Computation of Equilibria in Noncooperative Games

Salman Azhar *
College of Professional Studies, University of San Francisco, San Francisco, CA 94117
Andrew McLennan †
Department of Economics, University of Minnesota, Minneapolis, MN 55455
John H. Reif ‡
Department of Computer Science, Duke University, Durham, NC 27708

Abstract

This paper presents algorithms for finding equilibria of mixed strategy in multistage noncooperative games of incomplete information (like probabilistic blindfold chess, where at every opportunity a player can perform different moves with some probability). These algorithms accept input games in extensive form. Our main result is an algorithm for computing *Sequential equilibrium*, which is the most widely accepted notion of equilibrium (for mixed strategies of noncooperative probabilistic games) in mainstream economic game theory. Previously, there were no known algorithms for computing sequential equilibria strategies (except for the special case of single stage games).

The computational aspects of passage from a recursive presentation of a game to its extensive form are also discussed. For nontrivial inputs the concatenation of this procedure with the equilibrium computation is time intensive, but has low spatial requirements. Given a recursively represented game, with a position space bound $S(n)$ and a log space computable next move relation, we can compute an example mixed strategy satisfying the sequential equilibria condition, all in space bound $O(S(n)^2)$, Furthermore, in space $O(S(n)^3)$, we can compute the connected components of mixed strategies satisfying sequential equilibria.[1]

*Supported by DARPA/ARO Contract DAAL03-88-K-0195, DARPA/ISTO Contract N00014-88-K-0458, Office of Naval Research under contract N00014-87-K-0310, and Air Force contract AFSOR-87-0386. Also Supported by Duke University's James B. Duke Fellowship for Advanced Studies.

†Supported by NSF Grants MCS-8120790 and SES-8420114.

‡Supported by DARPA/ARO Contract DAAL03-88-K-0195, DARPA/ISTO Contract N00014-88-K-0458, Office of Naval Research under contract N00014-87-K-0310, and Air Force contract AFSOR-87-0386.

[1]Existence of sequential equilibria is guaranteed by Proposition 1 of Kreps and Wilson [1].

Preprint of paper appearing in Computers & Mathematics with Applications, (2005).

1 Introduction

1.1 The Equilibrium Problem in Classical Game Theory

In recent years, there has been a proliferation of applications of noncooperative game theory to economics, political science, evolutionary biology, and other disciplines. A typical research project consists of modeling an environment or strategic interaction as a formally laid out game, after which conclusions are derived from the application of a solution concept that (hopefully) embodies the relevant consequences of (simultaneously) rational or optimizing behavior by all agents. Even for quite simple games, the computation of a solution concept by hand can be an arduous endeavor. In this paper, we present algorithms for the computation of some of the most important solution concepts of noncooperative game theory, and we derive complexity measures for these algorithms.

The algorithms are based on the observation that, for the type of finite game studied here, the solution concepts of interest are described by finite systems of (real) polynomial equations and inequalities. In general, a solution set of a finite system of polynomial equations and inequalities is called a *semi-algebraic set*, and in recent years algorithms have been developed that pass from a given system of equations and inequalities to more direct and useful descriptions of the set determined by the system. For computer scientists, therefore, the main novelty in this paper is the elaboration of a new domain of application for these tools.

For pure and applied game theorists it is of interest to know that such algorithms are possible, of course, but it is also important to have a more detailed understanding of how they are constructed, and finally one should have some sense of their limitations. For the latter issue, the best available tools are the concepts of computational complexity studied in computer science. Our brief introduction to this subject is hopefully sufficient to enable the reader to understand the specific conclusions stated later.

Computer science has historically studied games from two perspectives. First, particular games present decision problems whose formal complexity can be studied. For example, one can consider the computational complexity of determining an optimal move in the game of *Go* as a function of the size of the board. Second, games can be used as models of particular computational problems or environments, particularly those involving parallel computation or networks of independent processors. Our work contributes to these traditions, so we will briefly review closely related literature.

As can be seen from the above description, this project is interdisciplinary, and the bulk of our work is expositional. Our aim is to provide each subaudience with enough information to appreciate the concepts and results of the unfamiliar discipline. The remainder of this introduction begins with a brief description of the game theoretic concepts studied here, followed by a quite cursory survey of the fundamental concepts of the theory of computation, intended for quite naive readers. For a more extensive introduction we recommend Hopcroft and Ullman [2] or Lewis and Papadimitriou [3]. We then briefly survey preexisting work on decision algorithms for games, first for pure strategies, then for probabilistic solution concepts. Finally we outline the work of subsequent sections.

1.2 An Outline of Our Results

Three methods of presenting a game are considered here: Recursive form, Extensive form, and Normal Form.

Perhaps most familiar in recreational games is the recursive presentation. There is a space of possible positions, with a designated initial position. There are rules (i.e. computational procedures) for passing from a nonterminal position to the player whose turn it is to move, the informational constraints faced by that player at the position, the set of allowed moves, and the new positions resulting from each allowed move. There are also rules determining payoffs at each terminal position. Both theoretically and practically, it is natural to require that these rules be computationally simple, as is the case, for instance, in chess.

The extensive form of a game is the most popular in modeling applications. Here the set of all legal paths through the space of positions is laid out in a tree. The informational constraints imposed on the players' choices are represented by grouping various 'nodes' of the tree into information sets; the interpretation being that whenever any node in an information set occurs a player must choose a move without knowing which particular node within the information set has occurred. Consideration of the example of chess show that the passage from a recursive presentation to the extensive form may be an expensive computation independent of any considerations of skillful play.

Given a game in its extensive form, each player has a space of 'pure strategies,' each of which is a vector specifying a legal action at each information set at which the player selects a move. One can imagine each player submitting a pure strategy to a referee, after which the referee plays out the game according to these instructions. (In the standard interpretation of a game, it is assumed that all possibilities for communicating to achieve coordinated behavior are already explicitly represented in the structure of the game.) Thus, we arrive at the third mode of description of games, the 'normal form': each player $i = 1, ..., n$ has a finite set S_i of pure strategies and a payoff function $u_i : S_1 \times ... \times S_n \to \Re$. The normal form is theoretically attractive insofar as the notation is simple, but it should be noted that a relatively simple extensive game can have a 'large' normal form, so that for any particular game the extensive form may be simpler.

What constitutes rational behavior while playing a game against rational opponents? For games of 'perfect information' (that is, whenever a player moves he or she knows the exact state of the game) such as chess, this is in principle a simple question. One can work backwards through the tree, at each point determining the moves that are best on the assumption that all subsequent choices will be optimal. When the game is not one of perfect information the problem becomes more difficult, both conceptually and practically. To begin with, it is not sufficient to consider only pure strategies; a player who was known to always play a particular pure strategy in the game 'rock-paper-scissors' could always be beaten. Thus we are led to consider probability distributions over pure strategies; these are called *mixed strategies.*

The most famous and central solution concept in noncooperative game theory is the notion of Nash equilibrium. A vector of mixed strategies, one for each agent, is a *Nash equilibrium* if no agent has any other mixed strategy that yields a higher expected payoff when his expectations concerning the behavior of the other agents are given by the equilibrium strategies.

The mixed strategies in a Nash equilibrium have a dual interpretation: they represent both what the agents expect of others and what the agents actually do, at least in a statistical sense. This raises serious conceptual questions concerning the relevance of Nash equilibrium, since it need not be the appropriate embodiment of rationality when the game is played more than once by the same agents, but at the same time some learning process seems to be required in order for the agents to form expectations, particularly (but not exclusively) when the game has more than one Nash equilibrium. We will not dwell on this point, except to say that it is a valid question in the context of each application of the theory.

A more germane conceptual difficulty with Nash equilibrium is the fact that it fails to capture

	Player A		
		acquiesce	resist
Player B	fight	0,0	-1,-1
	retreat	0,0	1,1

Table 1: Illustration of Nash Equilibrium

the assumption that all agents will continue to behave rationally at any stage of play. Consider the game illustrated in Table 1. Player A must decide whether to acquiesce or resist, and Player B must choose whether to fight or retreat. (Player B's choice is relevant only when A resists.) Assume that Player A's favorite outcome is (resist, retreat), and (resist, fight) is her least favorite outcome, while Player B prefers (resist, retreat) to (resist, fight). Then (acquiesce, fight) is a Nash equilibrium: given that Player B intends to fight, Player A does best to acquiesce; given that Player A always acquiesces, Player B's strategy has no effect on the outcome. However, Player A's behavior is rational only in response to a "threat" that could not rationally be carried out, in that Player A expects Player B to play a strategy that is *weakly dominated*: Player B never does better by fighting, and, in response to some strategies of Player A, fighting is worse. Consequently, this equilibrium is not plausible.

Historically the most influential attempt to address this problem is the concept of *Perfect Equilibrium* introduced by Selten [4]. A formal definition of this concept is given in § 3.2. Here it suffices to say that Selten's terminology is now regarded as excessively optimistic, and many solution concepts have subsequently been proposed, but the central philosophical issues are not yet resolved to the point of consensus.

Virtually all known solution concepts are semi-algebraic, so that they can be computed by means of the algorithms discussed here. From the point of view of computational tractability it is useful to distinguish between quantified solution concepts, in the sense that the definition involves the logical operators "for all" and "there exists," and unquantified concepts. (Perfect equilibrium is a quantified concept, Nash equilibrium is unquantified.) As proven by Renegar [5] (see also Renegar's related papers [6, 7, 8]) any quantified proposition is equivalent to an unquantified proposition, and the passage from the quantified to the unquantified version can be implemented on a computer, but current algorithms for doing so are slow, so that unquantified solution concepts are more likely to be practical from a computational viewpoint.

Currently, most popular among researchers in economics is the concept of *Sequential Equilibrium* due to Kreps and Wilson [1]. We will devote special attention to this concept, in part because of its popularity but also because the semi-algebraic nature of this concept is far from trivial. In fact we will show that it has an unquantified definition. This concept is applied to the extensive form, and it has the interesting feature that it involves not only strategies but also 'beliefs,' i.e. probability distributions over the nodes in each information set that are construed as the conditional distributions the agents would attribute to the nodes, when and if the information set occurred. In applications, these beliefs are both intuitively interesting and quite helpful in computing sequential equilibria by hand. It seems reasonable to hope that these beliefs might also facilitate the development of 'speed-ups' of the formal algorithms described here.

Given any solution concept, we can apply algorithms of Canny [9] and Renegar [10] for deciding the existential theory of real closed fields. These algorithms determine whether the solution set of the system is nonempty (which is actually not of great interest to us since all the solution concepts considered here have been proven to have nonempty solution sets for all possible parameters) and find sample solutions, all in space polynomial in the size of the information set for the game's normal form. Using the methods of Kozen and Yap [11], we can find a decomposition of the strategy space (or, in the case of sequential equilibrium, strategy-belief space) into simple pieces, each of which is either contained in the solution set or disjoint from it. Bounds on the space and time requirements of these algorithms imply bounds on the space and time requirements of the particular applications considered here.

1.3 Fundamentals of Computational Complexity

We devote this subsection to a concise overview of pertinent concerns in theoretical computer science. The following two issues are especially important for our work:

Computability and Noncomputability : Ascertain whether there exists a computational solution to a given problem.

Design and Analysis of Algorithms : Specify a set of instructions to solve a problem, provide measures of the 'costs' of the algorithm in time and computational resources (such as memory), and whenever possible prove that the algorithm is optimal in the sense that no other algorithm could possible solve the problem at a lower 'cost' (usually up to a constant factor).

1.3.1 Computability and Noncomputability

It is obvious that many problems have computational solutions. An example is the determination of an optimal move in a tic-tac-toe game. However, some problems do not permit an algorithmic solution. The *Halting Problem* is the most notorious of all these problems: ascertain whether an arbitrary algorithm will eventually halt (as opposed to continue forever) while working on a given input. The proof that no algorithm for this problem exists is by contradiction, and is obtained by considering how such an algorithm would behave when it was fed itself as input. Details can be found in any standard text on theory of computation like Lewis and Papadimitriou [3].

Another interesting and more relevant example of a problem which does not admit an algorithmic solution is the following:

> Compute a point whose ϵ-ball contains a fixed point of a given continuous function from the unit disk to itself.

Superficially this seems at odds with our goals because solution concepts in game theory are typically described as sets of fixed points. Fortunately, however, there is no real paradox because our algorithms exploit the fact that the problem is presented algebraically, so that a general algorithm for fixed points is not required.

We must formally define computational models to facilitate rigorous discussion of computability issues. The most famous and best studied model of computation is the Turing machine, which consists of a *processor* and a *storage tape*. In terms of current computational technology one would think of the processor as consisting of a *Central Processing Unit (CPU)* together with the *Random Access Memory (RAM)*. These days, the tape usually is a magnetic memory media along with input-output devices. In the abstract theory the relevant facts are that the processor has a finite set of internal states, one of which is designated as the initial state, and the tape is a doubly infinite and one dimensional recording medium on which 0's and 1's are written.[2] A computational cycle consists of reading the character on the space of the tape that is currently in the 'tape reader,' then combining this character with the current state of the processor to generate a new state and instructions for (possibly) changing the character in the space of the tape that has just been read, (possibly) moving the tape one space in either direction, and (possibly) declaring that the computation is complete. In mathematical terms a Turing machine is described by specifying finite sets of states and characters, an initial state, and a transition function δ:

$$\delta : \{states\} \times \{characters\} \mapsto \{states\} \times \{characters\} \times \{tape\ motion\} \times \{halt, continue\}$$

Given the similarities between Turing machines and actual computers, it is hardly surprising that Turing machines are capable of many computations. *Church's thesis* asserts that the behavior of any computational device can be mimiced by a Turing machine. This is a metamathematical proposition which can never be conclusively proved. However, it can be shown that some particular model of computation is equivalent to the Turing model, and since all models of computation proposed to date have this property, Church's thesis is accepted as a basis for theoretical work.

Roughly we think of the Turing machine as the algorithm, and the initial state of the tape as the input. This interpretation is somewhat at odds with current technology in which most computers are 'universal devices.' That is, they accept 'tapes' that specify both an algorithm and an input to which the algorithm is to be applied. The theoretical analogue is the notion of a universal Turing machine, i.e. a Turing machine capable of mimicing the behavior of any other Turing machine. Universal Turing machines exist and have interesting theoretical applications.

1.3.2 Design and Analysis of Algorithms

Once we are satisfied that an algorithmic solution exists for a given problem, we can investigate particular algorithms to solve those problems. The fundamental concerns are:

Correctness : Prove correctness after developing a rigorous functional description of the problem. Some generalized methods include Program Verification [12, 13, 14, 15, 16, 17, 18, 19, 20] and Program Checkers [21]. In reality, more specific and practical methods are employed [22, 23, 17].

Complexity : Ascertain the resources demanded by an algorithm. The most important resources are time consumed for the algorithm to terminate and memory space required for the algorithm to execute.

Optimality and Practicality : Compare the given algorithms with the theoretically best possible algorithm (even if only the existence of the 'best algorithm' has been proved without an explicit description of the algorithm).

Verifying the correctness of an algorithm is a topic in itself, and is tangential to the theme of this paper.

In order to address the complexity issue for any particular algorithm, a computer scientist needs a *measure* to determine how expensive computations performed by the algorithm are or, more roughly, whether the algorithm is 'practical' and 'effective'. The most important resources

[2] Other alphabets are possible, usually theoretically equivalent, and therefore convenient for some problems. The model can also be tailored by modifying size and structure of the recording medium (tape).

are the *time* required for an algorithm to execute and the *space*, in the sense of memory, required for storage of intermediate calculations. We will briefly discuss that the most useful measures of expense are rather crude for a number of reasons.

First, the expense of applying the algorithm to any particular input is simply a number which is highly sensitive to the particular device chosen within the class of Turing equivalent devices. For example, if we consider the running time complexity of an algorithm, there would be considerable disparity between the experimental values we would get using a CRAY YMP and what we would get using a Commodore C-64. It is more interesting (theoretically) to have some sense of how fast the expense of computation grows as some measure of the size of the input increases. Generally, computer scientists tend to regard an algorithm as 'efficient' if its running time is (up to order-of-magnitude) a polynomial function of the size of the input, and 'inefficient' if its expense grows exponentially (or worse) with the size of the input. Roughly speaking, these order of magnitude measures are independent of which Turing-equivalent devices is used as the basis of the calculation, and they are also independent of the different ways (e.g. roman character strings versus binary strings) that the size of the input can be measured.

Second, measures of expense are typically based on an assumption that only one computational resource is costly. By far the best studied notion of expense is running time. However, recently there has been work on space (i.e. memory) requirements of algorithms. To see what is meant by this, imagine that our model of computation is enhanced by adding a second tape to the Turing machine, so that one now has a 'working tape' for storing the results of intermediate computations in addition to a 'read-only input tape'. The spatial requirements of algorithms are compared by studying the required length of the working tape as a function of the length of the input tape (up to order-of-magnitude). The economists will be quick to point out that, in practice, there is a tradeoff between time and space: the results of intermediate computations can be stored (space intensive) or recomputed when needed (time intensive). Unfortunately, current theoretical tools are by and large too crude to illuminate this tradeoff, except in the dynamic programming paradigm[3] and special cases where a problem is well-understood (e.g. sorting a list of entries). Moreover, other important computational resources, in particular programming effort, are difficult to model, and are consequently not well treated by existing theory.

Third, the most tractable measures of cost are worst-case, giving the time or space requirements for the most expensive input of given size. It is important to recognize that this can give an excessively pessimistic view of the utility of an algorithm. For instance it has long been known that the *Simplex Algorithm* for linear programming problems has a poor worst-case performance, but in practical experience the *Simplex Algorithm* usually halts quite quickly. Recently, Smale have given a theoretical explanation, showing that for 'randomly generated' problems, the probability of generating a 'bad' problem diminishes quickly as the size of the problem increases. The basic algorithms of interest to us, those which compute the structure of semi-algebraic sets, have worst-case running times that are quite bad, but typical running times may be much better, as suggested by the work of Arnon and Mignotte [24].

We can now explain the style of terminology used to describe upper bounds on computational costs. We say that a function $g(n)$ is $O(f(n))$ if there exists a constant c such that $g(n) \leq c \cdot f(n)$ for all n. Saying that a problem is 'log space computable' means that there is an algorithm for solving the problem such that if $S(n)$ is the length of working tape required in the worst case to process

[3]Dynamic Programming views a algorithmic solution as a sequence of decisions. The technique involves splitting the main problem into smaller subproblems that are solved independently, and the corresponding subsolution is recombined to yield a solution to the main problem. Since Dynamic Programming techniques involve storing the repeated subsolutions, we achieve a faster solution at cost of increased memory requirements.

an input of length n, then $S(n) \leq O(\log n)$. Similarly, an algorithm runs in 'polylog time' if there is a polynomial P of one variable such that the running time for an input of size n is $O(P(\log n))$. A problem is 'polylog computable' if such an algorithm exists. Among the really bad problems are those whose space or time requirement are towers of exponentials, i.e. functions of the form $c2^{2^{P(n)}}$, $c2^{2^{2^{P(n)}}}$, etc., where c is a constant and P is a polynomial. This is due to the fact that the space requirement of such problems grows very fast (exponentially) as the size of the problem increases.

We say that $2^{2^{P(n)}}$ is a '*tower of two repeated exponentials*', $2^{2^{2^{P(n)}}}$ is a '*tower of three repeated exponentials*', etc. We use the notation $EXP_m(P(n))$ to denote a '*tower of m repeated exponentials*'. The following definitions formalize the concept of bounds.

Definition 1.3.1 (Time Bound) *An algorithm has time bound $T(n)$ on machine $\mathcal{M}$, if, when applied on machine $\mathcal{M}$ to any input string ω of length n, the algorithm terminates in time $T(n)$.*

Definition 1.3.2 (Space Bound) *An algorithm has space bound $S(n)$ on machine $\mathcal{M}$, if, when applied on machine $\mathcal{M}$ to any input string ω of length n, the algorithm does not consume more than $S(n)$ non-blank memory cells.*

For recursively presented games it is useful to have the following piece of terminology.

Definition 1.3.3 (Game's Position Space Bound) *We define the position space bound of a game to be the upper bound on the number of bits required to encode any position.*

Definition 1.3.4 (Algorithm's Position Space Bound) *An algorithm has space bound $PS(n)$ on machine $\mathcal{M}$, if, when applied on machine $\mathcal{M}$ to any input string ω of length n, it does not use more than $PS(n)$ memory cells to encode any position it traverses.*

1.4 Previous Work in Decision Algorithms for Pure Strategies

Games can be viewed as simple models of computational problems. Interpreting a game of perfect information whose only outcomes are 'Win' and 'Lose' in this way, the most fundamental question is the 'Outcome problem', which is the problem of determining if a given player (team) has a winning strategy over opposing players (teams). The Outcome problem is closely related to the membership question of language and machines, which is the problem of determining if a given string occurs in a language. A string in a language corresponds to a perfect information game, whereas the language corresponds to a game, and a class of languages corresponds to a class of games.

The definition of a Turing machine facilitated the development of computability theory by formalizing algorithmic procedures. Similarly, several other paradigms of computation (non-determinism, parallel, etc.) were associated with corresponding models of computations (non-deterministic Turing machine, parallel random access machine, etc.). The need for a formal computational model to address the computational aspects of games was fulfilled by Chandra, Kozen, and Stockmeyer [25, 26] with the *Alternating Turing Machine* (**A-TM**). Subsequently, this model has been extended and enhanced to model more intricate games. Reif [27] extended the **A-TM** model to incorporate private and blindfold two-player games by introducing private alternating Turing machines (**PA-TM**) and blind alternating Turing machines (**BA-TM**), respectively. Azhar, Peterson and Reif [28, 29] introduce private alternating Turing machines (**PA**$_k$**-TM**) and blind alternating Turing machines (**BA**$_k$**-TM**) to model private and blindfold multiplayer games, respectively.

Mode of COMPUTATION	Type of GAME
Deterministic	Solitaire, Perfect Information, Unique next move
Non-deterministic	Solitaire, Perfect Information, Open next move
Alternation	Two-Player, Perfect Information
Private Alternation	Two-Player, Incomplete Information
Multiperson Alternation	Multi-Player, Incomplete Information

Table 2: Comparing Computation and Games

All these new types of machines have provided a deeper insight into the relationships between time and space bounded computation. Different types of games correspond to different models of computation as shown in the Table 2. In particular, it is fascinating that the simplest type of game (Solitaire, Perfect Information, Unique Next Move) corresponds to our most natural notion of computation (deterministic). On the other hand, the most interesting type of game (Multi-player, Incomplete Information) corresponds to a novel and abstract notion of computation.

As treated in computer science, a normal form game is typically given as an input. However, we can actually start with any position-space-bounded game in recursive form, where the next move relationship is computable in log space. By the undecidability results of Azhar, Peterson, and Reif [29], there is no recursive procedure to determine whether a game necessarily reaches a terminal position, and to compute its normal form, if the game has more than two players, unless the game is hierarchical.[4] [5] However, if the game is hierarchical, we can utilize the technique for unraveling information of Azhar, Peterson, and Reif [28, 27]. Given a recursively represented hierarchical game with log space computable next move relation, we can transform it into an equivalent game G^*, with space bound which is a tower of $k-1$ repeated exponentials in original space bound $S(n)$, where k is the number of cliques (each clique is defined to be the maximal set of players with exactly the same rights to view components of the game). Note that k is no greater than the number of players.

1.5 Previous Computer Scientific Work on Probabilistic Games

Strategies for playing games can be classified into two flavors: deterministic and probabilistic. Deterministic strategies involve specifying exactly one alternative at each position: such strategies are known as 'pure' strategies. Non-probabilistic games follow a set course of play once the participating players have formulated their strategies. On the other hand, probabilistic strategies assign probabilities to various alternatives available at each position: such strategies are known as 'behavior' strategies.

Papadimitriou [30, 31] describes the notion of 'Games Against Nature' (not to be confused with game theorists' use of this term to denote games with one player). In these games, one player plays randomly simulating the randomness we associate with nature, and the other "existential" player selects a pure strategy which maximizes the probability of success against this random player. In this framework, the existential player is considered to have won the game if he can win with a probability greater than $\frac{1}{2}$. *Games Against Nature* paradigms assist in formulation of decision problems

[4] Hierarchical multiplayer games are multiplayer games in which the information is hierarchically arranged, i.e. players can be arranged $\{1, 2, 3, ...\}$ such that all information visible to player i is also visible to player $i-1$.

[5] Note the similarity to the halting problem.

under uncertainty. These games are similar to *Arthur-Merlin Games* of Babai [32], in which Arthur plays randomly, and Merlin plays existentially. *Interactive Proof Systems* of Goldwasser et al. [33] are also among examples of games in which one player plays randomly whereas the other existentially picks a strategy. Sipser and Goldwasser [34] have proved the equivalence of Interactive Proof Systems and Arthur-Merlin games. Shamir [35] proves both problems are in the same complexity class (PSPACE-complete). Condon and Ladner [36] investigated the complexity of probabilistic game automata. Blum, Shub, and Smale [37] developed a complexity theory for computations on real numbers, but this work did not have any game theoretic component.

1.6 Organization of This Paper

§ 1 has provided an introduction to this paper along with a concise overview of the main issues and results. § 2 defines our notation for games in recursive, extensive, and normal forms. It also introduces fundamental notions of Information Sets, Moves, Initial Assessment, Utility, and Strategy. § 3 is devoted to the development of the *Sequential Equilibrium* paradigm. In § 4, we develop an algebraic characterization of *Sequential Equilibrium*, and associated computational issues. The main results and conclusions are sketched in § 5.

2 Fundamentals of Noncooperative Game Theory

In this section, we precisely define three models of noncooperative games. We bear an unusual expositional burden because we desire this paper to be accessible to researchers in computer science as well as economics, and we should explain at the outset that the first model is most closely related to work in computer science, while the second two models are standard in economics.

2.1 Recursive Form

A *game in recursive form* is a set of rules specifying:

1. A set of positions.
2. A set of players (also known as agents).
3. A rule specifying the player whose turn it is to move at any position.
4. A specification of the knowledge a player has available at his or her turn to move. Different descriptions of this knowledge are possible. For example, the position might be a string of characters with each player observing some positions in the string and not others. However, any such description should be equivalent to the following: the positions at which a player moves are partitioned into information sets, the interpretation being that the player knows only that some position in the information set has occurred.
5. A set of legal next moves from any given information set or 'state of knowledge' for a player.
6. A rule specifying the subsequent position that results from any position when any legal move is chosen.
7. A rule specifying an initial position at which the game starts.

8. A rule specifying when the game terminates.
9. A vector of payoffs (real numbers awarded to each player at the termination of the game) associated with each possible outcome.

As we will see shortly, the description above is not dramatically different from the extensive form presentation of a game. The crucial difference, not formally stated above, is a matter of computational complexity, in the sense that in the recursive presentation there are algorithmic formulations of the rules for determining (a) whether a position is terminal, (b) payoffs at terminal positions, (c) the player to move, the set of legal moves, the information available to the player to move, and the positions resulting from legal moves at a nonterminal position. In theoretical analyses, it is assumed that an execution of one of these algorithms is either trivial, so its expense can be ignored, or is an elementary computation from the point of view of a theory that measures complexity in terms of numbers of elementary computations. The entire game tree, however, is not presented as an input, nor is it necessarily the result of a small number of computations. In complexity analysis that takes the extensive form as the given object, on the other hand, it is assumed that the entire tree is explicitly laid out in the input, so that the size of the input is roughly proportional to the number of nodes in the tree.

The description above is more general than might be immediately apparent. Some important features allowed by our structure are as follows.

1. Players need not take turns in round-robin fashion. The rules of the game will dictate whose turn is next.
2. A player may not know how many turns were taken by other players between its turns.
3. Communication is possible, though only through the explicitly laid out structure. That is, some of the moves may serve as messages.
4. Positions, as we use the term, may include more information than is "on the board." Thus, in chess, the position includes castling rights and enough information to implement the three fold repetition rule and the fifty move rule, i.e. the history of play since the last pawn move, capture, or loss of castling right.

2.2 Game in Extensive Form

2.2.1 Basic Notation

Instead of representing the possible plays of a game indirectly, by specifying recursive procedures for generating them, the extensive form presents them directly, in terms of an explicitly specified game tree. For example, Figure 1 lays out a simple two-player guessing game, known in the literature as "matching pennies" in which one player hides an object, perhaps a white pawn, in one hand, (L)eft or (R)ight, and the other player guesses which hand, (l)eft or (r)ight, the object is in. (Chess players will recognize that this game is commonly used to assign colors at the beginning of a game.)

A game tree consists of a set of play prefix nodes with the the root node(s) represents the starting position(s) of the game. In Figure 1, $\{\bigcirc, L, R, Ll, Lr, Rl, Rr\}$ are the set of play prefix nodes. Each node represents a position, and its children are the positions after the next move. Every node is connected to its children with branches labeled with each of the alternative moves that can be chosen by the player whose turn it is to move. For example, when player 2 moves to

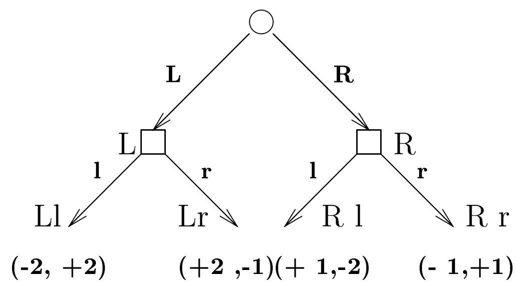

Figure 1: A Simple Guessing Game

l in response to L, we use an arc labeled l to connect L with Ll. It is important to note here that two equivalent situations in a game which occur at different stages of the game (by transposition of moves or otherwise) are considered distinct, and they correspond to different nodes in the game tree. In general, it is possible that the identity of the player who is to move next is determined by the situation of the game. A game represented by its game tree is said to be represented in its extensive form. We dedicate the rest of this section to develop the notation for games in extensive form.

Notation 2.2.1 (Multiperson Game) *:*

A multiplayer game can be defined as follows:

I is a finite set of players.

T : There is a finite set T of possible states of the game. Generic elements are denoted by π, π', etc. In Figure 1, $T = \{\bigcirc, L, R, Ll, Lr, Rl, Rr\}$.

(T,$\vdash$) : $\pi \vdash \pi'$ denotes that there is a legal next move from π to π' (i.e. we can transform π to π' in a single move). Each node of the game tree denotes a possible play situation, say π, of the game, and its children are all plays π' such that there is a legal next move relation from parent to child. (T,$\vdash$) forms an arborescence[6].

(T,$\models$) : $\pi \models \pi'$ if and only if there is a sequence of legal moves which can transform π to π' (i.e. there is a sequence $\pi_1, \pi_2, ...$ such that $\pi \vdash \pi_1 \vdash \pi_2 \vdash \cdots \vdash \pi'$). A particular subscript can be specified (like $\models_n$) to restrict $\models$ to a sequence of exactly n legal moves. Since (T,$\vdash$) is an arborescence, $\models$ is a strict partial ordering.

W,Y : *Initial Node(s)*[7] are nodes without predecessors. The set of initial nodes is denoted by W, and the non-initial nodes are denoted by Y. Y is defined to be the set difference $T - W$. In Figure 1, $W = \{\bigcirc\}$ and $Y = \{L, R, Ll, Lr, Rl, Rr\}$. In case there are several distinct initial positions (as in most card games), we also need to specify the probability distribution over the set of initial nodes. This distribution is called the *Initial Assessment*. In general, we will assume that every initial node in the initial assessment has positive probability.

[6]This means that each play has at most one predecessor, and any sequence of immediate predecessors must terminate rather than cycle. Arborescences are called *forests* in computer science.

[7]known as *root(s)* in computer science terminology

Z,X : A *terminal node*[8] is a node in the game tree which does not have any successors. Z denotes the set of terminal nodes, and $X = T - Z$ is the set of nonterminal nodes. In Figure 1, $Z = \{Ll, Lr, Rl, Rr\}$, and $X = \{\bigcirc, L, R\}$. Each terminal node is associated with a *vector of payoffs* specifying the gain and loss experienced by each participant at the end of the game. The payoffs are normally interpreted as von Neumann-Morgenstern utilities as specified by von Neumann and Morgenstern [38]. In Figure 1, we use ordered pairs to denote the payoffs to the players 1 and 2, respectively.

$\iota(\pi)$: The function $\iota : X \mapsto I$ defines whose turn is to move next at the nonterminal nodes.

$\ell(\pi)$: The **level** of $\pi \in T$ is the integer $\ell(\pi)$ which represents the number of plays preceding π (starting from some initial node).

p : There is a function $p_1 : Y \mapsto X$, such that $p_1(\pi) = \pi'$ if $\pi' \vdash \pi$ (i.e. $\pi' \models \pi$ in one move). For $n > 1$, we recursively define $p_n(\pi) = p_{n-1}(p_1(\pi))$ (or equivalently $p_n(\pi) = \pi'$ if $\pi' \models_n \pi$). $p(\pi)$ is defined to be the set of all predecessors of π: $p(\pi) = \{\pi' \mid \pi' \models \pi\}$. So, recursively $p(\pi) = \{p_1(\pi)\} \cup \{p(p_1(\pi)\}$ (or iteratively $\cup_{k=1,..,\ell(\pi)} p_k(\pi)$).

$\omega(\pi)$: The root of π is the initial position from which π is derived:

$$\omega(\pi) = p_{\ell(\pi)}(\pi) \in W.$$

$F(\pi)$: For $\pi \in T$ we let $F(\pi)$ be the set of *immediate successors* of $\pi \in X$, which are the plays reachable from π via one legal move. Note that $F(\pi) = p_1^{-1}(x)$.

ZOF(π) consists of the set of terminating plays derivable from π. $ZOF(\pi) = \{\pi' \in Z \mid$ if $\pi \in X$ then $\pi \models \pi'$ else $\pi' = \pi\}$.

A : There is a finite set A of *legal moves*.

α : There is a surjective function $\alpha : Y \mapsto A$ that labels each noninitial node with the last move chosen prior to the occurrence of the node. At every node, each allowed move has a unique consequence. Symbolically, if $p_1(\pi) = p_1(\pi')$ and $\pi \neq \pi'$, then $\alpha(\pi) \neq \alpha(\pi')$. The result of applying move $m \in \alpha(F(\pi))$ at π is denoted by $FC(\pi, m)$.

Table 3 summarizes the notation developed so far.

2.2.2 Information Sets

Definition 2.2.1 (Information Sets (H_i)) *: Information possessed by players is represented by a partition H of X into information sets. For each player i, there is a partition H_i of $\iota^{-1}(i)$. For $\pi \in \iota^{-1}(i)$, $H_i(\pi)$ is the cell of H_i containing π.*

We impose the following assumptions on this structure:

1. For all i, all $h \in H_i$, and all $\pi, \pi' \in h$, $\alpha(F(\pi)) = \alpha(F(\pi'))$: the alternatives available to the player are the same at π and π'.

[8] In computer scientific terminology a terminal node is better known as a *leaf*.

Notation	Name	Description
I	Finite set of players	Specified by rules of the game
T	Set of plays	$\{\pi \mid \pi$ is a legal play $\}$
$\vdash$	Next move relation	$\pi \vdash \pi'$ if there is a legal move from π to π'
$\models$	Derivable play relation	$\pi \models \pi'$ if there is a sequence of moves such that $\pi \vdash \pi_1 \vdash \pi_2 \vdash ... \vdash \pi$
W	Initial position	$\{\pi \in T — p(\pi) = \phi \}$
Y	Noninitial plays	T - $W = \{\pi \in T — p(\pi) \neq \phi \}$
Z	Terminal plays	$\{\pi \in T — F(\pi) = \phi \}$
X	Nonterminal plays	T - $Z = \{\pi \in T — F(\pi) \neq \phi \}$
$\iota(\pi)$	Player to move next	The next player to move
$\ell(\pi)$	Length of π	Number of positions in π in addition of the initial position
$p_1(\pi)$	Immediate predecessor of π	$\pi' \in T$ such that $\pi' \models \pi$
$p_m(\pi)$	mth predecessor of π	$p_1(p_{m-1}(\pi))$ for $m > 1$
$p(\pi)$	Predecessors of π	$\{ p_1(\pi) \} \cup \{ p(p_1(\pi)) \}$
$\omega(\pi)$	Initial predecessor of π	$p_\ell(\pi)(\pi)$
$F(\pi)$	Immediate successors of π	$\{ \pi' \mid \pi \models \pi' \}$
$ZOF(\pi)$	Terminal positions	$ZOF(\pi) \equiv \{\pi' \in Z — \pi \models \pi'\}$
A	Set of legal moves	$\{ m \mid \pi \vdash \pi'$ with move $m \}$
$\alpha(\pi)$	Move performed last	m such that $FC(p_1(\pi), m) = \pi$

Table 3: Notation and Definitions for Extensive Form

Notation	Name	Description
H_i	Information sets	Partitions of T into sets with equivalent knowledge for player i
$H_i(\pi)$	Info. set containing play π	$A \subseteq H_i$ such that $\forall \phi \in A : vis_i(\phi) = vis_i(\pi)$
$m(h)$	Moves available at $h \in H$	$\{\alpha(\phi) \mid \phi \in F(\pi), \pi \in h\}$
$\iota(h)$	Player choosing at $h \in H$	$\iota(\pi)$ for $\pi \in h$
$MOVSET_i(m)$	Info. set at which move m might be chosen	$MOVSET_i(m) = m^{-1}(m)$, i.e. $h \in H$ s.t. $m \in m(h)$.

Table 4: Capturing the Incomplete Information Content

2. For all $m \in A$ there is a unique $h \in H$ such that $m \in \alpha(F(\pi))$ for some (hence all) $\pi \in h$. Let $MOVSET(m)$ denote the information set at which move m can be chosen. Formally, $MOVSET(m)$ is $m^{-1}(m)$, i.e. the information set $h \in H$ such that $m \in m(h)$.

The first assumption is essential to the intended interpretation, namely that when a play in an information set occurs, the player to move knows only that *some* play in that information set has occurred, not which one. In contrast, the second assumption is a mathematical convenience rather than a substantive assumption, and is commonly violated in notational conventions for particular games: for example, 'N-KB3' is used to denote moves available in many different positions in chess.

For $i \in I$ we can define $H_i = \iota^{-1}(i)$ be the set of information sets at which player i chooses, and we let $m_i = \cup_{h \in \mathrm{H}_i} m(h)$ be the set of moves that could be chosen by i. For easy reference the notation above is summarized in Table 4.

Using the machinery of notation develop above, we now state the assumption of perfect recall.

Proposition 2.2.1 (Perfect Recall) *Each player knows what he or she knew previously (i.e. does not lose any knowledge). Consider an information set $h \in H_i$ at which player i moves and two plays $\pi, \pi' \in h$. Suppose that there is another information set $h_0 \in H_i$ and a play $\pi_0 \in h_0$ with $\pi_0 \models \pi$. Then at π player i should remember that h_0 occurred, and since player i cannot know whether π or π' occurred, this should also be the case at π'. That is, there should be a play $\pi_0' \in h_0$ such that $\pi_0' \models \pi'$. Moreover, the move chosen at π_0' on the way to π' should be the same as the move*

Notation	Name	Description
$\Delta(X)$	Probability measures on any finite set X	$\{\lambda : X \longrightarrow [0,1] \mid \sum_{x \in X} \lambda(x) = 1\}$
$\Delta^o(X)$	Interior probability measures on X	$\{\lambda \in \Delta(x) \mid \lambda(x) > 0 : \forall x \in X\}$
ρ	Initial assessment	$\rho \in \Delta^o(W)$
$u = \langle u_i \rangle_{i \in I}$	Utility/Payoff	u_i is the reward for player i (associated with every conclusion)

Table 5: Notation and Definitions for the Initial Assessment and Payoffs

chosen at π_0 *on the way to* π*: if* $\pi_0 = p_m(\pi)$ *and* $\pi_0^{'} = p_n(\pi')$*, then* $\alpha(p_{m-1}(\pi)) = \alpha(p_{n-1}(\pi'))$.

Among other things, perfect recall implies that if π and π' are in the same information set, then they are are not related by precedence. To see this suppose that $\pi' \models \pi$, and observe that perfect recall implies the existence of a node π'' in the same information set with $\pi'' \models \pi'$; proceeding inductively, we can generate an infinite sequence of nodes in the information set, all distinct since they are ordered by $\models$, contrary to the assumed finiteness of the tree.

In general, for any finite set X we let Δ(X) be the set of probability measures on X, and we let Δ^o(X) be the set of probability measures that assign a positive probability to all elements of X.

Definition 2.2.2 (Initial Assessment) *The initial assessment is a probability distribution on the initial position* ($\rho \in \Delta(W)$)*. For many purposes, though not all, an initial node that has zero probability of occurring can simply be eliminated from the game tree (along with its successors). Consequently, unless stated otherwise we will assume that* $\rho \in \Delta^o(W)$*. (We assume that every player's initial assessment is the same).*

Definition 2.2.3 (Utility) *For each player* $i \in I$*, the payoff function* $u_i : Z \mapsto \Re$ *assigns a real valued von Neumann-Morgenstern utility to each outcome. The payoff is a* $I \times Z$ *matrix where for each node* $\pi \in Z$*, there is an associated vector* $u = \langle u_i(\pi) \mid i \in I \rangle$*, where* u_i *is the utility/payoff to player* i*. For example, Figure 1 indicates the utilities as ordered pairs for each final outcome.*

We summarize these definitions in Table 5 for convenient reference.

Now, we have developed notation necessary to represent games in extensive form.

Definition 2.2.4 (Extensive Form Game) *An extensive form game is a tuple*

$$G = ((T, \models), (A, \alpha), (I, \iota), (H), \rho, u)$$

conforming to the description above.

The extensive form of a game contains the combinatoric information which describes the game. The remaining information consists of real numbers.

2.2.3 Strategy

A *strategy* for a player is a description of his or her behavior during the game. A variety of notions of strategy are useful since, among other things, it matters whether the behavior being described is intended, actual (perhaps in a statistical sense), or expected by others. We consider three concepts.

Definition 2.2.5 (Behavior, Pure, and Mixed Strategies) *A* behavior strategy *for player i is a function $\sigma_i : H_i \to \Delta(A)$ such that for each $h \in H_i$, $\sigma_i(h)$ assigns positive probability only to those moves in $\mathrm{m}(h)$. A* pure strategy *is a behavior strategy that assigns only degenerate probabilities, i.e. it specifies exactly one feasible move at each $h \in H_i$. The set of pure strategies for agent i is denoted by S_i. A* mixed strategy *for agent i is a probability measure on S_i.*

Intuitively a behavior strategy suggests that the agent makes decisions one by one as situations arise. In principle, though, the agent can make all decisions at the beginning by selecting a pure strategy, after which he or she (or perhaps a delegated subordinate) simply executes the decisions made initially. The information set-by-information set randomizations of a behavior strategy can be replicated by a mixed strategy that assigns to each pure strategy the product of the probabilities (given by the behavior strategy) of all the moves that the pure strategy specifies. This raises the question of whether the space of behavior strategies is large enough to capture all the possibilities presented to the agent by the space of mixed strategies. A theorem of Kuhn [39] shows that, for games of perfect recall, the answer is affirmative: for any mixed strategy there is a behavior strategy that is 'realization equivalent' in the sense that, for any mixed strategies for the other agents. the mixed strategy and the behavior strategy induce the same probability distribution on terminal nodes.

Kuhn's result [39] has a special significance for our work here, since the dimension of the space of behavior strategies (the sum, over all information sets where the agent moves, of the number of allowed moves minus one) is typically much lower than the dimension of the space of mixed strategies (negative one plus the product, over all information sets where the agent moves, of the number of allowed moves). Since the algorithms for systems of polynomials considered here have running times that grow exponentially with the number of variables, it should be more fruitful to apply them to solution concepts expressed in terms of behavior strategies.

Although the set of mixed strategies is ill-suited for computation, it gives a view of the game that is notationally simple, so that the following notion of a game is a popular starting point of theoretical analysis. An *n-person normal form game* is a $2n$-tuple $(S_1, ..., S_n; u_1, ..., u_n)$ where each S_i is a finite set of *pure strategies.* We call the elements of $S = S_1 \times \ldots \times S_n$ *pure strategy vectors.* The *utility function* $u_i : S \mapsto \Re$ associated with player i maps pure strategy vectors to von Neumann-Morgenstern utilities. As indicated above, there is a canonical procedure for passing from an extensive form game to an associated normal form game, and the possibility, in the extensive form, of choosing behavior in advance, suggests that the given extensive form and the derived normal form should be equivalent from the point of view of strategic considerations. This point has been argued with special force by Kohlberg and Mertens [40], but is not fully accepted since its consequences (that different extensive forms with the same derived normal form should be viewed as equivalent, and that solution concepts should be invariant under this notion of equivalence between games) are not completely understood, and seem paradoxical in some instances.

We can also think of *Strategy* as the approach used by the players to select which move to make from all the alternatives available, using information accessible to them, at their turn to move. Formulation of *strategy* is the most fundamental concept that emerges from investigating games in extensive forms.

A *pure strategy* for a player i specifies a single next move for that player from all the possible legal moves. On the other hand, a *mixed strategy* is one in which several next moves can be chosen with some probability distribution. In this paper, we are concerned with mixed strategies. We also require that a strategy of player i may depend only on components of the position visible to player i.

	Player 1		
		L	R
Player 2	l	(-2,+2)	(+1,-2)
	r	(+2,-1)	(-1,+1)

Table 6: Simple Guessing Game in Normal Form

Recall, $H_i \equiv \{\pi \in H \mid \iota(\pi) = i\}$. In § 3, we will need to restrict mixed strategies to satisfy certain equilibria criteria. Formally, we define mixed strategy as follows:

Definition 2.2.6 (Mixed Strategy) *For a player i the* strategy *is a partial function $\sigma_i : H_i \mapsto \Delta(T)$ such that:*

1. *For any $\pi \in H_i$, $\sigma_i(\pi)$ is a probability distribution on the legal next moves after π. This distribution reflects the probability that $\pi' \in F(\pi)$ is chosen by player i.*
2. *If $\pi, \pi' \in H_i$ and $vis_i(\pi) = vis_i(\pi')$, then $vis_i(\sigma_i(\pi)) = vis_i(\sigma_i(\pi'))$.*

Thus, σ_i is a guide for player i to select the next move. Rule 1 above restricts player i to legal moves, whereas rule 2 insures that the strategic decisions must be made using only the knowledge visible to player i. We say that a play π is *induced by strategy* σ if whenever π' is a (not necessarily proper) prefix of π, and π' is in the domain of σ, then all prefixes of π which occur with a nonzero probability are contained in the set $\sigma(\pi')$. Pure strategies can be thought of as a special case of mixed strategies in which exactly one node occurs with probability 1.

Let S_i be the set of strategies for player i, and $S = S_1 \times .. \times S_n$ be the set of strategies for the game. For any $\sigma \in \Delta(S)$ and $z \in Z$ the probability measure using strategy σ is:

$$Prob^{\sigma}(z) = \rho(\omega(z)) \prod_{l=1}^{\ell(z)} \sigma_{\iota(\mathrm{p}_{l(z)})}(\alpha(p_{l-1}(z)))$$

We denote the expected utility of player i under strategy σ by $E^{\sigma}[u_i(z)]$.

2.3 Games in Normal Form

Although representation of a game in extensive form contains all relevant details, often a significant fraction of information can be superfluous and over-specialized from the game theoretic perspective. A more concise representation of games is simply by strategies alone. This is known as *Normal Form representation.*

Consider Figure 1 which presents a game in extensive form. The game essentially has affords two strategies for each player. Player 1 can choose to move L or R, and Player 2 chooses between l and r. We can depict these strategies along with the associated payoffs in normal form as shown in Table 6.

Formally, an *n-person normal form game* is a $2n$-tuple $(S_1, ..., S_n; u_1, ..., u_n)$ where each S_i is a finite set of *pure strategies.* A probability distribution over $S_1 \times S_2 \times ... \times S_n$ can be used to specify *mixed strategies.* We call the elements of $S = S_1 \times .... \times S_n$ *pure strategy vectors*, and the probability distribution in $\Delta(S)$ is associated with *mixed strategies*. We also have a *utility function* $u_i : S \mapsto \Re$ ($\Re$ denotes the set of real numbers) associated with each player i, which maps strategies to payoffs expressed as real numbers. This function specifies the payoff as von Neumann-Morgenstern utility (in some conservative quantity like money) awarded to each one of the players from the game. When a game is represented in normal form as described above, the underlying rules of the game become irrelevant, and the game tree becomes superfluous. However, if one insists we can intuitively think of an n-person normal game as an extensive form game with the following rule: Player i chooses strategy σ_i from the set of strategies S_i before the choices of other involved players are known to him/her (player i).

We can also construct an equivalent normal form for a game initially presented in extensive form as described below. For each player i, compute the set of all available pure strategies open to player i in the extensive form of the game. Here, *pure strategy* means a specification of allowed actions at each state where player i is to "move". A pure strategy accounts for all possible sequences of "moves" which can occur. We, let $u_i(\sigma_1, ..., \sigma_n)$ be the "expected payoff" for player i, with respect to the probability distribution of the terminal plays resulting from the initial assessment, when the players select the strategies $\sigma_1, ..., \sigma_n$, respectively.

By results of Azhar, Peterson, and Reif [29], there is no recursive procedure to compute a game's normal form except if the game has less than three players, or if the game is hierarchical in the sense described in Section 1.4. Game theorists have disputed as to whether important information is lost in the process of the transformation from an extensive form game to a normal form game. This depends on details of the equilibrium concept, and in fact will prove not to be an issue for sequential equilibrium.

2.4 Game Theory Perspectives

In game theory, it is standard practice to consider only games of perfect recall, in the sense that all the plays in an information set must be consistent with everything observed by the player whose turn it is. The formal expression of this notion is cumbersome and tedious, but it has been elucidated in § 2.2.2.

Chance (or nature) may be a player in its own right. If the probabilities associated with its choices are known, we can construct a frequency distribution for all possible outcomes, and from this distribution we can compute the 'expected outcome' as well. However, in our discussion, we are not allowing chance (or nature) an opportunity to participate in the game. The reason is that we do not want to get distracted by superfluous complications, especially since, for all practical intents and purposes, we can transport all chance moves back to the initial stage of initial assessment. As a result of such treatment of chance moves, we are able to gain simplicity of notation and description. Nevertheless, we will discuss chance moves where it seems to us that the implicit generalization of the material may not be obvious.

In games of perfect information, each and every player is in a position to access all relevant information to know the exact state of the game before it decides which one of the possible alternative moves to select. Chess is an example of a perfect information game. In Chess, a player has complete knowledge of the board position on its turn before it selects one of the options available. It may very well be true that a player does not select the optimal move for the position at hand, but the fact remains that the player's computational shortcoming is not a result of lack of information.

On the other hand, in games of incomplete information the players are forced to make choices without complete information about the state of the game. An example of a game of incomplete information is the card game bridge. At any stage during hand play, a player knows everything about his cards, but does not have complete information about the hands of other players. In general, a player may be given a set of branch points to which the game may have progressed. Such sets are called the *information sets*. In bridge, information sets correspond to partial information gathered during the course of the play (e.g. one of the opponents is out of the trump spades but has high diamonds, and the other opponent is out of hearts). A nontrivial amount of information is still up to speculation. The rules of the game determine all legal moves which the player can make in a given situation. In order for this to make sense, the set of allowed actions must be the same at all points in an information set. Furthermore, for each node in an information set and each allowed action, there must be a unique successor node resulting from the action. In the running example of bridge, the content of opponents' or partner's hand does not affect a player's legal leads, or response to another player's leads. Some such plays might not be profitable in view of the contents of information sets, but are they nonetheless are legal.

3 Equilibrium

Many notions of "equilibrium" have been defined during the development of game theory. The first formal definition of an equilibrium concept for general games, which generalized the notion of a minimax point for two player zero-sum games, was formulated by Nash. Since then, modifications, mainly in the direction of increasing restrictiveness, have been proposed to overcome certain problematic features. This paper concentrates on one of these, namely the concept of *Sequential Equilibrium* proposed by Kreps and Wilson [1], but we should emphasize that, although this concept has computational advantages, the algorithms for semi-algebraic geometry presented below could be applied to almost every known solution concept for finite games, since (with few exceptions) these concepts are defined in terms of polynomial equations and inequalities in the payoffs and strategic probabilities.

Since sequential equilibrium is a rather complex notion, it can be better understood in relation to alternative equilibrium concepts.

3.1 Nash Equilibrium

3.1.1 Formulation of Nash Equilibrium Concept

Intuitively, a vector of mixed strategies is a Nash Equilibrium if every player's strategy is an optimal response to the other players' strategies in the sense that no player could increase his or her expected payoff by deviating from the given ('equilibrium') strategy, assuming other players persist with their ('equilibrium') strategies.

Let us develop some notational machinery to aid formal definition of Nash Equilibrium. Consider a normal form game $(S_1, ..., S_n; u_1, ...u_n)$ and the associated notation developed in § 2.2.3. Given a *mixed strategy vector* $\sigma = (\sigma_1, ..., \sigma_n) \in \times_i \Delta(S_i)$, one can compute an expected payoff for any player j. Assuming that the behaviors of various players are statistically independent, the expected payoff is given by the following expression:

$$\mathbf{E}^\sigma[u_j] = \sum_{s \in S} \sigma_1(s_1) \cdot \ldots \cdot \sigma_n(s_n) \cdot u_j(s).$$

Notation 3.1.1 (Substitution) *For $\sigma \in \times_i \Delta(S_i)$ and $\tau_j \in \Delta(S_j)$, let $\sigma|\tau_j$ be the mixed strategy vector obtained by replacing σ_j with τ_j.*

Definition 3.1.1 (Best Response) *We say that τ_j is a best response for player j to σ if $\mathbf{E}^{(\sigma|\tau_j)}(u_j) \geq \mathbf{E}^{(\sigma|\omega_j)}(u_j)$ for all $\omega_j \in \Delta(S_j)$; the set of such best responses is denoted by $BR_j(\sigma)$.*

Since $\mathbf{E}^{(u_j|\sigma)}$ is a multilinear function (in the obvious sense), hence linear in σ_j, $BR_j(\sigma)$ is the set of probability measures on S_j that assign positive probability to pure strategies that are best responses to σ. In particular, $BR_j(\sigma)$ is nonempty.

Definition 3.1.2 (Best Response Correspondence) *The best response correspondence BR: $\Delta(S_1) \times ... \times \Delta(S_n) \mapsto \Delta(S_1) \times ... \times \Delta(S_n)$ is defined by $BR(\sigma) = BR_1(\sigma) \times ... \times BR_n(\sigma)$.*

Definition 3.1.3 (Nash Equilibrium) *A mixed strategy vector σ^* is a Nash equilibrium if $\sigma^* \in BR(\sigma^*)$.*

In addition to proposing this equilibrium concept, Nash [41] pointed out that the best response correspondence satisfies the hypotheses of the Kakutani fixed point theorem [42], so the set of Nash equilibria is always nonempty. Clearly this is a minimal requirement for jointly rational behavior because if a mixed strategy vector does not satisfy this condition then some player can obtain a higher expected payoff by changing to another strategy.

3.1.2 Inherent Difficulties Associated with the Notion of Nash Equilibrium

Although Nash equilibrium has always had a central position in game theory, it has, in every period, been subject to attacks from one direction or another. We first briefly describe the questions that have been raised concerning its relevance, then proceed to the arguments suggesting that it is less restrictive than it should be; the latter considerations motivate the definition of sequential equilibrium.

Within the traditional "one shot" interpretation of a game – as a social situation that occurs only once, and encompasses all interactions between the agents – it is difficult to explain how the agents learn each other's strategies. Bernheim [43] and Pearce [44] (independently) formalized this critique, arguing that, for the one shot interpretation, a weaker solution concept is the correct description of the consequences of the assumption that the rationality of all agents is common knowledge. Recently another interpretation has been advanced as a foundation for Nash equilibrium, namely that the game occurs repeatedly, so that historical information can be consulted in forming expectations about the behavior of other agents, but the agents involved in any particular instance do not expect to encounter each other in the future, so that considerations of revenge, rewards, reputation, and so forth, do not drive a wedge between the current payoffs and true motivations. In sharp contrast to the one-shot interpretation, this view has no difficulty with multiple equilibria: different cultures can have different, stable, sets of customs.

The modern view implicitly regards Nash equilibria as stationary states of an underlying process of strategic adjustment, but general principles do not provide clear cut guidance for modeling these dynamics. This is an area of active research and presently the support for Nash equilibrium provided by this interpretation seems less than complete in the following sense: although any stationary point of a reasonable adjustment process should be a Nash equilibrium, it may happen that all equilibria are unstable, with the only stable phenomena being cycles or more complicated attractors.

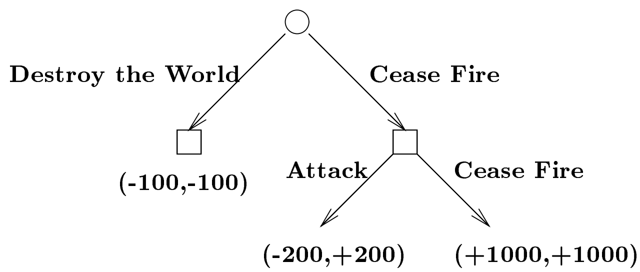

Figure 2: Death Before Dishonor

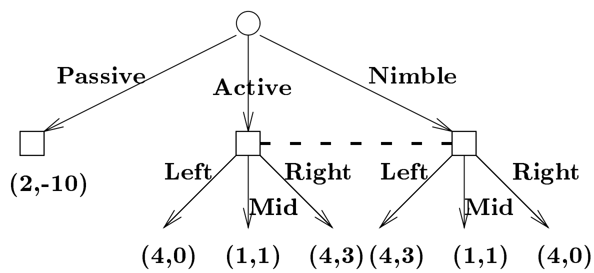

Figure 3: Extensive Form of a Game without 'Beliefs' with Nash Equilibrium (*Passive*, *Mid*).

Finally Nash equilibrium is vulnerable to the critique that, as a matter of both casual observation and extensive experimental evidence, it is simply wrong as a predictor of behavior in particular games. A careful evaluation of this point seems to require the elaboration of a philosophy of inexact science, so we will not discuss it further.

For us the most relevant criticism of Nash equilibrium is that it is incomplete as a description of the consequences of it being common knowledge that all agents are rational.

A simple and compelling example is the *Death before Dishonor* game shown in Figure 2. Clearly, a mutual *Cease Fire* is the optimal outcome for both players. However, *Destroy the World* is a Nash equilibrium if Player 1 believes that Player 2 will choose to *Attack*.[9]

Figure 3 presents a slightly more complicated example. Here, as the reader can easily verify, (*Passive*, *Mid*) is a Nash equilibrium. This time *Mid* is not a dominated move for Player 2, but it is unreasonable since there are no 'beliefs' about the relative likelihood of the plays in Player 2's information set for which *Mid* is a best response.

What these examples illustrate is that Nash equilibrium does not require rationality, in the sense of payoff maximization, in situations that occur with probability zero under the equilibrium strategies. Perhaps more important, agents are not required to believe that others will behave rationally in such situations. This seems clearly contrary to the spirit of the equilibrium concept, which is to elaborate those properties of behavior which follow from the assumption that the

[9] If Player 2 is attacking, destruction of the world is a Nash equilibrium because neither player can increase expected utility by changing strategy. This Nash equilibrium is not only illogical, but also unrealistic and absurd: Player 1 is countering a threat that Player 2 could not rationally carry out.

rationality (in any contingency) of all agents is common knowledge.

Although most researchers accepted the Nash equilibrium concept as central, a certain amount of confusion has surrounded its interpretation.

The players are required to behave as if they are unaware of their strategic interdependence. This assumption is plausible only if there are numerous players, and the consequence of a single deviation is negligible. Cournot tatonnement [45] is an example of dynamic process where each player adjusts its strategy optimally under the myopic assumption that the others will not alter the game situation. However, this presumption of every player is rendered erroneous after each move [45]. If this process converges we must arrive at a Nash equilibrium. Nevertheless, the players go through a series of adjustments without any guarantee of the process converging.

If, as can easily happen, there are several Nash equilibria, the theory seemed to be at odds with itself, since the information available is insufficient to lead the players to select one of the equilibria. Nash equilibria are the only expectations that are potentially self-reproducing when players respond rationally. They can be thought of self-fulfilling prophecies or self-enforcing agreements. Guaranteeing the uniqueness of Nash equilibrium is a difficult procedure which can be applied only to special types of games. On the other hand, in the interpretation of game as a social interaction, multiple equilibria are no more mysterious than the fact that two cultures can have different but stable sets of customs.

When a game is played repeatedly, the Nash equilibrium concept does not accommodate derivation of any benefit to a player by learning about the game, or understanding the behavior of other players, or incorporating any other knowledge. In reality, mixed strategies represent the expectations the players have about each other's behavior, where these expectations are derived from the society's history of behavior in this game.

Nash equilibria outcomes are not Pareto optimal in general. We refer interested readers to Grote's analysis [46] for a rigorous treatment of Pareto optimality concerns in reference to Nash equilibria.

Although Nash equilibrium seems to be a coherent description of rational behavior in such interpretations, it is far from a complete description. This can be shown by the simple and compelling example of *Death before Dishonor* game formulated in Figure 2. As we noted earlier, a mutual *Cease Fire* is the optimal outcome for both players. However, *Destroy the World* is a Nash equilibrium if Player 1 believes that Player 2 has chosen to *Attack*[10]. This Nash equilibrium is not only illogical, but also unrealistic and absurd: Player 1 is countering a threat that Player 2 could never carry out. This is illustrated more vividly by the next example.

Consider the extensive form in Figure 4, and the derived normal form shown in Table 7. The pair (*Draw*, *Passive*) is a Nash equilibrium of this game: Player 1 does best to play *Draw* if he expects Player 2 to play *Passive*, if Player 1 assigns all probability to *Draw*, then Player 2's strategy has no effect on his expected payoff (so there is no better response than *Passive* to *Draw*).

This equilibrium is clearly unreasonable when we examine the above game in its extensive form. If Player 1 chooses *Gamble* then Player 2's strategy will influence its payoff. On the other hand, Player 2 will be prefer to choose *Active*. Player 1 can anticipate this, so he will choose *Gamble*. Game theorists say that the equilibrium (*Draw*, *Passive*) is sustained by a 'non-credible' threat.

Figure 3 presents a slightly more complicated example. Here, as the reader can easily verify, (*Passive*, *Mid*) is a Nash equilibrium. This time *Mid* is not a dominated move for Player 2, but it

[10]If Player 2 is attacking, destruction of the world is a Nash equilibrium because neither player can increase his utility by changing its strategy

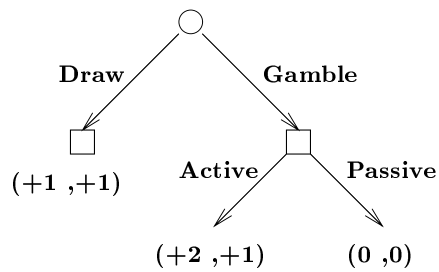

Figure 4: Extensive Form a Game with Nash Equilibrium (*Draw*, *Passive*) sustained by a 'Non-credible' Threat.

Player 2

Player 1		*Active*	*Passive*
	Draw	(+1,+1)	(+1,+1)
	Gamble	(+2,+1)	(0,0)

Table 7: Normal Form of a Game with Nash Equilibrium (*Draw*, *Passive*) Sustained by a 'Non-credible' Threat.

is unreasonable since there are no 'beliefs' about the relative likelihood of the plays in Player 2's information set for which *Mid* is a best response.

3.2 Perfect Equilibrium

The next major advance after Nash [41] was the paper by Selten [4], which defined the (rather optimistically titled) notion of perfect equilibrium. For sake of completeness, we present a brief overview of this notion. We first define this notion for the normal form, then consider how it can be applied to extensive form games.

In earlier work Selten had defined the notion of *subgame perfection* as a response to these difficulties. A node in the game tree is the initial node of a *subgame* if all agents always know whether it has occurred, i.e. every information set is either contained in the set consisting of the given node and its descendants or disjoint from this set. A behavior strategy is a *subgame perfect equilibrium* if it is a Nash equilibrium and its restriction to every subgame is a Nash equilibrium of the subgame. The undesirable Nash equilibrium of *Death Before Dishonor* is disqualified by this criterion, but this concept does not handle the other example presented above in which there are no nontrivial subgames, and in general the notion of a subgame is rather special, so that one should not expect subgame perfection to rule out all the "bad" equilibria of the sort that it addresses. The following more restrictive equilibrium notion does not suffer from this flaw.

Definition 3.2.1 *[Perfect Equilibrium]* $\sigma^* \in \times_i \Delta(S_i)$ *is a perfect equilibrium if there is a sequence* $\{\sigma^r\} \subset \times_i \Delta^o(S_i)$ *with* $\sigma^* \in BR(\sigma^r)$ *for all* r *and* $\sigma^r \mapsto \sigma^*$.

Remark 1 *The most important mathematical fact about the set of perfect equilibria is that it is a nonempty subset of the set of Nash equilibria.*

Remark 2 *Note that a perfect equilibrium necessarily assigns probability 0 to all pure strategies s_i that are* weakly dominated*: we say that strategy s_i is weakly dominated by strategy t_i if there $u_i(s_i, s_{I-\{i\}}) \leq u_i(t_i, s_{I-\{i\}})$ for all $s_{I-\{i\}} \in S_{I-\{i\}} = \times_{j \neq i} S_j$ with strict inequality for at least one $s_{I-\{i\}}$.*

The reader may wish to examine the examples above to verify that the concept of Nash equilibrium described above as unreasonable is also not perfect in the sense of Definition 3.2.1.

It turns out that applying the concept of perfection to the normal form derived as above is not the correct application of the perfect equilibrium concept to extensive form games. Informally, the difficulty is that normal form perfection does not require a player to behave rationally at an information set that cannot be reached under the player's equilibrium strategy due to a choice made by the player at an earlier information set. For a more expansive treatment of this point, with examples, the reader is referred to § 13 of Selten's paper [4].

Those unfamiliar with the intellectual history of game theory may wonder why such a long time elapsed between Nash's paper and later attempts to develop more refined concepts. Certainly an important factor was the belief that the normal form was sufficient for the description of rational behavior in equilibrium. The fact that notation for extensive form games is bulky and cumbersome may also have contributed to the delay.

Between 1950 and 1975, the accepted solution concept for games in extensive form was Nash equilibrium applied to the associated normal form game. This solution concept has the difficulty (which is not apparent when looking only at the normal form) that it allows irrational behavior at information sets that are reached with zero probability. In addition, any sensible theory of rational behavior at unreached information sets must incorporate some theory of beliefs concerning the relative likelihood of the nodes in such informations sets. Presuming that the beliefs are irrelevant at information sets reached with zero probability would not render complete treatment, since the equilibrium concepts has to be well-defined to account for all possible deviations of the players from equilibrium.

3.3 Sequential Equilibrium

A more suitable approach is to apply the perfect equilibrium concept to the 'agent normal form', which is obtained by regarding each information set as a different player. The mixed strategy vectors of the agent normal form are, of course, precisely the behavior strategies. To see the import of perfect equilibrium in this context, suppose that σ^* is a perfect equilibrium of the player normal form, and that $\langle \sigma^r \rangle$ is a sequence of totally behavior strategies converging to σ^* with $\sigma^* \in BR(\sigma^r)$ for all r. Suppose the initial assessment assigns positive probability to all initial plays. Then, for any r, the behavior strategy σ^r induces a probability distribution on terminal plays that is totally mixed, since any allowed sequence of moves has positive probability. In particular, every play in the game tree occurs with positive probability, so the condition $\sigma^* \in BR(\sigma^r)$ implies utility maximizing behavior at information sets that occur with probability zero under σ^*.

Moreover, each σ^r induces a well defined conditional probability distribution on the plays in each information set. For a given information set this distribution may be thought of as the Bayesian beliefs of the player who chooses there. Knowledge of the strategy σ^* alone is enough to compute that player's expected payoff conditional on any node in the information set and any choice of action. Combining this information with a belief, in the sense of a distribution over the nodes in the information set, one can compute expected payoffs for each of the available actions, thereby defining a notion of rationality.

We must confront the following problem. The beliefs at an information set are conditional probabilities, and are not unambiguously determined by σ^* unless the conditioning event (i.e. the information set) occurs with positive probability. At the same time, even when they are not completely determined, they should not be completely arbitrary; among other things an agent should not believe that his opponents' actions are correlated when the opponents have no way to coordinate their behavior. Kreps and Wilson [1] discuss a number of conditions that could be imposed on the relationship between strategies and beliefs, settling, with misgivings, on Definition 3.3.1 below.

Definition 3.3.1 (System of Beliefs) *A* system of beliefs *is a function $\mu : X \mapsto [0,1]$ such that $\sum_{x \in h} \mu(x) = 1$ for each $h \in H$.*

Definition 3.3.2 (Assessments) *An* assessment *is a pair (σ, μ) in which σ is a behavior strategy and μ is a system of beliefs.*

Definition 3.3.3 (Consistent Assessments) *We say that an assessment (σ, μ) is* consistent *if it is the limit of a sequence $\{(\sigma^r, \mu^r)\}$ where each σ^r is totally mixed and μ^r is the system of beliefs derived from σ^r by forming conditional probabilities.*

Definition 3.3.4 (Sequentially Rational Assessments) *We say that an assessment (σ, μ) is* sequentially rational *if the behavior strategy σ assigns positive probability only to those moves whose expected payoffs, computed from (σ, μ) in the manner described above, are not less than the expected payoff of some other move allowed at the same information set.*

Definition 3.3.5 (Sequential Equilibrium) *A sequential equilibrium is an assessment that is both consistent and sequentially rational.*

Remark 3 *The sequential equilibrium concept is weaker than the notion of agent normal form perfection in the following sense:*

> *If σ^* is a perfect equilibrium, then it is the limit of a sequence $\langle \sigma^r \rangle$ of totally mixed strategies with $\sigma^* \in BR(\sigma^r)$ for all r, and this implies that σ^* is a best response to each assessment (σ^r, μ^r) in the sense described above. The expected payoffs of moves are continuous functions of the assessment, so if μ^* is a limit point of the sequence $\langle \mu^r \rangle$, then σ^* is a best response to the assessment (σ^*, μ^*), and of course this assessment must also be consistent. This argument shows that every perfect equilibrium of the player normal form is the behavior strategy component of a sequential equilibrium.*

There are sequential equilibria whose behavior strategy components are not perfect equilibria. Among other things, a (weakly) dominated action may have positive probability in a sequential equilibrium, but not in a perfect equilibrium. The sequential equilibrium can be regarded as the natural generalization of the Nash equilibrium for games in extensive form. It is similar in spirit, since sequential rationality is a minimal condition for rational behavior in the environments in question. (The notion of consistency may not be a minimal requirement for beliefs to be sensible. See §5 of Kreps and Wilson's paper [1] for a discussion of this issue.) It is possible [47] to define a best response correspondence that has the set of sequential equilibria as its set of fixed points and that coincides with the best response correspondence defined above in the special case of normal form games, so the sequential equilibrium is also a natural generalization of the Nash equilibrium from a mathematical viewpoint.

4 Algebraic Characterization of Sequential Equilibrium

4.1 Consistency and Sequential Rationality

The language of the theory of real closed fields consists of formulae and propositions built up out of polynomial equations and inequalities, together with the logical quantifiers "for all" and "there exists." In this subsection we present formulae that make it evident that the definition of the sequential equilibrium is a formula in this language. This result alone suffices to imply the applicability of the algorithms for computing semi-algebraic sets discussed in §4.5. In subsequent subsections we reexpress the definition in ways that do not employ any logical quantifiers. Insofar as the computational complexity of the algebraic algorithms is exponential in the number of variables, and the typical method of computing sets defined by quantified formulae is to compute the set defined by the associated unquantified formula, then project, this reexpression should result in a considerable reduction in computational complexity.

Let BS denote the space of behavior strategies, and let BS^o denote the subspace of interior behavior strategies. Given $\sigma \in BS$, we can compute the probability that an arbitrary play $\pi \in T$ will occur by multiplying the initial assessment of the initial position (from which that play originates) with probabilities of each move along the play.

$$Prob^{\sigma}(\pi) = \rho(\omega(\pi)) \cdot (\sigma(\alpha(p_1(\pi))) \cdots \sigma(\alpha(p_{\ell(\pi)}(\pi)))).$$

This equation reflects our assumption that the players' behavior is statistically independent, that is, they cannot correlate their behavior with each other, or with unobserved choices of nature.

Let $M = \times_{h \in \mathrm{H}} \Delta(h)$ be the space of *beliefs*, and let $M^o = \times_{h \in \mathrm{H}} \Delta^o(h)$ be the space of *interior beliefs*. Recall that ρ is assumed to be interior, so if σ is interior then $Prob^{\sigma}(\pi) > 0$ for all $\pi \in T$. In this situation, we can define Bayesian beliefs by taking conditional probabilities:

$$\text{For } \pi \in T: \qquad \mu_{\sigma}(\pi) = \frac{Prob^{\sigma}(\pi)}{Prob^{\sigma}(H(\pi))}$$

These numbers constitute a vector of probability distributions $\mu_{\sigma} \in \Delta^o(\mathrm{h})$.

Let the space of *consistent interior assessments* be $\Psi^o = \{(\mu_{\sigma}, \sigma) \mid \sigma \,\epsilon\, BS^o\}$, and let Ψ be the space of *consistent assessments* (recall that this is the closure of Ψ^o in $M \times BS$). The formulae above show that Ψ^o is a semi-algebraic set, and in general the closure of any semi-algebraic set is a semi-algebra set since, if F is the formula defining a set X, i.e. $X = \{x | F(x)\}$, then the closure is the set of $\overline{x}$ satisfying the formula "for all $\epsilon > 0$ there exists x such that $F(x)$ and $\|\overline{x} - x\|^2 < \epsilon$."

We now explain how an assessment (μ, σ) determines expected payoffs for the various moves. To begin with we note that σ alone determines expected payoffs for all players at all plays in a game tree.

Lemma 4.1.1 *For each behavior strategy σ there is a unique system of expected payoffs for each $\pi \in T$ where the payoff to player i at this node is denoted by $\mathbf{E}^{\sigma}(u_i \mid \pi)$ and defined (backward) inductively by:*

$$\mathbf{E}^{\sigma}(u_i \mid \pi) = \left| \begin{array}{l} u_i(\pi), \text{ for } \pi \in Z, \\ \\ \sum_{a \in \mathrm{m}(\pi)} \sigma(a) \cdot \mathbf{E}^{\sigma}(u_i \mid FC(\pi, a)), \text{ for } \pi \in X. \end{array} \right.$$

Proof: Define Z_j inductively by letting $Z_0 = Z$, and $Z_j = \{\pi \mid FC(\pi) \subset Z_{j-1}\}$. It is clear that the condition above defines $\mathbf{E}^{\sigma}(u_i \mid \pi)$ uniquely for all $\pi \in Z_j - Z_{j-1}$ if $\mathbf{E}^{\sigma}(u_i \mid \pi)$ has already been defined for all $\pi \in Z_{j-1}$. The proof follows from the principle of mathematical induction. □

At an information set $h \in H$, the possible consequences of choosing $a \in m(h)$ are the plays $FC(\pi, a)$, for $\pi \in h$. The expected payoff associated with move a is the belief-weighted average of the expected payoffs of the plays $FC(\pi, a)$.

$$\mathbf{E}^{\sigma,\mu}(u_i \mid h, a) = \sum_{\pi \in h} \mu(\pi) \cdot \mathbf{E}^{\sigma}((u_i \mid FC(\pi, a)).$$

Definition 4.1.1 (Sequentially Rational) *An assessment (μ, σ) is sequentially rational if, for all $h \in H$ and $a \in m(h)$, $\sigma(a) > 0$, implies that $\mathbf{E}^{\sigma,\mu}(u_i \mid h, a) \geq \mathbf{E}^{\sigma,\mu}(u_i \mid h, a')$ for all $a' \in m(h)$, that is, players do not assign positive probability to moves that have suboptimal expected payoffs.*

For our purposes, it is crucial that the equilibrium conditions be of the form $Q \geq 0$, where Q is a polynomial whose variables are the components of μ, σ, u, and possibly other quantities. To make this completely explicit note that the inductive definition of $\mathbf{E}^{\sigma}(u_i \mid \pi)$ shows that this term is a polynomial in the components of σ and u, so $\mathbf{E}^{\mu,\sigma}(u_i \mid h, a)$ is a polynomial in the components of μ, σ, and u. Now observe that sequential rationality is equivalent to the condition:

$$\forall a, a' \in m(h),\ h \in H: \quad \sigma(a) \cdot (\mathbf{E}^{\mu,\sigma}(u_{\iota(h)} \mid h, a) - \mathbf{E}^{\mu,\sigma}(u_{\iota(h)} \mid h, a')) \geq 0.$$

4.2 Functional Notations

In the remaining section, we will show how the set of consistent assessments can be decomposed into finitely many sets, each of which can be described by means of finitely many (unquantified!) polynomial inequalities, and subsequently we can employ results from the theory of real closed fields to compute solutions. The basic technique is to develop a set of constraining non-algebraic relationships, and then apply logarithms to these relationships to achieve algebraic constraints.

Definition 4.2.1 (Basis) *The* basis *of an assessment (μ, σ) is $b(\mu, \sigma) = b_{\mathrm{X}}(\mu) \cup b_{\mathrm{A}}(\sigma)$, where $b_{\mathrm{X}}(\mu) = \{\pi \in X \mid \mu(\pi) > 0\}$ and $b_{\mathrm{A}}(\sigma) = \{a \in A \mid \sigma(a) > 0\}$. In general, a basis is a set $b = b_{\mathrm{X}} \cup b_{\mathrm{A}} \subseteq X \cup A$.*

Intuitively, basis $b(\mu, \sigma)$ consists of the elements of Z (leaf nodes) which occur with non-zero probability under the system of beliefs μ, and all the moves ($\in A$) which are made under the strategy σ.

Definition 4.2.2 (Quasiconsistent) *A basis is said to be quasiconsistent if there is at least one move at every information set. Formally, a basis $b = b_{\mathrm{X}} \cup b_{\mathrm{A}}$ is quasiconsistent if $b_{\mathrm{A}} \cap m(h) \neq \{\}$ for all $h \in H$.*

Definition 4.2.3 (Consistent) *We say that a basis b is consistent if $b = b(\mu, \sigma)$ for some consistent assessment (μ, σ).*

Our principal concerns are:

1. To determine computable conditions for the consistency of a basis .
2. To provide an algebraic characterization of the set of consistent assessments for each consistent basis.
3. To bound the complexity of these tasks.

We now fix a basis b that will be the focus of our discussion for the remainder of this section. Since b cannot possibly be consistent unless it is quasiconsistent, the quasiconsistency of b will be a maintained hypothesis.

The following notation facilitates the analysis. For each $\pi \in X$ we let $m^\pi : (\Re^+)^{\mathrm{A}} \mapsto \Re^+$ be the monomial, whose variables are the components of some strategy ω, that computes the probability of reaching π from the root of the its tree:

$$m^\pi(\omega) = \prod_{k=0}^{\ell(\pi)-1} \omega(\alpha(p_k(\pi)))$$

For $\pi \in W$, $m^\pi(\omega) \equiv 1$ (for any ω). We can regard BS^o as a subset of $(\Re)^{\mathrm{A}}$ in the obvious way to derive the following formula for the probability that node π occurs:

$$Prob^\sigma(\pi) = \rho(\omega(\pi)) \cdot m^\pi(\sigma) \text{ for } \pi \in X \text{ and } \sigma \in BS^o.$$

For $\pi \in\ X$ we let $l^\pi : \Re^{\mathrm{A}} \mapsto \Re$ be the linear function:

$$l^\pi(\omega) = \sum_{k=0}^{\ell(\pi)-1} \omega(\alpha(p_k(\pi)))$$

Now observe that:

$$\ln m^\pi(\omega) = l^\pi(\ln \omega)$$

where $\ln \omega$ denotes the vector in $\Re^{|\mathrm{A}|}$ whose components are the natural logarithms of the components of ω. Our overall strategy is to conduct the analysis in logarithmic terms so that linear algebra can be brought to bear, then rephrase the results in multiplicative terms so that the relevant equations are algebraic.

We are now ready to define the requisite linear functions. Let $m(b) = b_{\mathrm{A}}$, and let $L_{\mathrm{m}(b)} : \Re^{\mathrm{A}} \mapsto \Re^{\mathrm{m}(b)}$ be the standard projection onto a coordinate subspace. We also define $EQ(b)$ to be the set of pairs of strategic plays in the basis that are in a common information set:

$$EQ(b) = \{(\pi, \pi') \mid \text{there is some } h \text{ with } \pi, \pi' \in h \cap b_{\mathrm{X}}\}$$

Let $L_{\mathrm{EQ}(b)} : \Re^{\mathrm{A}} \mapsto \Re^{\mathrm{EQ}(b)}$ be the linear function whose (π, π')-coordinate is $l^\pi - l^{\pi'}$. For $\sigma \in BS^o$, the (π, π')-component of $L_{\mathrm{EQ}(b)}(\ln \sigma)$ is then

$$L_{\mathrm{EQ}(b)}(\ln \sigma)[\pi, \pi'] = \ln\left(\frac{Prob^\sigma(\pi)}{Prob^\sigma(\pi')}\right) - \ln\left(\frac{\rho(\omega(\pi))}{\rho(\omega(\pi'))}\right),$$

so, among other things, for each information set h the information in $L_{\mathrm{EQ}(b)}(\ln\sigma)$ allows one to compute the conditional probability distribution on $h \cap b_{\mathrm{X}}$ induced by σ. Let $L : \gamma \mapsto (L_{\mathrm{m}(b)}(\gamma), L_{\mathrm{EQ}(b)}(\gamma))$ be the Cartesian product of $L_{\mathrm{m}(b)}$ and $L_{\mathrm{EQ}(b)}$. Roughly, the use of the following result will be to characterize "orders of improbability" that support the given basis b: if each move has an order of improbability, which we think of as the negation of the logarithm of the probability assigned to the move in a some behavior strategy, then (up to order of magnitude) the belief probabilities at each information set will be dominated by those nodes in the information set that are minimal with respect to the sum of the orders of improbability associated with moves required to reach the node.

Lemma 4.2.1 *There is a set of moves $B \subset A$ with the property that $L|_{R^B}$ is 1-1 and $L(\Re^B) = L(\Re^{\mathrm{A}})$. There exist algorithm(s) for finding such a set, which runs in space polynomial in the cardinality of the set T, i.e. $|T|$. Any such set satisfies $b_{\mathrm{A}} \subset B$.*

> **Proof**: Consider two vector spaces W_1 and W_2 such that $L : W_1 \mapsto W_2$ is linear, and $w_1, ..., w_n$ is a basis for a vector space W_1. We know from elementary linear algebra that $L(span\{w_1, ..., w_{i-1}, w_{i+1}, ..., w_n\}) \neq L(W_1)$ if and only if $\ker(L) \subseteq span\{w_1, ..., w_{i-1}, w_{i+1}, ..., w_n\}$. Consequently, in general, there must be some w_i with $L(span\{w_1, ..., w_{i-1}, w_{i+1}, ..., w_n\}) = L(W_1)$ unless $\ker(L) = \{0\}$. By computing certain determinants, we can ascertain whether or not this latter inclusion holds, provided we are given a matrix for L in terms of $w_1, ..., w_n$, and some basis for W_2. This can be accomplished by standard polylog algorithms for solving linear systems like Csanky's algorithm [48]. It is necessarily the case that $b_{\mathrm{A}} \subset B$, since otherwise $L_{\mathrm{m}(b)}(\Re^B) \neq L_{\mathrm{m}(b)}(\Re^{\mathrm{A}})$. The lemma follows. $\Box$

Let V be the kernel of L. Fixing a B satisfying the conditions of Lemma 4.2.1, we have $V \cap \Re^B = \{0\}$ and $V + \Re^B = \Re^{\mathrm{A}}$. Let $Proj_V : \Re^{\mathrm{A}} \mapsto V$ and $Proj_{\Re^B} : \Re^{\mathrm{A}} \mapsto \Re^B$ be the projections of $\Re^{\mathrm{A}}$ onto V and $\Re^B$, respectively (parallel to $\Re^B$ and V, respectively). Consequently, for $\gamma \in \Re^{\mathrm{A}}$ we have $Proj_V(\gamma) \in V$, $Proj_{\Re^B}(\gamma) \in \Re^B$, and $Proj_V(\gamma) + Proj_{\Re^B}(\gamma) = \gamma$.

4.3 Labeling

Our next step is to introduce a labeling system which assigns a non-negative real number for each node, and every edge.

Definition 4.3.1 (Labeling) *A labeling for the extensive form is a function K taking A and Y into the non-negative integers.*

In particular, we define b-labeling, and show that it is equivalent to the consistency properties of b:

Definition 4.3.2 (b-labeling) *A b-labeling is a function $K : Y \cup A \longrightarrow \Re^+$ satisfying the following conditions:*

1. *For $a \in A$: $K(a) = 0$ if and only if $a \in b_{\mathrm{A}}$.*
2. *For $\pi \in Y$: $K(\pi) = \sum_{k=0}^{\ell(\pi)-1} K(\alpha(p_k(\pi)))$.*

3. For $\pi, \pi' \in T$: If $(\pi, \pi') \in EQ(b)$ then $K(\pi) = K(\pi')$.

4. For $\pi, \pi' \in T$: If $\pi, \pi' \in h$, with $\pi \in b_{\mathrm{X}}$ and $\pi' \notin b_{\mathrm{X}}$, then $K(\pi) \leq K(\pi') + 1$.

We now show that the consistency of b is equivalent to the existence of a b-labeling because we have phrased the definition of a b-labeling in such a way that the set of b-labelings coincides with the set of feasible solutions of a particular linear program. Consequently, the following results show that the enumeration of the set of consistent bases can be achieved by the simplex algorithm.

Lemma 4.3.1 *If b has a b-labeling, then it is consistent.*

Proof: Suppose that K is a b-labeling.
Fix $\sigma \in BS^o$, and for $a \in A$ and positive integers n, we define σ_n:

$$\sigma_n(a) = c(n, MOVSET(a)) \cdot (\frac{1}{n})^{K(a)} \cdot \sigma(a)$$

where $MOVSET(a)$ is defined to be $m^{-1}(a)$, and the normalizing constant c is defined as follows:

$$c(n,h) = \frac{1}{\sum_{a \in \mathrm{m}(h)} (\frac{1}{n})^{K(a)} \cdot \sigma(a)}$$

Let μ_n be the belief derived from σ_n, and let (μ^*, σ^*) be a limit point of the sequence $\{(\mu_n, \sigma_n)\}$, i.e. $(\mu^*, \sigma^*) = \lim_{n\to\infty}(\mu^n, \sigma^n)$.

Since b is quasiconsistent, there must be a move available at every information set. Furthermore, by Condition 1, $K(a') = 0$ if and only if $a' \in b_{\mathrm{A}}$, therefore $(\frac{1}{n})^{K(a')} = 1$ if $a' \in b_{\mathrm{A}}$. Consequently,

$$\lim_{n\to\infty} c(n,h) = \frac{1}{\sum_{a' \in (\mathrm{m}(h) \cap \mathrm{m}(b))} \sigma(a)}.$$

Since the constant $c(n, h)$ is non-zero, it follows that $\lim_{n\to\infty} \sigma_n(a) > 0$ if and only if $\lim_{n\to\infty} (\frac{1}{n})^{K(a)} \sigma(a) > 0$, and since $\sigma(a)$ is always positive, this will be the case if and only if $K(a) = 0$, which by Condition 1 happens if and only if $a \in b_{\mathrm{A}}$. Recall that $\sigma^*(a) = \lim_{n\to\infty} \sigma_n(a)$. It follows that $\sigma^*(a) > 0$ if and only if $a \in b_{\mathrm{A}}$.

Recall,

$$Prob^{\sigma_n(\pi)} = \rho(\omega(\pi)) \cdot \prod_{k=0}^{\ell(\pi)-1} (\sigma_n(\alpha(p_k(\pi))))$$

For $\pi \in X$, our labeling scheme yields

$$Prob^{\sigma_n}(\pi) =$$

$$\rho(\omega(\pi)) \cdot \prod_{k=0}^{\ell(\pi)-1} c(n, MOVSET(\alpha(p_k(\pi)))) \cdot (\frac{1}{n})^{K(\alpha(\mathrm{p}_k(\pi)))} \cdot \sigma(\alpha(p_k(\pi)))$$

Applying Condition 2, we get:

$$Prob^{\sigma_n}(\pi) =$$

$$\rho(\omega(\pi)) \cdot (\frac{1}{n})^{K(\pi)} \cdot \prod_{k=0}^{\ell(\pi)-1} c(n, MOVSET(\alpha(p_k(\pi)))) \cdot \sigma(\alpha(p_k(\pi)))$$

By its definition:

$$\mu_n(\pi) = \frac{Prob^{\sigma_n}(\pi)}{Prob^{\sigma_n}(H(\pi))}$$

We see now that $\mu_n(\pi)$ is a constant times $(1/n)^{K(\pi)}$ divided by a sum in which the term corresponding to each $\pi' \in slH(\pi)$ is a constant times $(1/n)^{K(\pi')}$, so $\mu^*(\pi) = \lim_{n\to\infty} \mu_n(\pi) > 0$ if and only if $K(\pi) \leq K(\pi')$ for all $\pi' \in slH(\pi)$, and in view of Condition 4 this occurs precisely when $\pi \in b_X$.

Lemma 4.3.2 *If b is consistent then there exists a b-labeling.*

Proof: Suppose that b is consistent, so that $b = b(\mu^*, \sigma^*)$ for some $(\mu^*, \sigma^*) = \lim_{n\to\infty}(\mu_n, \sigma_n)$, where, for each n, σ_n is an interior strategy and μ_n is the belief derived from σ_n. We note that $\lim_{n\to\infty} \ln(\sigma_n(a)) = \ln(\sigma^*(a))$ for all $a \in b_{\mathrm{A}}$. Moreover, for $(\pi, \pi') \in EQ(b)$, we have:

$$\begin{aligned}
&\lim_{n\to\infty} l^{\pi}(\ln \sigma_n) - l^{\pi'}(\ln \sigma_n) \\
&\quad = \lim_{n\to\infty} \ln[Prob^{\sigma_n}(\pi)/\rho(\omega(\pi))] - \ln[Prob^{\sigma_n}(\pi')/\rho(\omega(\pi'))] \\
&\quad = \lim_{n\to\infty} \ln[\mu_n(\pi)/\rho(\omega(\pi))] - \ln[\mu_n(\pi')/\rho(\omega(\pi'))] \\
&\quad = \ln[\mu^*(\pi)/p(\omega(\pi))] - \ln(\mu^*(\pi')/\rho(\omega(\pi'))] \\
&\quad = \ln[\tfrac{(\mu^*(\pi)\rho(\omega(\pi')))}{(\mu^*(\pi')\rho(\omega(\pi)))}].
\end{aligned}$$

Consequently, the sequence $\langle L(\ln \sigma_n)\rangle$ is convergent.

Since $L = L \circ Proj_{R^B}$ and L is 1-1 on $\Re^B$, the sequence $\langle Proj_{R^B}(\ln \sigma_n))\rangle$ must also converge, and in particular it must be bounded.

On the other hand, for any $\pi, \pi' \in h$ such that $\pi \in b_{\mathrm{X}}$ and $\pi' \notin b_{\mathrm{X}}$, $l^{\pi}(\ln \sigma_n) - l^{\pi'}(\ln \sigma_n) \longrightarrow \infty$ as $n \longrightarrow \infty$. Furthermore, for any $\gamma \in \Re^{\mathrm{A}}$ we have $\gamma = Proj_{R^B}(\gamma) + Proj_V(\gamma)$ so the fact that $(Proj_{R^B}(\ln\ \sigma_n))$ is bounded implies $l^{\pi}(Proj_V(\ln\ \sigma_n)) - l^{\pi'}(Proj_V(\ln\ \sigma_n)) \longrightarrow \infty$ as $n \longrightarrow \infty$. Fixing an n such that $l^{\pi}(Proj_V(\ln\ \sigma_n)) - l^{\pi'}(Proj_V(\ln\ \sigma_n)) \geq 1$ for all $\pi, \pi' \in h$ such that $\pi \in b_{\mathrm{X}}$ and $\pi' \notin b_{\mathrm{X}}$, for each $a \in A$ let $K(a)$ be the a-component of $-Proj_V(\ln \sigma_n)$, and let $K(\pi)$ be defined by Condition 2. Conditions 1 and 3 follow from the fact that $Proj_V(\ln \sigma_n) \in \ker(L)$, and the inequality above implies Condition 4, so K is a b-labeling, and the Lemma follows. □

Theorem 4.3.1 *b is consistent if and only if it has a b-labeling.*

Proof: The theorem follows from Lemma 4.3.1 and 4.3.2 above. □

Remark 4 *Our proof that consistency implies the existence of a b-labeling is different from Kreps and Wilson's paper [1], and in fact we would like to take this opportunity to point out that their proof is incorrect. Specifically the K constructed at the bottom of their page 887 need not be a b-labeling, as the readers can verify for themselves.*

4.4 Algebraic Characterization of Consistent Assessments

Having characterized the consistent basis, we now turn to the problem of giving an algebraic characterization of the set of consistent assessments for a given consistent basis. Henceforth, we assume that b is consistent, and we fix a b-labeling K.

Our method is to introduce a set of auxiliary variables that is homeomorphic to the set of consistent assessments with basis b by an algebraic homeomorphism. Let

$$\Xi_{b,B} = \{\xi \in (\Re^{+})^{\mathrm{A}} \mid \xi(a) = 1 \text{ if } a \notin B, \text{ and } \forall h : \sum_{a \in \mathrm{m}(h) \cap \mathrm{m}(b)} \xi(a) = 1\}$$

Recall that $b_{\mathrm{A}} \subset B$, so $\Xi_{b,B}$ is nonempty.

Definition 4.4.1 (Strategy and Belief) *Given $\xi \in \Xi_{b,B}$, we define a behavior strategy $\sigma(\xi)$ by*

$$\sigma(\xi)(a) = \left| \begin{array}{l} 0, \text{ if } a \notin b_{\mathrm{A}}, \\ \\ \xi(a), \text{ if } a \in b_{\mathrm{A}} \end{array} \right.$$

We define an associated belief $\mu(\xi)$ by

$$\mu(\xi)(\pi) = \left| \begin{array}{l} 0, \text{ if } \pi \notin b_{\mathrm{X}}, \\ \\ \rho(\omega(\pi)) \cdot \frac{m^{\pi}(\xi)}{\sum_{\pi' \in \mathrm{H}(\pi) \cap b_{\mathrm{X}}} [\rho(\omega(\pi')) \cdot m^{\pi'}(\xi)]}, \quad \text{if} \pi \in b_{\mathrm{X}}. \end{array} \right.$$

Lemma 4.4.1 *The function $\xi \longrightarrow (\mu(\xi), \sigma(\xi))$ is a homeomorphism between Ξ_b and the set of consistent assessments (μ, σ) with $b(\mu, \sigma) = b$.*

Proof: Fix $\xi \in \Xi_{b,B}$. For $n = 1, 2, ...$ we define $\sigma_n \in BS^o$ by:

$$\sigma_n(a) = c(n, MOVSET(a)) \cdot \xi(a) \cdot (\frac{1}{n})^{K(a)}$$

where $c(n, h) = [\sum_{a \in \mathrm{m}(h)} \xi(a) \cdot (\frac{1}{n})^{K(a)}]^{-1}$.

Clearly, as $n \longrightarrow \infty$, $c(n, h) \longrightarrow 1$ for all h. Therefore, $\sigma_n \longrightarrow \sigma(\xi)$. Supposing μ_n be the belief derived from σ_n, straightforward computations show that $\mu_n \longrightarrow \mu(\xi)$. Consequently, the assessment $(\mu(\xi), \sigma(\xi))$ is consistent, and of course our construction guarantees that $b(\mu(\xi), \sigma(\xi)) = b$.

Conversely, suppose that (μ, σ) is consistent with $b(\mu, \sigma) = b$. Let $\langle \sigma_n \mid n = 1, 2, 3, ... \rangle$ be a sequence in BS^o with $(\mu_n, \ \sigma_n) \longrightarrow (\mu, \sigma)$ where, for each n, μ_n is the belief derived from σ_n. We claim that

$$(\mu, \sigma) = (\mu(\xi), \sigma(\xi)) \text{ for } \xi = \exp(\lim_{n \to \infty} Proj_{R^B}(\ln \ \sigma_n))^{11}$$

To begin with we must show that the limit exists, and this is equivalent to the convergence of the sequence $\langle L(\ln \sigma_n) \mid n = 1, 2, 3, ... \rangle$. However, for $a \in b_{\mathrm{A}}$ we have $\lim_{n \to \infty} \ln \sigma_n(a) = \ln \sigma(a)$, and for $(\pi, \pi') \in EQ(b)$ we have:

[11] For $x \in \Re^{\mathrm{A}}$, $\exp(x)$ is the vector with components $\exp(x(a))$.

$\lim_{n\to\infty} l^{\pi}(\ln\sigma_n) - l^{\pi'}(\ln\sigma_n)$
$\quad = \lim_{n\to\infty} \ln[\mu_n(\pi)/\mu_n(\pi')] - \ln[\rho(\omega(\pi))/\rho(\omega(\pi'))]$
$\quad = \ln[\mu(\pi)/\mu(\pi')] - \ln[\rho(\omega(\pi))/\rho(\omega(\pi'))]$.

We must now verify that $(\mu(\xi), \sigma(\xi)) = (\mu, \sigma)$. For $a \in b_{\mathrm{A}}$ we have:

$\sigma(\xi)(a) = \xi(a)$
$\quad = \exp(\lim_{n\to\infty} \ln\ \sigma_n(a))$
$\quad = \lim_{n\to\infty}\ \sigma_n(a) = \sigma(a)$.

For $(\pi, \pi') \in EQ(b)$ we have:

$\ln[\mu(\xi)(\pi)/\mu(\xi)(\pi'))] - \ln[\rho(\omega(\pi))/\rho(\omega(\pi'))]$
$\quad = l^{\pi}(\ln\ \xi) - l^{\pi'}(\ln\ \xi)$
$\quad = \lim_{n\to\infty}\ l^{\pi}(\ln\ \sigma_n) - l^{\pi'}(\ln\ \sigma_n)$
$\quad = \lim_{n\to\infty} \ln[\mu_n(\pi)/\mu_n(\pi')] - \ln[\rho(\omega(\pi))/\rho(\omega(\pi')))]$
$\quad = \ln[\mu(\pi)/\mu(\pi')] - \ln[\rho(\omega(\pi))/\rho(\omega(\pi'))]$.

For any finite set X and any $\lambda \in \Delta^o(X)$, the ratios $\frac{\lambda(\pi)}{\lambda(\pi')}$ completely determine λ. Consequently, we have shown that $\mu(\xi) = \mu$. We have proved that the image of $(\mu(\cdot), \sigma(\cdot))$ is precisely the set of consistent assessments with basis b.

We will now show that $(\mu(\cdot), \sigma(\cdot))$ is one-to-one. Suppose $(\mu(\xi_1), \sigma(\xi_1)) = (\mu(\xi_2), \sigma(\xi_2))$. Subsequently, $\sigma(\xi_1) = \sigma(\xi_2)$ implies that $\xi_1(a) = \xi_2(a)$ for all $a \in m(b)$. Hence, $L_{\mathrm{m}(b)}(\ln\xi_1) = L_{\mathrm{m}(b)}(\ln\xi_2)$. For $(\pi, \pi') \in EQ(b)$, the equation $\mu(\xi_1) = \mu(\xi_2)$ is easily seen to imply that:

$$l^{\pi}(\ln\xi_1) - l^{\pi'}(\ln\xi_1) = l^{\pi}(\ln\xi_2) - l^{\pi'}(\ln\xi_2).$$

Consequently, $L_{\mathrm{EQ}(b)}(\ln\xi_1) = L_{\mathrm{EQ}(b)}(\ln\xi_2)$. Thus $L(\ln\xi_1) = L(\ln\xi_2)$, but the condition $\xi_1(a) = \xi_2(a) = 1$ (for $a \notin B$) implies that $\ln\xi_1, \ln\xi_2 \in \Re^B$. Therefore, Lemma 4.2.1 yields $\xi_1 = \xi_2$.

The map $(\mu(\cdot), \sigma(\cdot))$ is visibly continuous, so it now suffices to show that its inverse is also continuous. Let $f_1 = L_{\mathrm{m}(b)} : \Re^{\mathrm{A}} \mapsto \Re^{\mathrm{m}(b)}$ be the standard projection. For $(\pi, \pi') \in EQ(b)$, let $f_2 : (\Re^+)^{b_{\mathrm{X}}} \mapsto (\Re^+)^{\mathrm{EQ}(b)}$ be given by

$$f_2(\mu)(\pi, \pi') = \frac{\mu(\pi)/\rho(\omega(\pi))}{\mu(\pi')/\rho(\omega(\pi'))}$$

Let $f(\mu, \sigma) = (f_1(\sigma), f_2(\mu)) \in \Re^{\mathrm{m}(b)} \times \Re^{\mathrm{EQ}(b)}$. The point of this construction is that if $(\mu(\xi), \sigma(\xi)) = (\mu, \sigma)$ for some $\xi \in \Xi_{b,B}$, then we must have $L(\ln\xi) = \ln f(\mu, \sigma)$. Since $\ln\xi \in \Re^B$ it follows that:

$$\xi = \exp[(L|_{\Re^B})^{-1}(\ln f(\mu, \sigma))].$$

The continuity of this formula establishes the continuity of $(\mu(\cdot), \sigma(\cdot))^{-1}$, and the lemma follows. □

4.5 Repercussions of Algebraic Consistency: Our Algorithms for Sequential Equilibria

We now summarize the results of this section in a way that displays the set of consistent assessments for a given basis as the projection of an algebraic variety.

Proposition 4.5.1 *We have established the following:*

- *A basis b is consistent if and only if it has a b-labeling.*
- *If a basis b is consistent and B is as in Lemma 4.2.1, then the set of consistent assessments with basis b is the set of $(\mu, \sigma) \in \Re^{\mathrm{X}} \times \Re^{\mathrm{A}}$ for which there is $\xi \in \Re^{\mathrm{A}}$ satisfying the following conditions*

  1. *For $(\pi \notin b_{\mathrm{X}})$:* $\mu(\pi) = 0$.
  2. *For $(a \notin b_{\mathrm{A}})$:* $\sigma(a) = 0$.
  3. *For $(a \notin B)$:* $\xi(a) = 1$.
  4. *For $(a \in b_{\mathrm{A}})$:* $\sigma(a) = \xi(a)$.
  5. *For $(h \in H)$:*
  $$\sum_{a \in \mathrm{m}(h) \cap \mathrm{m}(b)} \xi(a) = 1.$$
  6. *For $\pi \in b_{\mathrm{X}}$:*
  $$\mu(\pi) \cdot [\sum_{\pi' \in \mathrm{H}(\pi) \cap b_{\mathrm{X}}} \rho(\omega(\pi')) \cdot m^{\pi'}(\xi) = \rho(\omega(\pi)) \cdot m^{\pi}(\xi).$$

The procedure for computing the set of sequential equilibria has two phases, the first of which is the enumeration of the consistent bases. This can be accomplished using linear programming, and should be a relatively small part of the total computational burden. The second phase analyzes the set of sequential equilibria for each consistent basis. One possibility is to ask, for each consistent basis, whether the basis has any sequential equilibria. Since the set of consistent assessments is a semi-algebraic set, by virtue of Proposition 4.5.1, and the equilibrium conditions are semi-algebraic, such queries are instances of the existential question for the theory of real closed fields. Ben-Or, Kozen, and Reif [49], Canny [9] and Renegar [10] present algorithms for this problem. The most efficient, in terms of asymptotic requirements, is Renegar's, which requires polynomial space, and whose temporal requirements grow exponentially with the number of variables, but are polynomial in the size of the system for a fixed number of variables. More detailed information (dimension, number of components, etc.) can be obtained by applying the cellular decomposition algorithm of Kozen and Yap [11] to those consistent bases with nonempty sets of sequential equilibria. The latter algorithm also requires exponential (parallel) time.

The exponential time requirements of the algorithms described above are discouraging, since they suggest that the range of practical application of our procedures will be rather small, and will grow slowly with advances in computing technology. While we do not disagree completely with this assessment, we think it is mitigated by at least two factors. The first is simply the observation that algorithmic computation can hardly fail to be an improvement over calculation by hand, so that implementations of our procedures would have some usefulness. Second, there is the possibility of speed-up. To a certain extent this could be a matter of improving the general algorithms for dealing with systems of polynomials. Perhaps more interesting and fruitful, though, would be those improvements that take advantage of the specific nature of the problem. To a certain extent at least, the qualities of mind described vaguely by the phrase "strategic insight" are a matter of deft application of ideas that allow one to eliminate certain possibilities without extensive computation; the most elementary examples are the implications of dominance. Implementations of the algorithms given here would provide a framework in which such ideas could be precisely specified, tested, and refined.

In conjunction with our observation that the sequential rationality condition can be described in this fashion, this shows that the set of sequential equilibria decomposes into a finite system of semialgebraic sets (i.e. a set of polynomials with rational coefficients and real variables). Subsequently, we can apply the decision algorithm for deciding the existential theory of real closed fields Canny [9] and Renegar [10]. These algorithms run in space polynomial in its input, i.e. size of semialgebraic sets. Subsequently, we can apply the methods of Kozen and Yap [11] to compute the algebraic cell decomposition of the semialgebraic sets. Finally, using the results of Ben-Or, Feig, Kozen, and Tiwari [50], we can compute the roots in exponential space. This will give us (in the reverse transformed space), the connected components of consistent assessments, which give connected components of mixed strategies satisfying sequential equilibria. This algorithm runs in space exponential in its input, i.e. size of semialgebraic sets. Since the size of semialgebraic sets is polynomial in the size of the information sets, we get the following theorem specifying the complexity of our algorithm:

Theorem 4.5.1 *In space polynomial in the size of the information sets we can compute an example mixed strategy satisfying the sequential equilibria condition. Furthermore in space exponential in a polynomial of size of the information sets, and we can compute the connected components of mixed strategies satisfying sequential equilibria.*

5 Conclusion

In this paper, we reduced producing an example mixed strategy satisfying the sequential equilibrium to the existential theory of real closed fields. Furthermore, we can see that the solution of the resulting semialgebraic sets are polynomial in the size of the information set.

Result 1 *In space polynomial in the size of the information sets, we can compute an example mixed strategy satisfying the sequential equilibria condition. In space exponential in a polynomial of the size of the information sets, we can compute the connected components of mixed strategies satisfying sequential equilibria.*

We can unravel information by using techniques of Azhar, Peterson, and Reif [28, 29]. This yields the following result:

Result 2 *Given a recursively represented game, with a position space bound $S(n)$ and a log space computable next move relation, we can determine the existence of sequential equilibrium and compute an example mixed strategy satisfying the sequential equilibria condition, all in space bound $O(S(n)^2)$, Furthermore, in space $O(S(n)^3)$, we can compute the connected components of mixed strategies satisfying sequential equilibria.*

The sequential equilibrium concept is generally regarded as a powerful description of simultaneous rational behavior in an environment in which rationality is common knowledge. One possible application of the decision algorithm proposed here is as an experimental tool in the development and analysis of applications of game theory in computer science and economics. Another application of our algorithm is for exploration of new refinements to equilibria concepts.

There is an extensive literature concerned with the possibility of even further refining the sequential equilibrium concept, where certain equilibria are disallowed to accommodate further restrictions of concern to the economists. For example, see McLennan [51], Cho [52], and Kohlberg and Mertens [40].

References

[1] Kreps and Wilson. Sequential equilibria. *Econometrica*, 50(4):863–894, July 1982.

[2] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley Publishing Company, 1975.

[3] Harry Lewis and C.H. Papadimitriou. *Elements of Theory of Computation*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981.

[4] R. Selten. Spieltheoretische behandlung eines oligopolmodells mit nachfragetragheit. *Zeitschrift fur die gesamte Staatswissenschaft*, 121:301–324, 1975.

[5] James Renegar. On the computational complexity and geometry of the first-order theory of the Reals: Part iii - quantifier elimination. Technical Report 856, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853-7501, August 1989.

[6] James Renegar. On the computational complexity and geometry of the first-order theory of the Reals: Part i. Technical Report 853, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853-7501, July 1989.

[7] James Renegar. On the computational complexity and geometry of the first-order theory of the Reals: Part ii. Technical Report 854, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853-7501, July 1989.

[8] James Renegar. On the computational complexity of approximating solutions for Real algebraic formulae. Technical Report 858, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853-7501, August 1989.

[9] John Canny. A new algebraic method for robot motion planning and real geometry. In *Proceedings, 28th Annual IEEE Symposium on Foundations of Computer Science*, pages 39–48, October 1987.

[10] James Renegar. A faster pspace algorithm for deciding the existential theory of real closed fields. In *Proceedings, 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 291–295, October 1988.

[11] D. Kozen and C.K̃. Yap. Algebraic cell decomposition in nc. In *Proceedings of 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 515–521, October 1985.

[12] E. Dijkstra. *A Discipline of Programming*. Prentice-Hall, Inc., 1976.

[13] C. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–583, 1969.

[14] R. Floyd. Assigning meanings to programs. In *Proc. Symposium Applied Math*, volume 19, pages 19–32, 1967.

[15] S. Hantler and J. King. An introduction to proving the correctness of programs. *ACM Computing Surveys*, 8(3):331–353, 1976.

[16] J. King. Proving programs to be correct. *IEEE Transactions on Computers*, C-20(11):1331–1336, 1971.

[17] S. Owicki and D. Gries. An axiomatic proof technique for parallel programs. *Acta Informatica*, 6:319–340, 1976.

[18] Z. Manna and A. Pnueli. Temporal verification of concurrent programs. In R. Boyer and J. Strother, editors, *The Correctness Problem in Computer Science*. Academic Press, 1981.

[19] L. Deutsch. *An Interactive Program Verifier*. PhD thesis, Univ. Of Calif., Berkeley, 1973.

[20] D. Good, R. Cohen, C. Hoch, L. Hunter, and D. Har. Report on the language Gypsy. Technical Report ICSCA-CMP-10, The University of Texas, Austin, Texas, 1978.

[21] Manuel Blum and Sampath Kannan. Designing programs that check their work. In *Proceedings, 21st Annual ACM Symposium on Theory of Computing*, pages 86-97, May 1989.

[22] R. Apt and W. Roever. A proof system for communicating sequential processes. *ACM Transactions on Programming Languages and Systems*, 2(3):359–385, 1981.

[23] G.M. Levin and D. Gries. A proof technique for communicating sequential process. *Acta Information*, 15:281–302, 1981.

bitemOG76 S. Owicki and D. Gries. An axiomatic proof technique for parallel programs. *Acta Informatica*, 6:319–340, 1976.

[24] Dennis S. Arnon and Maurice Mignotte. On mechanical quantifier elimination for elementary algebra and geometry. *Journal of Symbolic Computation*, 5:237–259, 1988.

[25] A.K̃. Chandra and L.J̃. Stockmeyer. Alternation. In *Proceedings of 17th Annual IEEE Symposium on Foundations of Computer Science*, pages 98–108, October 1976.

[26] A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer. Alternation. *Journal of ACM*, 28(1):114–133, 1981.

[27] John H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29(2):274–301, 1984.

[28] Salman Azhar, Gary L. Peterson, and John H. Reif. Decision Algorithms for Multiplayer Non-Cooperative Games of Incomplete Information. In *Journal of Computers and Mathematics with Applications*, Vol. 43, Jan. 2002, pp 179-206.

[29] Salman Azhar, Gary L. Peterson, and John H. Reif. Lower Bounds for Multiplayer Non-Cooperative Games of Incomplete Information. In *Journal of Computers and Mathematics with Applications*, Volume 41, April 2001, pp 957-992.

[30] C.H. Papadimitriou. Games against nature. In *Proceedings of 24th IEEE Symposium on Foundations of Computer Science*, pages 446–450, October 1983.

[31] C.H. Papadimitriou. Games against nature. *Journal of Computer and System Sciences*, 31:288–301, 1985.

[32] L. Babai. Trading group theory for randomness. In *Proceedings, 17th Annual ACM Symposium on Theory of Computing*, pages 421–429, May 1985.

[33] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive protocols. In *Proceedings, 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, May 1985.

[34] S. Goldwasser M. Sipser. Public coins versus private coins in interactive proof systems. In *Proceedings, 18th Annual ACM Symposium on Theory of Computing*, May 1986.

[35] A. Shamir. Ip = pspace. In *Proceedings, 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 11–15, October 1990.

[36] A. Condon and R. Ladner. Probabilistic Game Automata. In *Journal of Computer and System Sciences*, June, 1988, Volume 36, Number 3, pages 452-489.

[37] L. Blum, M. Shub, and S. Smale, On a theory of Computation and Complexity over the Real Numbers: NP-Completeness, Recursive Functions and Universal Machines. In *Bulletin of American Mathematical Soceity*, July, 1989, volume 21, number 1, pages 1-46.

[38] John von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior.* Princeton University Press, Princeton, New Jersey, 3rd edition, 1953.

[39] H. Kuhn. *Extensive Games and the Problem of Information*, volume 2, pages 193–216. Princeton University Press, Princeton, New Jersey, 1953.

[40] E. Kohlberg and J.-F. Mertens. On the strategic stability of equilibria. *Econometrica*, 54:1003–1037, September 1986.

[41] J. Nash. Noncooperative games. *Annals of Mathematics*, 54:443–460, 1951.

[42] S. Kakutani. A generalization of brouwer's fixed point theorem. *Duke Mathematical Journal*, 8:457–458, 1944.

[43] B.D. Bernheim. Rationalizable strategic behavior. *Econometrica*, 52:1007–1028, July 1984.

[44] D.G. Pearce. Rationalizable strategic behavior and the problem of perfection. *Econometrica*, 52:1029–1050, 1984.

[45] Herve Moulin. *Game Theory for Social Sciences*, page 129 and 146. New York University Pres, New York, NY, 2nd edition, 1986.

[46] J. Grote. Global theory of games. *J. of Mathematical Economics*, 1(3):223–236, 1974.

[47] Andrew McLennan. Consistent conditional systems in noncooperative game theory. *International Journal of Game Theory*, 18, 1989.

[48] L. Csanky. Fast parallel matrix inversion algorithms. *SIAM J. Computing*, 5:618–623, 1976.

[49] M. Ben-Or, D. Kozen, and J.H. Reif. The complexity of elementary algebra and geometry. *Journal of Computer and System Sciences*, 32:251–264, 1986.

[50] M. Ben-Or, Ephraim Feig, D. Kozen, and P. Tiwari. A fast parallel algorithm for determining all roots of a polynomial with real roots. *SIAM Journal of Computing*, 17(6):1081–1092, December 1988.

[51] Andrew McLennan. Justifiable beliefs in sequential equilibrium. *Econometrica*, 53:889–904, July 1985.

[52] I.-K. Cho. A refinement of sequential equilibrium. *Econometrica*, 55:1367–1389, November 1987.