CHAPTER 2

The Complexity of Finding Nash Equilibria

Christos H. Papadimitriou

Abstract

Computing a Nash equilibrium, given a game in normal form, is a fundamental problem for Algorithmic Game Theory. The problem is essentially combinatorial, and in the case of two players it can be solved by a pivoting technique called the Lemke–Howson algorithm, which however is exponential in the worst case. We outline the recent proof that finding a Nash equilibrium is complete for the complexity class PPAD, even in the case of two players; this is evidence that the problem is intractable. We also introduce several variants of succinctly representable games, a genre important in terms of both applications and computational considerations, and discuss algorithms for correlated equilibria, a more relaxed equilibrium concept.

2.1 Introduction

Nash's theorem – stating that every finite game has a mixed Nash equilibrium (Nash, 1951) – is a very reassuring fact: Any game can, in principle, reach a quiescent state, one in which no player has an incentive to change his or her behavior. One question arises immediately: Can this state be reached in practice? Is there an *efficient algorithm* for finding the equilibrium that is guaranteed to exist? This is the question explored in this chapter.

But why should we be interested in the issue of computational complexity in connection to Nash equilibria? After all, a Nash equilibrium is above all a conceptual tool, a prediction about rational strategic behavior by agents in situations of conflict – a context that is completely devoid of computation.

We believe that this matter of computational complexity is one of central importance here, and indeed that the algorithmic point of view has much to contribute to the debate of economists about solution concepts. The reason is simple: If an equilibrium concept is not efficiently computable, much of its credibility as a prediction of the behavior of rational agents is lost – after all, there is no clear reason why a group of agents cannot be simulated by a machine. Efficient computability is an important modeling perequisite for solution concepts. In the words of Kamal Jain, "If your laptop cannot find it, neither can the market."[1]

2.1.1 Best Responses and Supports

Let us thus define NASH to be the following computational problem: Given a game in strategic form, find a NASH equilibrium. Since NASH calls for the computation of a real-valued distribution for each player, it seems *primae facie* to be a quest in continuous mathematics. However, a little thought reveals that the task is *essentially combinatorial.*

Recall that a mixed strategy profile is a NASH equilibrium if the mixed strategy of each player is a *best response* to the mixed strategies of the rest; that is, it attains the maximum possibly utility among all possible mixed strategies of this player. The following observation is useful here (recall that the *support* of a mixed strategy is the set of all pure strategies that have nonzero probability in it).

• **Theorem 2.1** *A mixed strategy is a best response if and only if all pure strategies in its support are best responses.*

To see why, assume for the sake of contradiction that a best response mixed strategy contains in its support a pure strategy that is not itself a best response. Then the utility of the player would be improved by decreasing the probability of the worst such strategy (increasing proportionally the remaining nonzero probabilities to fill the gap); this contradicts the assumption that the mixed strategy was a best response. Conversely, if all strategies in all supports are best responses, then the strategy profile combination must be a NASH equilibrium.

This simple fact reveals the subtle nature of a mixed NASH equilibrium: Players combine pure best response strategies (instead of using, for the same utility, a single pure best response) in order to create for other players a range of best responses that will sustain the equilibrium!

Example 2.2 Consider the *symmetric game* with two players captured by the matrix

$$A = \begin{pmatrix} 0 & 3 & 0 \\ 0 & 0 & 3 \\ 2 & 2 & 2 \end{pmatrix}$$

A game with two players can be represented by two matrices (A, B) (hence the term *bimatrix game* often used to describe such games), where the rows of A are the strategies of Player 1 and the columns of A are the strategies of Player 2, while the entries are the utilities of Player 1; the opposite holds for matrix B. A bimatrix game is called *symmetric* if $B = A^T$; i.e., the two players have the same set of strategies, and their utilities remain the same if their roles are reversed.

In the above symmetric game, consider the equilibrium in which both players play the mixed strategy $(0, 1/3, 2/3)$. This is a *symmetric* NASH equilibrium,

[1] One may object to this aphorism on the basis that in markets agents work in *parallel*, and are therefore more powerful than ordinary algorithms; however, a little thought reveals that parallelism cannot be the cure for exponential worst case.

because both players play the same mixed strategy. (A variant of NASH's proof establishes that every symmetric game, with any number of players, has a symmetric equilibrium – it may also have nonsymmetric ones.) We can check whether it is indeed an equilibrium, by calculating the utility of each strategy, assuming the opponent plays (0, 1/3, 2/3): The utilities are 1 for the first strategy, and 2 for the other two. Thus, every strategy in the support (i.e., either of strategies 2 and 3) is a best response, and the mixed strategy is indeed a NASH equilibrium. Note that, from Player 1's point of view, playing just strategy 2, or just strategy 3, or any mixture of the two, is equally beneficial to the equilibrium mixed strategy (0, 1/3, 2/3). The only advantage of following the precise mix suggested by the equilibrium is that it motivates the other player to do the same.

Incidentally, in our discussion of NASH equilibria in this chapter, we shall often use the simpler two-player case to illustrate the ideas. Unfortunately, the main result of this section says that two-player games are not, in any significant sense, easier than the general problem.

It also follows from these considerations that finding a mixed NASH equilibrium means finding the right supports: Once one support for each player has been identified, the precise mixed strategies can be computed by solving a system of algebraic equations (in the case of two players, linear equations): For each player i we have a number of variables equal to the size of the support, call it k_i, one equation stating that these variables add to 1, and $k_i - 1$ others stating that the k_i expected utilities are equal. Solving this system of $\sum_i k_i$ equations in $\sum_i k_i$ unknowns yields k_i numbers for each player. If these numbers are real and nonnegative, and the utility expectation is maximized at the support, then we have discovered a mixed NASH equilibrium.

In fact, if in the two-player case the utilities are integers (as it makes sense to assume in the context of computation), then the probabilities in the mixed NASH equilibrium will necessarily be rational numbers, since they constitute the solution of a system of linear equations with integer coefficients. This is not true in general: NASH's original paper (1951) includes a beautiful example of a three-player poker game whose only NASH equilibrium involves irrational numbers.

The bottom line is that *finding a NASH equilibrium is a combinatorial problem:* It entails identifying an appropriate support for each player. Indeed, most algorithms proposed over the past half century for finding NASH equilibria are combinatorial in nature, and work by seeking supports. Unfortunately, none of them are known to be efficient – to always succeed after only a polynomial number of steps.

2.2 Is the NASH Equilibrium Problem NP-Complete?

Computer scientists have developed over the years notions of complexity, chief among them *NP-completeness* (Garey and Johnson, 1979), to characterize computational problems which, just like NASH and SATISFIABILITY,[2] seem to resist efficient solution. Should we then try to apply this theory and prove that NASH is NP-complete?

[2] Recall that SATISFIABILITY is the problem that asks, given a Boolean formula in conjunctive normal form, to find a satisfying truth assignment.

It turns out that NASH is a very different kind of intractable problem, one for which NP-completeness is not an appropriate concept of complexity. The basic reason is that *every game* is guaranteed to have a NASH equilibrium. In contrast, in a typical NP-complete problem such as SATISFIABILITY, the sought solution may or may not exist. NP-complete problems owe much of their difficulty, and their susceptibility to NP-completeness reductions, to precisely this dichotomy.[3] For, suppose that NASH is NP-complete, and there is a reduction from SATISFIABILITY to NASH. This would entail an efficiently computable function f mapping Boolean formulae to games, and such that, for every formula ϕ, ϕ is satisfiable if and only if any NASH equilibrium of $f(\phi)$ satisfies some easy-to-check property Π. But now, given any unsatisfiable formula ϕ, we could guess a NASH equilibrium of $f(\phi)$, and check that it does not satisfy Π: This implies NP = coNP!

Problems such as NASH for which a solution is guaranteed to exist require much more specialized and subtle complexity analysis – and the end diagnosis is necessarily less severe than NP-completeness (see Beame et al., 1998; Johnson et al., 1988; Papadimitriou, 1994 for more on this subject).

2.2.1 NASH *vs* Brouwer

In contemplating the complexity of NASH, a natural first reaction is to look into NASH's proof (1951) and see precisely how existence is established – with an eye towards making this existence proof "constructive." Unfortunately this does not get us very far, because NASH's proof relies on *Brouwer's fixpoint theorem,* stating that every continuous function f from the n-dimensional unit ball to itself has a fixpoint: a point x such that $f(x) = x$. NASH's proof is a clever reduction of the existence of a mixed equilibrium to the existence of such a fixpoint. Unfortunately, Brouwer's theorem is well-known for its nonconstructive nature, and finding a Brouwer fixpoint is known to be a hard problem (Hirsch et al., 1989; Papadimitriou, 1994) – again, in the specialized sense alluded to above, since a solution is guaranteed to exist here also.

Natural next question: Is there a reduction in the opposite direction, one establishing that NASH is precisely as hard as the known difficult problem of finding a Brouwer fixpoint? The answer is "yes," and this is in fact a useful alternative way of understanding the main result explained in this chapter.[4]

2.2.2 NP-Completeness of Generalizations

As we have discussed, what makes NP-completeness inappropriate for NASH is the fact that NASH equilibria always exist. If the computational problem NASH is twisted

[3] But how about the traveling salesman problem? Does it not always have a solution? It does, but this solution (the optimum tour) is hard to verify, and so the TSP is not an appropriate comparison here. To be brought into the realm of NP-completeness, optimization problems such as the TSP must be first transformed into decision problems of the form "given a TSP instance and a bound B, does a tour of length B or smaller exist?" This problem is much closer to SATISFIABILITY.

[4] This may seem puzzling, as it seems to suggest that Brouwer's theorem is also of a combinatorial nature. As we shall see, in a certain sense indeed it is.

in any one of several simple ways that deprive it from its existence guarantee, NP-completeness comes into play almost immediately.

Theorem 2.3 (Gilboa and Zemel, 1989) *The following are NP-complete problems, even for symmetric games: Given a two-player game in strategic form, does it have*

- *at least two NASH equilibria?*
- *a NASH equilibrium in which player 1 has utility at least a given amount?*
- *a NASH equilibrium in which the two players have total utility at least a given amount?*
- *a NASH equilibrium with support of size greater than a given number?*
- *a NASH equilibrium whose support contains strategy s?*
- *a NASH equilibrium whose support does not contain strategy s?*
- *etc., etc.*

A simple proof, due to (Conitzer and Sandholm, 2003), goes roughly as follows: Reduction from SATISFIABILITY. It is not hard to construct a symmetric game whose strategies are all literals (variables and their negations) and whose NASH equilibria are all truth assignments. In other words, if we choose, for each of the n variables, either the variable itself or its negation, and play it with probability $\frac{1}{n}$, then we get a symmetric NASH equilibrium, and all NASH equilibria of the game are of this sort. It is also easy to add to this game a new pure NASH equilibrium (d, d), with lower utility, where d (for "default") is a new strategy. Then you add new strategies, one for each clause, such that the strategy for clause C is attractive, when a particular truth assignment is played by the opponent, only if all three literals of C are contradicted by the truth assignment. Once a clause becomes attractive, it destroys the assignment equilibrium (via other strategies not detailed here) and makes it drift to (d, d). It is then easy to establish that the NASH equilibria of the resulting game are precisely (d, d) plus all satisfying truth assignments. All the results enumerated in the statement of the theorem, and more, follow very easily.

2.3 The Lemke–Howson Algorithm

We now sketch the Lemke–Howson algorithm, the best known among the combinatorial algorithms for finding a NASH equilibrium (this algorithm is explained in much more detail in the next chapter). It works in the case of two-player games, by exploiting the elegant combinatorial structure of supports. It constitutes an alternative proof of NASH's theorem, and brings out in a rather striking way the complexity issues involved in solving NASH. Its presentation is much simpler in the case of symmetric games. We therefore start by proving a basic complexity result for games: looking at symmetric games is no loss of generality.