general fixed points and for Nash equilibria. Note that the goal of the algorithms is to compute a (any) fixed point or Nash equilibrium, not a specific one; computing a specific one, for example the one with highest payoff, is NP-hard [20].

In [35], Papadimitriou introduced a complexity class, PPAD, to capture problems like (approximate) fixed points and Nash, and showed that a certain linearly interpolated version of the Brouwer fixed point problem is complete for PPAD. The class PPAD lies between (the search problem versions of) P and NP. The Nash problem has been investigated intensely recently in the TCS community, and last year in a breakthrough set of papers [10, 7, 8] it was shown that computing an (exact) Nash equilibrium for 2 players is PPAD-complete, and so is the problem of computing a ϵ-Nash Equilibrium (ϵ-NE) for any number of players. An ϵ-NE is a profile of mixed strategies where no player can improve its own payoff by more than ϵ by switching strategies unilaterally. ϵ-NEs correspond in a precise sense (they are polynomially equivalent, see section 2) to weak ϵ-fixed points of Nash's function.

One major difference between the 2- and 3-player case is that in the 2-player case there are always rational NEs (the game payoff table is assumed to be rational) and thus can be computed exactly, whereas for 3 and more players this is not the case: in general all NEs can be irrational. The same phenomenon occurs in most applications of Brouwer's theorem: the domain is not discrete (the theorem depends after all on the function being continuous) and the fixed points are in general irrational. What does it mean then to compute fixed points and equilibria in this case? A natural goal is to compute or approximate such a solution up to a specified precision, e.g., compute the first k bits in the binary representation, or compute a (strong) ϵ-approximation of a solution, i.e., a point within ϵ of a fixed point or an equilibrium. Note that this is different from a ϵ-NE, which can be very far from any actual NE (see Corollary 4 for a precise statement about how far it can be).

So how hard is it to compute or approximate to within distance ϵ a Nash equilibrium (any one) for 3 or more players? Is it in PPAD? Is it even in NP? And if not, what is the right class that captures these problems, i.e., the class for which they are complete? These are some of the questions we address in this paper. First, we show that placing the (strong) approximate Nash problem in NP will imply a breakthrough on longstanding open problems. In the *Square Root Sum* problem (SQRT-SUM for short) we are given positive integers $d_1, \ldots, d_n$ and k, and we want to decide whether $\sum_{i=1}^n \sqrt{d_i} \leq k$. This problem arises in many contexts, e.g., in geometric computations where the square root sum represents the sum of Euclidean distances between given pairs of points with integer (or rational) coordinates; for example, determining whether the length of a specific spanning tree, or a TSP tour of given points on the plane is bounded by a given threshold k amounts to answering such a problem. This problem is solvable in PSPACE, but it has been a major open problem since the 1970's (see, e.g., [18, 34, 40]) whether it is solvable even in NP (or better yet, in P). A related (and in a sense more powerful) problem is the PosSLP problem: given a division-free straight-line program, or equivalently, an arithmetic circuit with operations $+, -, *$ and inputs 0 and 1, and a designated output gate, determine whether the integer N that is the output of the circuit is positive. As shown in [1], the class $P^{\texttt{PosSLP}}$, i.e., decision problems that can be solved in polynomial time using an oracle for PosSLP, is equal to the Boolean part (restriction to inputs over $\{0,1\}$) of decision problems over the reals that can be solved in polynomial time in the Blum-Shub-Smale model of real computation [4] using algebraic numbers as constants. This is a powerful model, which is equivalent to the unit cost algebraic RAM model (operations on arbitrary numbers take unit time); in particular the SQRT-SUM problem can be decided in polynomial time on this model [40]. Allender et al. [1] showed that PosSLP and SQRT-SUM lie in the Counting Hierarchy. We show that SQRT-SUM and PosSLP reduce to the problem of (strong) approximation of 3-player Nash equilibria. Specifically, for any $\epsilon > 0$, they reduce to this problem: given a game, Γ, with the property (promise) that either, in every NE, a particular strategy is played with probability 0, or, in every NE, it is played with probability at least $1 - \epsilon$, decide which of the two is the case for Γ. Note: any non-trivial approximation of an actual NE, say to within any constant distance $c < 1/2$ in the desired coordinate, would enable us to distinguish the two cases for small enough $\epsilon > 0$.

Second, we define a class FIXP of problems, for which Nash for 3 players (or more) is complete. FIXP is the class of search problems that can be expressed as fixed point problems for functions represented by polynomial size algebraic circuits over the basis $\{+, -, *, /, \max, \min\}$ with rational constants. Nash is complete both in the sense of exact and approximate computation. PSPACE is an upper bound on the complexity of the discrete computational tasks associated with search problems in FIXP (eg. the approximation and decision problems). We know no better bound in general.

There are a number of other well-studied problems from different areas that can be cast as fixed point problems of suitably defined functions given by simple algebraic formulas, and thus are also in FIXP. We discuss several of them in this paper. Despite the rich theory developed over the years, and extensive work on these models, the complexity of many fundamental problems is open. *Stochastic games* were first introduced by Shapley [39] in 1953; and have been extended in various directions and studied extensively since then. A simpler version, called simple stochastic games was introduced by Condon [9] in computer science and has attracted a lot of attention. The quantities of interest in these games are to compute or bound the values of the games (they are unique) and find optimal strategies for the players. *Branching processes* (BP) were first introduced, in the 1-type case, by Galton and Watson in the 19th century to model population dynamics, and later generalized to the multi-type case by Kolmogorov and studied by him and Sevastyanov ([29]) and others. They are a basic probabilistic model for many applications (e.g., biological processes and many others [23, 24, 27]). The most basic quantities of interest here are the extinction probabilities of entity types. *Stochastic context-free grammars* (SCFG) are a model in common use in Natural Language Processing [32] and biological sequence analysis [13]. *Recursive Markov chains* (RMC), a more powerful model that encompasses in a precise sense both BPs and SCFGs, were introduced in [16] to model recursive probabilistic programs (see also [15] for an equivalent model). Basic quantities of interest here are the termination probabilities.

In all of the above models, one can define an appropriate function F such that the desired quantities x^* are a fixed point of F; in fact in all cases except for general RMCs, the function and the domain can be defined so that x^* is the *unique* fixed point (for RMCs it is the least nonnegative fixed point). In all these problems, the function F is just a tool to get a handle on the problem. For problems like Nash, weak fixed points have a game-theoretic/economic meaning (cf. ϵ-NEs) and thus are also of interest. For the above models however, weak ϵ-fixed points have no significance, unless they help us find, approximate, or answer questions about, the quantities of interest which are the actual fixed points. As an example, consider the approximation of the value of simple stochastic games. We can compute easily in polynomial time a weak ϵ-fixed point of the associated function F for $\epsilon > 0$ a constant or even inverse polynomial ($1/n^c$ for any c); however, this is not useful since it does not tell us how to compute even a constant approximation to the value of the game, which is what we are interested in.

The distinction between strong (near) and weak (almost) approximate fixed points was noted early on, sometimes with statements which on the surface seem contradictory. For example, Scarf in his original paper [37], remarks that obtaining strong fixed points from weak fixed points is non-constructive for general mappings. On the other hand, Anderson [2], among several "almost implies near" theorems, notes that for every Brouwer function F and $\epsilon > 0$ there is a $\delta > 0$ such that every weak δ-fixed point is a strong ϵ-fixed point (a fact also observed earlier). What is going on? Scarf's remark concerns algorithms that use F as a black box, i.e., work for all possible instances of all problems (indeed, impossibility results are known for strong approximation of fixed points in the black-box oracle model); Anderson's results concern a fixed function analysed with respect to the precision, i.e., a single instance of a problem (for example, Nash for a specific game Γ). In a concrete problem like Nash it is important to set up the framework properly to study the complexity as a function of the instance size. Indeed, our results show that from a quantitative computational perspective, "almost" emphatically does not imply "near" for Nash and other classes of Brouwer functions (see Corollary 4 for a precise statement).

For some types of functions, weak and strong approximation for sufficiently small ϵ and even exact computation can be related. We define a general class of *polynomial piecewise linear functions*, and show that for them exact fixed point computation is in PPAD; the piecewise linear class includes simple stochastic games, the discretized Brouwer problem of [35] which is PPAD-complete, and the subclass of FIXP, denoted *Linear-FIXP*, where the circuits do not use multiplication and division, except by a constant. Indeed, (exact) fixpoints of polynomial piecewise linear functions, Linear-FIXP, and PPAD are all polynomially equivalent. For Shapley's game (which has a nonlinear F) strong approximation is reducible to weak for sufficiently small ϵ and is also in PPAD. However, bounding the value of the game, e.g., deciding whether Player 1 can achieve reward $\geq r$ is harder: we show that it is at least as hard as `SQRT-SUM`, and hence placing it in PPAD (or NP) would solve a longstanding open problem.

For branching processes and SCFGs we show that the problem is in FIXP; the challenging part here is to constrain the domain of the function in a polynomial-time computable way so that we get a Brouwer function with the desired probabilities forming the *unique* fixed point in the domain. The decision problem (comparing the probabilities with a given rational r) is also SQRT-SUM-hard (shown in [16]); we do not know the status of the approximation problem for the relevant probabilities. For the more general Recursive Markov Chain model, we show that approximating the probabilities within any non-trivial constant additive factor $c < 1/2$ is at least as hard as SQRT-SUM and PosSLP.

The rest of this extended abstract is organized as follows. In Section 2 we set up the framework, and give basic definitions and properties. In Section 3 we define the class FIXP, and study the complexity of Nash equilibria. In Section 4 we define and discuss the class of piecewise-linear fixed point problems, and stochastic games. Section 5 concerns branching processes, SCFGs, and RMCs. *Due to space constraints, most proofs are omitted.*

2 Preliminaries

We describe a general framework for search problems where the solution sets may be real-valued, and thus not computable exactly. A *search problem* Π has a set of instances, represented by strings over a fixed finite alphabet Σ, and each instance I has an associated set $Sol(I)$ of *solutions*. As usual, it is assumed that, given a string over Σ, one can determine in polynomial time if the string is an instance of the problem. The problems we will be interested in here (equilibria, fixed points, probabilities, etc.) are *total*: every instance I has a nonempty set $Sol(I)$ of solutions. For some problems there may be a unique solution (for example, probabilities of certain events in a stochastic model), while for others there may be multiple solutions (for example, NEs of a game). Unlike usual discrete problems, solutions here are in general real-valued vectors of finite dimension, polynomial in the size $|I|$ of the instance. We would like to solve the following problems:
1. Exact Computation: Given input I, compute a solution x in $Sol(I)$. Note that if there are multiple solutions, then any one of them is a correct output. If there is a rational solution we can output it explicitly, but we cannot do this for irrational solutions, which are common in the problems under consideration. In this case we can only bound the solution or approximate it up to a desired precision. Attention has to be paid in the formulation of the problem to stay faithful to the search problem and not make it harder: in particular, if there are multiple solutions, any one of them should enable us to answer the question. There are several reasonable ways to pose the problem, and this can make a difference in the complexity.
2. Partial Computation: Given instance I and integer $k > 0$, compute the k most significant bits of some solution (any one). We would like to do this in time polynomial in $|I|$ and k.
3. Decision Problem: Given instance I, rational vector r and a comparison operator vector θ (e.g., $\geq, \leq$, etc.) return a truth value that holds for at least one of the solutions, i.e. if all solutions x satisfy $x\theta r$ then return 'Yes', if none satisfies $x\theta r$ then return 'No', and if some do and some do not, then either answer is correct. Alternatively, and more usually, the decision question may be posed on a particular entry, e.g., $x_1 \geq r$?, for $r \in \mathbb{Q}$. This formulation reflects the standard way of turning optimization problems and other problems with output to decision problems. We would like running time polynomial in $|I|$ and the size (number of bits) of r. Note that the **Existence question**, given instance I and rational r, is there a solution x with $x_1 \geq r$?, is equivalent to the decision question for problems with unique solutions, but not otherwise. For many search problems (e.g., 2-player NEs, [20]) the existence question is NP-hard while the search problem is not.
4. Approximation Problem: Given instance I and rational $\epsilon > 0$, compute a vector x that is within (additive) ϵ of some solution, i.e., there is a $x^* \in Sol(I)$ such that $|x^* - x|_\infty \leq \epsilon$. Alternatively we could require to approximate only a particular entry of a solution vector, e.g., approximate x_1^* within additive ϵ. We would like polynomial time in $|I|$ and $\log(1/\epsilon)$; this permits approximation within 2^{-k} in time polynomial in I and k. For several problems we will show that the approximation problem is hard for some class, by showing hardness of a corresponding **Promised Gap Decision Problem** PGD(a,b): Given instance I, rationals $a < b$, and the promise that either all solutions $x \in Sol(I)$ have $x_1 \leq a$ or they all have $x_1 > b$, determine which of the two is the case.

There are some simple relations between these problems, but in general they are not equivalent, i.e. for a search problem Π, the associated Partial Computation, Decision, and Approximation problems may well have different complexity.
Example: Nash Equilibria. In a (normal form) game with k players, the instance I consists of k (disjoint) finite sets of (pure) strategies $S_i, i = 1, \ldots, k$, and k rational-valued payoff functions u_i from the product strategy space $S = \Pi_i S_i$ to $\mathbb{Q}$. A mixed *strategy profile* x is a non-negative vector of length $\sum_i |S_i|$ that is a probability distribution on the set of (pure) strategies of each player. The (expected) payoff $U_i(x)$ of x for

player i is $\sum x_{1,j_1} \dots x_{k,j_k} u_i(j_1, \dots, j_k)$ where the sum is over all $(j_1, \dots, j_k) \in S$ and $x_{i,j}$ is the probability with which player i plays strategy j. A *Nash equilibrium* (NE) is a strategy profile x^* such that no player can increase its payoff by switching its strategy unilaterally. Every finite game has at least one NE [33]. Two-player games have rational NEs, thus exact computation is possible, and as shown in [7] the problem is PPAD-complete.

For three and more players, the NEs can in general all be irrational, so we would like to approximate one (or solve the related decision problem); this is one of the main subjects of this paper. Note that approximating a (actual) NE is different from the notion of an ϵ-NE studied previously and shown PPAD-complete [10, 21, 8]. An ϵ-NE is a strategy profile x such that no player can improve its payoff by more than ϵ by switching unilaterally to another strategy. That is, a profile x is an ϵ-NE if it is *almost* at equilibrium, rather than being *near* an equilibrium. In general, an ϵ-NE can be very far from all actual NEs. On the other hand, all profiles that are sufficiently close to a NE are also ϵ-NEs: for every game I and $\epsilon > 0$, we can take a δ of size polynomial in the size of I and ϵ, such that every profile x' such that $|x' - x^*|_\infty < \delta$ for some NE x^*, is an ϵ-NE ([30]). For this reason, we will call the problem of computing a strategy profile that is close to a (actual) NE *strong approximation* to distinguish it from the ϵ-NE notion of approximation. ∎

Nash proved his theorem in [33] using Brouwer's theorem: he showed that for every finite game, Γ, the Brouwer fixed points of the following function, F_Γ, are the NEs of Γ: $F_\Gamma(\mathbf{x})_{(i,j)} \doteq \frac{\mathbf{x}_{i,j} + \max\{0, g_{i,j}(\mathbf{x})\}}{1 + \sum_{l=1}^{m_i} \max\{0, g_{i,l}(\mathbf{x})\}}$, where $g_{i,j}(\mathbf{x})$ is the "gain" of player i if he switches to pure strategy j (which is a polynomial in $\mathbf{x}$).

In this paper we study search problems that can be cast in a fixed point framework: every instance I of the search problem Π is associated with a continuous function F_I mapping a convex compact domain D_I to itself, such that the set $Sol(I)$ of solutions is $Fix(F_I)$, the set of fixed points of F_I. More generally, we may allow polynomial time reduction from the search problem Π for I to the fixed point problem for F_I, i.e., it is not necessary to require $Sol(I) = Fix(F_I)$.

For simplicity, it is customary to take the domain D_I to be a box or a simplex, usually the *unit cube* $\square_n = [0,1]^n$ or the *unit simplex* $\Delta_n = \{x \mid x \geq 0,\ \sum_i x_i = 1\}$. If we have a function F on a convex compact domain D, we can embed D into a suitable box or simplex D' that contains it, define a continuous functions π from D' to D that is the identity on D (e.g., project D' onto D) and compose π with F to form a continuous function on D' whose FPs are the FPs of F. The box or simplex can be translated and scaled to a unit cube or unit simplex. For the cases we will be interested in, an appropriate mapping π can be constructed.

Consider a search problem Π that has been cast as a fixed point problem and the class $\mathcal{F}$ of functions F_I indexed by the instances I of Π. We say $\mathcal{F}$ is *polynomially computable* if for every instance I and rational vector $x \in D_I$, the image $F_I(x)$ is rational and can be computed in time polynomial in the size of I and of x. $\mathcal{F}$ is called *polynomially continuous* if there is a polynomial $q(z_1, z_2)$ such that for all instances I and all rational $\epsilon > 0$, there is a rational $\delta > 0$ such that $size(\delta) \leq q(|I|, size(\epsilon))$ and such that for all $x, y \in D_I$, $|x - y|_\infty < \delta \Rightarrow |F_I(x) - F_I(y)|_\infty < \epsilon$; this definition is robust with respect to the norm. $\mathcal{F}$ is called *polynomially contracting* with respect to norm l_k, $k \in \mathbb{N} \cup \{\infty\}$, if there is some polynomial $q(z)$ such that for all instances I there is some rational $\beta < 1 - 2^{-q(|I|)}$, such that for all $x, y \in D_I$, $|F_I(x) - F_I(y)|_k < \beta |x - y|_k$.

We are interested in the problems mentioned before: compute an exact fixed point if possible, or the decision and approximation problems. We refer to the approximation of a fixed point as the *strong approximation* problem to contrast it with the following version that is specific to the formulation of the search problem as a fixed point problem:

Weak Approximation: Given instance I and given a rational $\epsilon > 0$ as input, compute a rational vector $x' \in D_I$ such that $|F_I(x') - x'|_\infty < \epsilon$.

Remark: A search problem may be expressible as a fixed point problem in several different ways, using different functions. For example, besides Nash's function, there are other functions whose fixed points also are the NEs [19, 3]. In general, the notion of a weak approximation depends on the function that is used. The notion of a (strong) approximation does not depend on the function, it is inherent to the search problem itself. We now give basic facts about the relationship between these different problems (proofs omitted).

Proposition 1 *Let $\mathcal{F}$ be the class of functions associated with a fixed point search problem.*
1. If $\mathcal{F}$ is polynomially continuous, then weak approximation for $\mathcal{F}$ is P-time (many-one) reducible to strong approximation.
2. If $\mathcal{F}$ is polynomially continuous and polynomially computable, then weak approximation for $\mathcal{F}$ is in PPAD.
3. If $\mathcal{F}$ is polynomially contracting and polynomially computable, then strong approximation is P-time reducible to weak approximation. Consequently, the

strong approximation problem for such a class of functions is in PPAD.
4. Computing an ϵ-NE for a game, Γ, is P-time equivalent to computing a weakly approximate fixed point for Nash's function F_Γ.

3 Nash Equilibria and the class FIXP

We now define a class of fixed point problems, **FIXP**, for which the Nash equilibrium problem (for 3 or more players) will be shown to be complete. The class of fixed point problems induces a corresponding class for each of the associated types of questions: e.g., a class of decision problems, a class of (strong) approximation problems, etc. As usual in the definition of classes (cf. PLS, PPAD, MAXSNP, etc.), it is convenient to define a syntactic version and then close it under (polynomial-time) reductions. Recall that in the framework of search problems cast as fixed points, each instance I is associated with a continuous function F_I from a compact convex domain D_I to itself, such that $Sol(I) = Fix(F_I)$. The basic characteristic of the class FIXP is that (i) the function F_I is represented by an *algebraic circuit* (or *straight line program*) C_I over the basis $\{+, -, *, /, \max, \min\}$ using rational constants, that computes the function over the domain, and (ii) there is a polynomial time algorithm that computes the circuit from I. Note that the circuit operates on real numbers, but the algorithms we study do not do any computation on reals; the circuit is a finite representation of the function, just like a formula, but more succinct of course. The underlying model of computation for us is still the standard discrete Turing machine.

The circuit C_I is a sequence of gates $g_1, \ldots, g_r$, where for $i = 1, \ldots, n$, $g_i = x_i$ is an input variable, for $i = n+1, \ldots, n+m$, $g_i = c_i \in \mathbb{Q}$ is a rational constant (encoded in the standard way, with numerator and denominator given in binary), and for $i > n + r$, we have gates $g_i = g_j \circ g_k$, with $j, k < i$, where the operator is $\circ \in \{+, -, *, /, max, min\}$ (infix max and min have the obvious meaning). The last n gates are the output gates. Of course some of the operations are redundant (can be defined in terms of others), and we could just have the constant 1 and build the other rationals, but we include them for convenience. We will also standardize the domain here for simplicity and assume that it is the unit cube $\square_n = [0,1]^n$. (Reductions can be used to embed other domains into it.) The circuit C_I represents the function F_I. Thus, if we input any vector $x \in \square_n$ into C_I it will output a vector in $\square_n$; in particular this means that the evaluation of all the gates is well-defined and there is no division by 0.

We close the class by reductions. In the usual case of discrete search problems, a reduction from problem A to problem B consists of two polynomial-time computable functions, a function f that maps instances I of A to instances $f(I)$ of B, and a second function g that maps solutions y of the instance $f(I)$ of B to solutions x of the instance I of A. The difference here is that the solutions are real-valued, not discrete, so we have to specify what kind of functions g are allowed. It is sufficient for our purposes in this paper to restrict the reverse function g to have a particularly simple form: a separable linear transformation with polynomial-time computable rational coefficients; that is, $x = g(y)$, where each $g_i(y)$ is of the form $a_i y_j + b_i$ for some j, where a_i, b_i are rationals computable from I in polynomial time. A reduction of this form from A to B induces corresponding P-time reductions for the Decision and the (Strong) Approximation problems.

Corresponding to the class FIXP of search problems (with real solutions) there is a class of Decision problems, a class of Approximation problems and a class of Partial computation problems. To emphasize the distinction between them, we can attach a subscript to FIXP, denoting these classes of associated problems as $FIXP_d$, $FIXP_a$, $FIXP_{pc}$, respectively. Note that these are classes of discrete search problems (hence their complexity can be studied in the standard Turing machine model), as opposed to FIXP which is a class of, in general, real-valued search problems (whose complexity can be studied in a real computation model, e.g. [4]). By appealing to PSPACE decision procedures for the existential theory of reals [6, 36], we can readily obtain the following.

Proposition 2 *The Partial Computation, Decision, (Strong) Approximation and Existence versions of all problems in FIXP can be solved in PSPACE.*

Recall `SQRT-SUM` and `PosSLP` from Section 1.

Theorem 3

1. *The* `SQRT-SUM` *and* `PosSLP` *problems are P-time (many-one) reducible to the promised-gap-decision problem PGD(0,1) for 4-player Nash.*
2. *The* `SQRT-SUM` *and* `PosSLP` *problems are P-time reducible to PGD(0,1 $-\ \epsilon$) for 3-player Nash for any constant ϵ (and even with $\epsilon = 2^{-poly}$).*

Remarks: 1. For 3 players, it is not hard to show that the PGD(0,1) problem is in PPAD, hence showing that it is as hard as `SQRT-SUM` or `PosSLP` would imply that these problems are in NP∩coNP, which would be a breakthrough (and looks very doubtful at present for `PosSLP`). 2. Similar results hold for the approximation of the payoff of one player, or of all the players, in *any* NE, i.e., formalizing the questions again as promised gap decision problems on payoffs. 3. All the reductions of the theorem have the property that the particular entry (strategy probability) x_1 of the PGD question has the same value in all NEs of the constructed game, thus there is no ambiguity; in fact in the reduction from SQRT-SUM to 4-player Nash the game is guaranteed to have a unique NE. 4. A corollary of the proof is this:

Corollary 4 *For every n there is a 4-player game, Γ_n, of size $O(n)$, with an ϵ-NE, x', where $\epsilon = 1/2^{2^{\Omega(n)}}$, such that x' has l_∞-distance 1 from every actual NE. For 3-players, the same statement holds with distance 1 replaced by $1 - 2^{-poly}$.*

To prove Theorem 3 we give separate reductions from SQRT-SUM and PosSLP. For part 1, from a given instance of SQRT-SUM (or PosSLP) we construct a 4-player game Γ such that player 4 has only two strategies in the game and at any NE it always plays the same pure strategy (one of the two) with probability 1 and the other with 0; determining which of the two strategies it plays solves the SQRT-SUM (or PosSLP) instance. For part 2, we use a general reduction from many-player to 3-player games by Bubelis [5], extending it to give different weights to different players:

Lemma 5 *For any (finite) game Γ with d players and any d positive numbers $\lambda_i > 0, i = 1, \ldots, d$ with $\sum_{i=1}^{d} \lambda_i = 1$, we can construct a 3-player game Γ' such that the pure strategies of player 1 in Γ' correspond 1-1 to the pure strategies of all the players in Γ, and the following properties hold.*
1. For every NE y of Γ', if y_{ij} denotes the probability with which player 1 plays in y strategy (i, j), i.e., the jth strategy of player i in Γ, then the vector x defined by $x_{ij} = y_{ij}/\lambda_i$ is a NE in Γ.
2. Conversely, for every NE x of Γ, there is a NE y of Γ' such that player 1 plays in y each strategy (i, j) with probability $\lambda_i x_{ij}$. Furthermore, if x is fully mixed (all strategies are in the support), then there is a unique corresponding such NE y in Γ' and it is also fully mixed.
3. If the payoffs in Γ and the λ_i's are rationals, then the construction of Γ' can be done in polynomial time in the size of Γ and the λ_i's.

Part 2 of Theorem 3 follows from part 1 by applying Lemma 5 with $\lambda_4 = 1 - \epsilon$ and $\lambda_1 = \lambda_2 = \lambda_3 = \epsilon/3$.

One of the implications of the theorem is that computing the first bit of a (any) NE is at least as hard as SQRT-SUM and PosSLP. In [1] it is shown that computing a specified bit of the integer computed by an arithmetic circuit is #P-hard, when the index of the desired bit is given in binary. Using our reductions from PosSLP, we can deduce a similar result for NEs. The key property is that certain probabilities in the game are shifts of the numbers computed by the arithmetic circuit.

Corollary 6 *Given a 3-player game and i in binary, it is #P-hard to compute the i-th bit of the probability of the first strategy of player 1 in a (any) NE.*

We show that Nash exactly characterizes the search problems that can be cast as fixed points of functions represented by algebraic circuits:

Theorem 7 *The Nash equilibrium problem for 3 (or more) players, is FIXP-complete. In particular, the corresponding Decision, (Strong) Approximation and Partial Computation problems are complete respectively for FIXP$_d$, FIXP$_a$, and FIXP$_{pc}$.*

Sketch. Regarding membership in FIXP, the Nash function F_Γ of a game is given by an algebraic circuit (in fact a formula); furthermore we can suitably embed the domain Δ (which is the product of unit simplices for the players) in the unit cube $\square_n$ (n the total number of pure strategies), i.e., we can define a function $\pi : \square_n \to \Delta$ by a suitable algebraic formula so that π is the identity on Δ, and the composed function $F_\Gamma \circ \pi$ on $\square_n$ has the NEs as its fixpoints. For the hardness part, we first subject the circuit C_I of the instance I to a series of (polynomial) transformations that bring it to an equivalent 'normal' form C' which uses in addition a conditional assignment gate (with restricted condition), so that C' has the property that for every input $x \in \square_n$, all gates of C' have value in $[0, 1]$ and $C'(x) = C_I(x)$. Then we reduce to the Nash problem with a fixed number of players using gadgets for the gates; the gadgets for $+, -, *, \max, \min$ are similar to gadgets of [10, 7], and we also have gadgets for division and conditional assignment. Finally, we apply Lemma 5 to reduce to a 3-player game Γ, picking suitable λ's so that the fixpoints x of I correspond to NEs y of Γ, and the x-values are multiples (in fact by powers of 2, i.e. binary shifts) of corresponding y-values. Details in the full paper. ■

As another example of a FIXP-complete problem, consider the following exchange equilibrium problem [38]. We have m agents and n commodities. For each vector p of prices for the commodities, each agent l has an (positive or negative) excess demand $g_i^l(p)$ for each commodity i. The standard assumptions are that the functions $g_i^l(p)$ (i) are homogeneous of degree 0, thus the price vectors may be normalized to lie on the unit simplex Δ_n, (ii) they satisfy Walras' law $\sum_{i=1}^{n} p_i g_i^l(p) = 0$, (iii) they are continuous on the unit simplex. Let $g_i(p) = \sum_l g_i^l(p)$ be the (total) market excess demand for each commodity i. The functions $g_i(p)$ clearly satisfy the same constraints. A vector p

of prices is an *equilibrium* if $g_i(p) \leq 0$ for all i, with equality for all commodities i that have $p_i > 0$. There is always at least one equilibrium, and the proof is via Brouwer's theorem. Namely, the equilibria are the fixed points of the function $F : \Delta_n \mapsto \Delta_n$, defined by the formula $F_i(p) = \frac{p_i + \max(0, g_i(p))}{1 + \sum_k \max(0, g_k(p))}$. Clearly, if the functions g_i^l, and hence also g_i, are defined by algebraic circuits, then so is the function F and the exchange equilibrium problem is in FIXP. Conversely, Brouwer's theorem can be derived from the equilibrium theorem [41] and the proof gives a reduction. Namely, given a Brouwer function $f : \Delta_n \rightarrow \Delta_n$, one can define a (total) market excess demand function $g : \Delta_n \rightarrow R^n$ where $g(p) = f(p) - (\langle p, f(p) \rangle / \langle p, p \rangle) p$. One can see that g satisfies the constraints of an excess demand function (e.g., $\langle p, g(p) \rangle = 0$ for all p , Walras' law) and hence has an equilibrium. Furthermore, any price equilibrium is a fixed point of f. Clearly, if f is given by an algebraic circuit we can construct a circuit for g. Thus:

Proposition 8 *The exchange equilibrium problem with excess demand functions given by algebraic circuits over* $\{+, -, *, /, \max, \min\}$ *is FIXP-complete.*

4 Piecewise Linear Functions, PPAD, and Stochastic Games

We will define a general class of fixed point problems with piecewise linear, polynomial-time computable functions and show that (i) they have rational fixed points and (ii) the exact computation problem is in PPAD. The simple stochastic games of Condon [9] are an example of such a problem. We also consider the more general stochastic games of Shapley [39], which are nonlinear.

Consider a fixed point search problem Π: every instance I is associated with a continuous function F_I on a (convex compact) domain D_I, which we assume for convenience to be the unit cube $\square_n$ (n polynomial in $|I|$). We say that Π is a **polynomial piecewise linear** problem if the following hold: The domain is divided by hyperplanes into polyhedral cells, the function F_I is linear in each cell and is of course continuous over the whole domain. The coefficients of the function in each cell and of the dividing hyperplanes are rationals of size bounded by a polynomial in $|I|$. These are not given explicitly in the input, in fact there may be exponentially many dividing hyperplanes and cells. Rather, there is an underlying (polynomial-depth) arithmetic decision tree T_I for deciding if a point x is in the domain and determining its cell, using linear comparisons of the form $ax \leq b$ (a, b poly-size rationals), and a polynomial-time algorithm for tracing the appropriate path in the tree for a given point x till it reaches a leaf, where it outputs the appropriate linear function $F_I(x)$. Formally, there is an oracle algorithm that runs in time polynomial in $|I|$ which generates a sequence of queries of the form $ax \leq b$? adaptively (i.e., the next query depends on I and the sequence of previous answers), and at the end outputs 'No' (i.e., x is not in the domain) or outputs the coefficients c, c' of the function $F_I(x) = cx + c'$.

Examples:

1. **Simple Stochastic Games (SSG).** This is a two-player game on a directed graph $G = (V, E)$ that has two sink nodes labeled 0 and 1, and the rest of the nodes are partitioned into three sets, V_0 (random), V_1 (max), V_2 (min). The edges out of each random node are labelled with rational probabilities that sum to 1. A token is placed initially at a node s and then moved along edges until it reaches a sink. If the token is at a node u in V_0 then the edge is chosen randomly according to the probabilities, if $u \in V_1$ then it is chosen by Player 1 who tries to maximize the probability of reaching sink 1, and if $u \in V_2$ then the edge is chosen by Player 2 who tries to minimize the probability of reaching sink 1. For every starting node s, there is a well-defined value x_s of this (zero-sum) game, which is the probability that the token reaches eventually sink 1 if both players play optimally. The goal is to compute the value x_s for a specific node s or for all nodes s. It is known that the values are rational of bit complexity polynomial in the input, and that the players have deterministic optimal strategies. The decision problem $x_s \geq 1/2$? (does Player 1 win with probability at least 1/2) is in $NP \cap coNP$, and it is a well known open problem whether it is in P.

The values x_s satisfy a system of equations $x = F(x)$, which are as follows. If u is the 0 sink or the 1 sink then $x_u = 0$ or $x_u = 1$ respectively; if $u \in V_0$ then $x_u = \sum_v p_{uv} x_v$, where the sum ranges over all edges (u, v) with p_{uv} the corresponding probability; if $u \in V_1$ then $x_u = \max\{x_v | (u, v) \in E\}$; if $u \in V_2$ then $x_u = \min\{x_v | (u, v) \in E\}$. In general there may be multiple solutions, however one can transform by a small perturbation δ any SSG game Γ to a 'stopping' game Γ' whose system has a unique solution in the unit cube [9]; for a suitably small δ the *exact* values of Γ can be obtained efficiently from the values of Γ' by rounding or by solving a linear system of equations. Clearly max and min, and hence also F are piecewise linear functions. In this example, the dividing hyperplanes are of the form $x_v = x_w$, for pairs of nodes v, w that are successors of the same node in V_1 or V_2 (in this case there are a polynomial number of hyperplanes).

2. **Linearly interpolated functions.** The following model of Brouwer functions F is used in [35]. The input is an integer N (in binary) and a Turing ma-

chine M, which given a "grid" point x in the unit cube $\Box_n$ with coordinates multiples of $1/N$ returns in polynomial time a displacement vector $\mu(x)$ such that $F(x) = x + \mu(x) \in \Box_n$ (the displacement is also constrained to satisfy $|\mu(x)| \leq 1/N^2$, but this restriction is actually not necessary as we'll see). The function then is extended to a piecewise linear map throughout $\Box_n$, by partitioning $\Box_n$ into $1/N$ cubelets along the grid hyperplanes, partitioning each cubelet into simplexes in a standard simplicization, and then interpolating the values at the vertices of each simplex. The cells here are the simplexes. It is not hard to give an algorithm that determines efficiently for a given $x \in \Box_n$ the simplex that contains it, using binary search and linear tests. Evaluate F at the vertices of the simplex and compute the linear form of the function $F(x)$ by interpolation. Note: Every Brouwer function F (say on $\Box_n$) can be approximated by a linearly interpolated function G by imposing a suitable grid. If F is polynomially continuous then a fixed point of G (for a suitably small but polynomially encodable grid size) is a weak ϵ-fixed point of F, but in general may be very far from any actual fixed point of F (cf. 3-player Nash).
3. **2-Player Nash.** Nash's function is nonlinear even for 2 players. However, we can use another function [22] which becomes piecewise linear in the 2-player case. For a mixed strategy profile x, let $u(x)$ be a vector which gives the expected payoff of each pure strategy with respect to the profile x of the other players, let $h(x) = x + u(x)$, and let $G(x) = \pi(h(x))$ be the projection of $h(x)$ on the domain Δ (product of the unit simplices of the players). Specifically, the projection π can be constructed explicitly as follows: for any vector y over the set of pure strategies S, choose for each player i a value v_i so that $\sum_{j \in S_i} \max(y_{ij} - v_i, 0) = 1$ (there is a unique such v_i) and define $\pi(y)_{ij} = \max(y_{ij} - v_i, 0)$. If x is a fixpoint of G, then $u(x)_{ij} = v_i$ for all $x_{ij} > 0$ and $u(x)_{ij} \leq v_i$ for all $x_{ij} = 0$, hence x is a NE. Conversely, it is easy to see that a NE x is a fixpoint of G. In the 2-player case, $u(x)$, and hence also $h(x)$, is clearly linear. It is not hard to see that π is piecewise linear (for any number of players); see the proof sketch of Theorem 11. Thus G is piecewise linear for 2 players. The same holds for polymatrix games (multiplayer games where the payoffs are sums of the payoffs of the pairwise interactions). ■

Theorem 9 *The exact fixed point computation problem for polynomial piecewise linear functions is in PPAD.*

Sketch. We use one application of Scarf's algorithm and Linear Programming. Given instance I, we pick a suitably small ϵ, compute a weak ϵ-fixpoint y, form a LP involving the inequalities on the path traced by the decision algorithm on y, and from the function at y, and show that the solution to the LP gives us an exact fixpoint. ■

Corollary 10 *The following problems are in PPAD: 1. Compute the value (and optimal strategies) of a simple stochastic game. 2. Compute an exact fixed point of a linearly interpolated function.*

Membership of simple stochastic games in PPAD was observed in [26], although it should be mentioned that there is an oversight in the proof there, because it uses a theorem of [35] without noticing that the theorem does not apply to the actual function, but to a modified function obtained by linear interpolation at grid points.

It follows from Theorem 9 and the examples 2 and 3 that the class of piecewise linear fixpoint problems is equivalent to PPAD. The same can be shown for the (piecewise) 'linear' part of FIXP: Let **Linear-FIXP** be the class of problems that can be expressed as (polynomially reduced to) exact fixpoint computation problems for functions given by algebraic circuits using $\{+, -, \max, \min\}$ (equivalently, $\{+,\max\}$) and multiplication with rational constants only; no division or multiplications of two gates/inputs is allowed.

Theorem 11 *Linear-FIXP=PPAD.*

Sketch. In one direction, it is not hard to show that a circuit as above that does not use multiplication (and division) except by a constant, computes a polynomial piecewise linear function, hence the exact fixed point problem is in PPAD by Theorem 9. In the other direction, we implement by a circuit C the function G of example 3 for a 2-player game Γ. For input x, the circuit C first computes $y = h(x)$ (using + and *-constant gates). For each player $i = 1, 2$, we have a (polynomial) sorting network N_i that sorts in decreasing order the corresponding subvector $y_{i.}$ of y to a permuted vector $z_{i.}$, using max, min gates for the comparators. Then compute v_i as $\max\{(1/l) * ((\sum_{j=1}^{l} z_{ij}) - 1) | l = 1, \cdots, |S_i|\}$; output $x'_{ij} = \max(y_{ij} - v_i, 0)$ for each $i = 1, 2; j \in S_i$. ■

From the FIXP-completeness of (3-player) Nash, using the function G and the above construction, we can show that FIXP does not change if we omit division: $\{+, *, \max\}$ and rational constants suffice.

Stochastic games were first introduced by Shapley in his seminal paper [39] in a more general form where the players take actions simultaneously instead of one at a time (as in SSGs). In **Shapley's game** there is a (finite) graph $G = (V, E)$, each node (state) u has an associated one-shot zero-sum finite game with a reward (payoff) matrix A_u for player 1 from player 2. If the play is in state u and the players choose actions (pure strategies) i, j then Player 1 receives reward

$A_u[i,j]$ from Player 2, the game stops with probability $q^u_{ij} > 0$, and it transitions to state v with probability p^{uv}_{ij}, where $q^u_{ij} + \sum_v p^{uv}_{ij} = 1$. Since there is at least positive probability $q = \min\{q^u_{ij} | u, i, j\} > 0$ of stopping in each step, the game stops a.s. in a finite number of steps. (Another standard equivalent formulation is as a discounted game, where the game does not stop but future rewards are discounted by a factor $1 - q$ per step). The goal of Player 1 is to maximize (and of Player 2 to minimize) the total expected reward, which is the value of the game. Let $x = (x_u | u \in V)$ be the vector of game values for the different starting states u. As usual we assume that all rewards and probabilities are rationals. The values in general may be irrational now however, so we may not be able to compute them exactly. We would like to bound them (i.e., answer decision questions) and/or approximate them.

The vector x of values satisfies a fixed point equation as follows. For each node u, let $B_u(x)$ be the matrix whose i, j entry is $A_u[i,j] + \sum_v p^{uv}_{ij} x_v$. Let $Val(B_u(x))$ be the value of the one-shot zero-sum game represented by the reward matrix $B_u(x)$. The optimal value vector satisfies the system of equations $x = F(x)$ where $F_u(x) = Val(B_u(x))$, $u \in V$. Furthermore, there is a unique solution, because F is a Banach function (a contraction map) under the l_∞ norm with contraction factor $1 - q$. Note that F (the Val function) is nonlinear. For the domain we can pick a box $[-M, M]^n$ with M large enough ($M = \max\{|A_u[i,j]| | u, i, j\}/q$ suffices).

Theorem 12 *The following hold for Shapley's game.*
1. Computing the values is in FIXP; hence Shapley reduces to (3-player) Nash.
2. The strong approximation problem is in PPAD.
3. The decision problem is at least as hard as `SQRT-SUM`.

Sketch. For part 1, we cannot construct directly an algebraic circuit for Val (we do not even know if LP can be solved in a number of operations that is polynomial in the dimension of the problem). We introduce instead additional variables for the probabilities of the strategies and corresponding fixed point equations. Part 2 is proved by noting that the function F is polynomially contracting and polynomially computable, hence strong approximation reduces to weak and is in PPAD. We note that the contraction principle can be used also to give an alternative proof for exact SSGs being in PPAD. Part 3 is shown by modification of a construction in [17] for a stochastic reachability game. ■

From (3), the decision problem for Shapley's game, i.e., "is the value $\geq r$ for given rational r?", is unlikely to be placed easily in PPAD.

5 RMCs, Branching Processes, and SCFGs

A Recursive Markov Chain (RMC) informally consists of a collection of Markov chains that can call each other in a potentially recursive manner like recursive procedures. RMCs generalize a number of classic probabilistic models. In particular, *Multi-Type Branching Processes* (MT-BPs) a fundamental probabilistic model ([29, 24]), correspond in a precise sense to 1-exit RMCs (1-RMCs), where each component has exactly one exit node where it can terminate and return control to the component that called it. Both these models also correspond to *Stochastic Context-Free Grammars* (SCFGs), a model in common use in Natural Language Processes ([32]), and biological sequence analysis ([13]).

A key to the analysis of all these models is the problem of computing the probability of termination (extinction); it can be expressed as the *Least Fixed Point* (LFP) of a corresponding monotone system of polynomials, $P : \mathbb{R}^n_{\geq 0} \mapsto \mathbb{R}^n_{\geq 0}$, i.e., each coordinate $P_i(x)$ is given by a polynomial with non-negative rational coefficients. The LFP, $x^* \in [0,1]^n$, is the solution to $x = P(x)$ such that for all solutions $x' \in \mathbb{R}^n_{\geq 0}$, $x^* \leq x'$. The LFP is guaranteed to exist for the polynomial maps that arise from these models, and gives precisely the vector of termination probabilities for them. Obviously, such polynomial functions can be described by algebraic circuits. We in fact show that the problem of computing/approximating termination probabilities for MT-BPs, SCFGs, and 1-RMCs, is in FIXP, by showing that the LFP for this class of RMCs can be isolated in such a way that it constitutes the unique fixed point of a corresponding FIXP function. Due to lack of space, we have to forgo formal definitions of these models. Background on these models and their algorithmic theory can be found in [16].

Theorem 13 *For every $\epsilon > 0$, the* `SQRT-SUM` *and* `PosSLP` *problems are P-time (many-one) reducible to the promised gap problem PGD(ϵ,1) for the termination probability of RMCs.*

Theorem 14 *Computing the termination probabilities for 1-RMCs, MT-BPs, and SCFGs, is in FIXP. Thus the decision/approximation problem reduces to decision/strong approximation for (3 player) NEs.*

Acknowledgement. It is a pleasure to thank Peter Bro Miltersen for an extended email discussion about PPAD and Nash Equilibria.

References

[1] E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. B. Miltersen. On the complexity of numerical analysis. In *21st IEEE Comp. Compl. Conf.*, 2006.

[2] R. M. Anderson. "Almost" implies "Near". *Trans. Am. Math. Soc.*, 296, pp. 229-237, 1986.

[3] R. A. Becker and S. K. Chakrabarti. Satisficing behavior, Brouwer's fixpoint theorem and Nash equilibrium. *Economic Theory*, 26:63–83, 2005.

[4] L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and Real Computation*. Springer-Verlag, 1998.

[5] V. Bubelis. On equilibria in finite games. *Intl. J. Game Theory*, 8(2), 65-79, 1979.

[6] J. Canny. Some algebraic and geometric computations in PSPACE. In *STOC*, pp. 460–467, 1988.

[7] X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. *FOCS*, pp. 261–272, 2006.

[8] X. Chen, X. Deng, and S. H. Teng. Computing Nash equilibria: approximation and smoothed complexity. In *FOCS*, pp. 603–612, 2006.

[9] A. Condon. The complexity of stochastic games. *Inf. & Comp.*, 96(2):203–224, 1992.

[10] C. Daskalakis, P. Goldberg, and C. Papadimitriou. The complexity of computing a Nash equilibrium. In *STOC*, pp. 71–78, 2006.

[11] C. Daskalakis, A. Fabrikant, and C. Papadimitriou. The game world is flat: The complexity of Nash equilibria in succinct games. *Proc. ICALP*, 2006.

[12] R. S. Datta. Universality of Nash equilibria. *Math. of Oper. Res.*, 28(3), 424-432, 2003.

[13] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic models of Proteins and Nucleic Acids*. Cambridge U. Press, 1999.

[14] B. C. Eaves, H. Scarf. The solution of systems of piecewise linear equations. *Math. of O.R.*, 1(1), 1-27, 1976.

[15] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. In *LICS*, 2004.

[16] K. Etessami and M. Yannakakis. Recursive Markov chains, stochastic grammars, and monotone systems of non-linear equations. In *STACS*, 2005. (See full expanded version from http://homepages.inf.ed.ac.uk/kousha/).

[17] K. Etessami and M. Yannakakis. Recursive Concurrent Stochastic Games. In *ICALP*, 2006.

[18] M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *8th ACM STOC*, pp. 10–22, 1976.

[19] J. Geanakoplos. Nash and Walras equilibrium via Brouwer. *Economic Theory*, 21:585–603, 2003.

[20] I. Gilboa and E. Zemel. Nash and correlated equilibria: some complexity considerations. *Games and Economic Behavior*, 1:80–93, 1989.

[21] P. Goldberg and C. Papadimitriou. Reducibility among equilibrium problems. In *STOC*, 2006.

[22] F. Gul, D. Pearce, E. Stacchetti. A bound on the proportion of pure strategy equilibria in generic games. *Math. of Oper. Res.* 18, pp. 548-552, 1993.

[23] P. Haccou, P. Jagers, and V. A. Vatutin. *Branching Processes: Variation, Growth, and Extinction of Populations.* Cambridge U. Press, 2005.

[24] T. E. Harris. *The Theory of Branching Processes.* Springer-Verlag, 1963.

[25] R. J. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge U. Press, 1985.

[26] B. Juba. On the hardness of simple stochastic games. Master's thesis, CMU, 2005.

[27] M. Kimmel and D. E. Axelrod. *Branching processes in biology.* Springer, 2002.

[28] K.-I. Ko. Computational complexity of fixed points and intersection points. *J. Complexity*, 11, pp. 265-292, 1995.

[29] A. N. Kolmogorov and B. A. Sevastyanov. The calculation of final probabilities for branching random processes. *Doklady*, 56:783–786, 1947. (Russian).

[30] R. Lipton and E. Markakis. Nash equilibria via polynomial equations. In *LATIN'04*, 2004.

[31] R.J. Lipton, E. Markakis, A. Mehta. Playing large games using simple strategies. *Proc. ACM Conf. Elec. Comm.*, 36-41, 2003.

[32] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[33] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54:289–295, 1951.

[34] C. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theor. Comp. Sci.*, 4:237–244, 1977.

[35] C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994.

[36] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals, parts I-III. *J. Symb. Comp.*, 13(3):255–352, 1992.

[37] H. Scarf. The approximation of fixed points of a continuous mapping. *SIAM J. Appl. Math.*, 15:1328–1343, 1967.

[38] H. Scarf. *The Computation of Economic Equilibria.* Yale University Press, 1973.

[39] L.S. Shapley. Stochastic games. *Proc. Nat. Acad. Sci.*, 39:1095–1100, 1953.

[40] P. Tiwari. A problem that is easier to solve on the unit-cost algebraic RAM. *Journal of Complexity*, pp. 393–397, 1992.

[41] H. Uzawa. Walras's existence theorem and Brouwer's fixed point theorem. *Econ. Stud. Q.*, 13:59–62, 1962.