Princeton Univ. F'21 COS 521: Advanced Algorithm Design

Lecture 16: Games, Min-Max and Equilibria, and PPAD

Lecturer: *Matt Weinberg* Last updated: November 4, 2021

Economic and game-theoretic reasoning —specifically, how agents respond to economic incentives as well as to each other's actions– has become increasingly important in algorithm design. Examples: (a) Protocols for networking have to allow for sharing of network resources among users, companies etc., who may be mutually cooperating or competing. (b) Algorithm design at Google, Facebook, Netflix etc.—what ads to show, which things to recommend to users, etc.—not only has to be done using objective functions related to economics, but also with an eye to how users and customers *change* their behavior in response to the algorithms and to each other.

Algorithm design mindful of economic incentives and strategic behavior is studied in a field called *Algorithmic Game Theory.* (See the book by Nisan et al., or many excellent lecture notes on the web.)

1 Game Theory

In the 1930s, polymath John von Neumann (professor at IAS, now buried in the cemetery close to downtown) was interested in applying mathematical reasoning to understand strategic interactions among people —or for that matter, nations, corporations, political parties, etc. He was a founder of *game theory*, which models rational choice in these interactions as maximization of some payoff function.

A starting point of this theory is the *zero-sum* game. There are two players, 1 and 2, where 1 has a choice of m possible moves, and 2 has a choice of n possible moves. When player 1 plays his ith move and player 2 plays her jth move, the outcome is that player 1 pays A_{ij} to player 2. Thus the game is completely described by an $m \times n$ *payoff* matrix.

-	scissors	paper	rock
rock	1	−1	0
paper	−1	0	1
scissors	0	1	−1

Figure 1: Payoff matrix for Rock/Paper/Scissor

This setting is called *zero sum* because what one player wins, the other loses. By contrast, war (say) is a setting where both parties may lose material and men. Thus their combined worth at the end may be lower than at the start. (Aside: An important stimulus for development of game theory in the 1950s was the US government's desire to behave

"strategically "in matters of national defence, e.g. the appropriate tit-for-tat policy for waging war —whether nuclear or conventional or cold.)

von Neumann was interested in a notion of equilibrium. In physics, chemistry etc. an equilibrium is a stable state for the system that results in no further change. In game theory it is a pair of strategies g_1, g_2 for the two players such that each is the optimum response to the other.

Let's examine this for zero sum games. If player 1 announces he will play the ith move, then the *rational* move for player 2 is the move j that maximises A_{ij}. Conversely, if player 2 announces she will play the jth move, player 1 will respond with move i' that minimizes $A_{i'j}$. In general, there may be no *equilibrium* in such announcements: the response of player 1 to player 2's response to his announced move i will not be i in general:

$$\min_i \max_j A_{ij} \neq \max_j \min_i A_{ij}.$$

In fact there is no such equilibrium in Rock/paper/scissors either, as every child knows.

von Neumann realized that this lack of equilibrium disappears if one allows players' announced strategy to be a *distribution* on moves, a so-called *mixed* strategy. Player 1's distribution is $x \in \Re^m$ satisfying $x_i \geq 0$ and $\sum_i x_i = 1$; Player 2's distribution is $y \in \Re^n$ satisfying $y_j \geq 0$ and $\sum_j y_j = 1$. Clearly, the expected payoff from Player 1 to Player 2 then is $\sum_{ij} x_i A_{ij} y_j = x^T A y$.

But has this fixed the problem about nonexistence of equilibrium? If Player 1 announces first the payoff is $\min_x \max_y x^T A y$ whereas if Player 2 announces first it is $\max_y \min_x x^T A y$. The next theorem says that it doesn't matter who announces first; neither player has an incentive to change strategies after seeing the other's announcement.

Theorem 1 (Famous Min-Max Theorem of Von Neumann)**.** $\min_x \max_y x^T A y = \max_y \min_x x^T A y$.

Turns out this result is a simple consequence of LP duality and is equivalent to it. You will explore it further in the homework.

What if the game is not zero sum? Defining an equilibrium for it was an open problem until John Nash at Princeton managed to define it in the early 1950s; this solution is called a Nash equilibrium.

2 Nonzero sum games and Nash equilibria

Recall that a 2-player game is *zero sum* if the amount won by one player is the same as the amount lost by the other. Today we relax this. Thus if player 1 has n possible actions and player 2 has m, then specifying the game requires two a $n \times m$ matrices A, B such that when they play actions i, j respectively then the first player wins A_{ij} and the second wins B_{ij}. (For zero sum games, $A_{ij} = -B_{ij}$.)

A Nash equilibrium is defined similarly to the equilibrium we discussed for zero sum games: a pair of strategies, one for each player, such that each is the optimal response to the other. In other words, if they both announce their strategies, neither has an incentive to deviate from his/her announced strategy. The equilibrium is *pure* if the strategy consists of deterministically playing a single action.

Example 1 (Prisoners' Dilemma)**.** *This is a classic example that people in myriad disciplines have discussed for over six decades. Two people suspected of having committed a crime have been picked up by the police. In line with usual practice, they have been placed in separate cells and offered the standard deal: help with the investigation, and you'll be treated with leniency. How should each prisoner respond: Cooperate (i.e., stick to the story he and his accomplice decided upon in advance), or Defect (rat on his accomplice and get a reduced term)?*

Let's describe their incentives as a 2×2 *matrix, where the first entry describes payoff for the player whose actions determine the row. If they both cooperate, the police can't prove*

	Cooperate	Defect
Cooperate	3, 3	0, 4
Defect	4, 0	1, 1

much and they get off with fairly light sentences after which they can enjoy their loot (payoff of 3*). If one defects and the other cooperates, then the defector goes scot free and has a high payoff of* 4 *whereas the other one has a payoff of* 0 *(long prison term, plus anger at his accomplice).*

The only pure Nash equilibrium is (Defect, Defect), with both receiving payoff 1*. In every other scenario, the player who's cooperating can improve his payoff by switching to Defect. This is much worse for both of them than if they play (Cooperate, Cooperate), which is also the social optimum —where the sum of their payoffs is highest at* 6*—is to cooperate. Thus in particular the social optimum solution is not a Nash equilibrium. ((OK, we are talking about criminals here so maybe social optimum is (Defect, Defect) after all. But read on.)*

One can imagine other games with similar payoff structure. For instance, two companies in a small town deciding whether to be polluters or to go green. Going green requires investment of money and effort. If one does it and the other doesn't, then the one who is doing it has incentive to also become a polluter. Or, consider two people sharing an office. Being organized and neat takes effort, and if both do it, then the office is neat and both are fairly happy. If one is a slob and the other is neat, then the neat person has an incentive to become a slob (saves a lot of effort, and the end result is not much worse).

Such games are actually ubiquitous if you think about it, and it is a miracle that humans (and animals) cooperate as much as they do. Social scientists have long pondered how to cope with this paradox. For instance, how can one change the game definition (e.g. a wise governing body changes the payoff structure via fines or incentives) so that cooperating with each other —the socially optimal solution—becomes a Nash equilibrium? The game can also be studied via the repeated game *interpretation, whereby people realize that they participate in repeated games through their lives, and playing nice may well be a Nash equilibrium in that setting. As you can imagine, many books have been written.* □

Example 2 (Chicken)**.** *This dangerous game was supposedly popular among bored teenagers in American towns in the 1950s (as per some classic movies). Two kids would drive their cars at high speed towards each other on a collision course. The one who swerved away first to avoid a collision was the "chicken."How should we assign payoffs in this game? Each player has two possible actions,* Chicken *or* Dare*. If both play Dare, they wreck their cars and risk injury or death. Lets call this a payoff of* 0 *to each. If both go Chicken, they both*

live and have not lost face, so let's call it a payoff of 5 *for each. But if one goes Chicken and the other goes Dare, then the one who went Dare looks like the tough one (and presumably attracts more dates), whereas the Chicken is better of being alive than dead but lives in shame. So we get the payoff table:*

	Chicken	Dare
Chicken	5, 5	1, 6
Dare	6, 1	0, 0

This has two pure Nash equilibria: (Dare, Chicken) and (Dare, Chicken). We may think of this as representing two types of behavior: the reckless type may play Dare and the careful type may play Chicken.

Note that the socially optimal solution —both players play chicken, which maximises their total payoff—is not a Nash equilibrium.

Many games do not have any pure Nash equilibrium. Nash's great insight during his grad school years in Princeton was to consider what happens if we allow players to play a *mixed* strategy, which is a probability distribution over actions. An equilibrium now is a pair of mixed strategies x, y such that each strategy is the optimum response (in terms of maximising expected payoff) to the other.

Theorem 2 (Nash 1950)**.** *For every pair of payoff matrices A, B there exists a mixed equilibrium.*

(In fact, Wilson's theorem from 1971 says that for random matrices A, B, the number of equilbria is odd with high probability.)

Unfortunately, Nash's proof doesn't yield an efficient algorithm for computing an equilibrium: when the number of possible actions is n, computation may require $\exp(n)$ time. Recent work has shown that this may be inherent: computing Nash equilibria is PPAD-complete (Daskalakis, Goldberg, Papadimitriou '05, Chen and Deng '06).

The Chicken game has a mixed equilibrium: play each of Chicken and Dare with probability 1/2. This has expected payoff $\frac{1}{4}(5+1+6+0) = 3$ for each, and a simple calculation shows that neither can improve his payoff against the other by changing to a different strategy.

3 Algorithms for Nash Equilibria

Nash's proof of existence is non-constructive: it relies on Brouwer's fixed point theorem (every continuous function f from a compact, convex space to itself has a fixed point: some x such that $f(x) = x$). The proof essentially constructs a continuous function from the unit simplex to itself where all fixed points are Nash equilibria.

As discussed above, no known algorithms for Nash Equilibria run in poly-time, and it is widely believed that there are no poly-time algorithms for PPAD (hence, there should be no poly-time algorithms for Nash). Here, we'll discuss two algorithms for Nash: a QPTAS, and an exponential-time exact algorithm. We'll assume that $A_{ij} \in [-1, 1]$ for all i, j.

3.1 QPTAS: Lipton-Markakis-Mehta

First, we'll describe the QPTAS. Before getting started, we need to define what we mean by an approximation.

Definition 1. *An ϵ-Nash equilibrium is a strategy x for the row player and y for the column player such that $xAy \geq \max_i \langle A_i, y\rangle - \epsilon$, and $yBx \geq \max_i \langle B_i, x\rangle - \epsilon$. That is, both players are best responding up to an additive ϵ.*

The crux of the algorithm is the following simple lemma:

Lemma 3 (Lipton-Markakis-Mehta)**.** *For a two-player game with $n \times n$ payoff matrices A and B, there exists an ϵ-Nash where each player uses a strategy that uniformly samples from a multi-set of size $O(\log n/\epsilon^2)$.*

Proof. We know that a Nash equilibrium exists (we don't know what it is, but it exists), call the two strategies x and y. Consider randomly sampling k strategies from x (with repetition), and ditto for y. Call these multisets X and Y, respectively, and let x_i^*, y_i^* denote the number of times that i appears in X (respectively, Y) divided by k. We want k to be large enough so that the following events all hold with positive probability:

- For all i, $|A_i \cdot y - A_i \cdot y^*| \leq \epsilon$, for all i.
- For all i, $|B_i \cdot x - B_i \cdot x^*| \leq \epsilon$, for all i.
- $|xAy - x^*Ay| \leq \epsilon$.
- $|yBx - y^*Bx| \leq \epsilon$.

If the first event holds, we can conclude that the performance of any action i against y is very similar to its performance against y^* (within ϵ). This immediately lets us conclude that any mixed strategy has simliar performance against y and y^* as well (within ϵ), which then immediately lets us conclude that x is a 2ϵ-best response to y^*. The third event lets us conclude that x^* is a 3ϵ-best response.

$$x^*Ay^* \geq x^*Ay - \epsilon \geq xAy - 2\epsilon \geq \max_i\{A_i \cdot y\} - 2\epsilon \geq \max_i\{A_i \cdot y^*\} - 3\epsilon.$$

The same reasoning holds for the column player. So when all four events happen, we have a 3ϵ-Nash. Now we just need to figure out how big to set k. Note that the expected value of $A_i \cdot y^*$ is exactly $A_i \cdot y$. Ditto for x^*Ay and xAy. Notice also that $A_i \cdot y^*$ is a sum of independent random variables. So we just need to take enough samples to apply a Chernoff bound and union bound over $O(n)$ events. This is exactly $O(\log n/\epsilon^2)$. □

So we know that an equilibrium of this form exists, but we don't know the distribution we're supposed to sample from: if we knew that then we'd already have an equilibrium. But the trick is that we don't need to actually do the sampling, we can just exhaust over all possible distributions of this form. There are at most $n^{\log n/\epsilon^2}$ different sets of size $\log n/\epsilon^2$, so there are also at most this many distributions that are uniform over multisets of size $\log n/\epsilon^2$. So the algorithm is super simple: just exhaust over all possibilities until we find an ϵ-Nash, and we're guaranteed to succeed by the lemma above.

Side note: it was only very recently shown by Rubinstein that it is extremely unlikely that a faster algorithm exists to find an ϵ-Nash. Specifically, assuming that no subexponential-time algorithms exist for PPAD, no faster algorithm exists (this is kind of like the "exponential time hypothesis for PPAD").

3.2 Exact Nash: Lemke-Howson

Now, let's go over how to compute Nash exactly. There are multiple exponential time algorithms for this (including simpler ones than described below), but the Lemke-Howson algorithm demonstrates the most reproducible ideas. To make the exposition simpler, we'll consider *symmetric* games, where $B = A^T$ (this is w.l.o.g. as we'll see on the homework). We'll also consider symmetric equilibria (where the row and column player plays the same strategy, also w.l.o.g.). Consider the following polytope:

$$\begin{aligned} &\langle A_i, x\rangle \leq 1, \quad \forall i. \\ &x_i \geq 0, \quad \forall i. \end{aligned}$$

The first constraints ensure that strategy i doesn't yield payoff more than $1/|x|_1$ against $x/|x|_1$. The second constraints make sure that $x/|x|_1$ is an actual probability distribution. It's unclear what to make of a point in this polytope, and actually we don't want to optimize anything over it. What's interesting is the following: we take a point x in this polytope such that for all i, at least one the equations is tight (i.e. either $\langle A_i, x\rangle = 1$ or $x_i = 0$), then this is a Nash.

To see this, note that if $\langle A_i, x\rangle = 1$, then i is a best response to $x/|x|_1$. Note that if $x_i = 0$, then i is not played. So this means that for all i, either i is not played, or i is a best response, and x only plays best responses to x and therefore x is a symmetric Nash.

So now we just need to find a vertex of this polytope that satisfies these properties. The Lemke-Howson algorithm is a *pivot rule* that does this. We start with the vertex $\langle 0, \ldots, 0\rangle$, which satisfies all the bottom inequalities with equality and none of the top ones. Then we pick an arbitrary constraint and relax it, keeping the others fixed, until we hit a new tight constraint. This is called the "pivot."

To see one concrete example, for our first pivot we might choose to relax $x_1 = 0$. So we could maximize x_1 over all x such that $x_i = 0$ for all $i \geq 2$ (keep all other tight constraints tight), and $\langle A_i, x\rangle \leq 1, \forall i$. The solution will have one of the other constraints tight. At this point, we again have n tight constraints. If each coordinate is "covered" exactly once (i.e. if we now have $\langle A_1, x\rangle = 1$ then we're done. If not, then there's a *double-covered* coordinate (because we instead had $\langle A_i, x\rangle = 1$ for $i \neq 1$). So relax the old tight constraint of the double-covered coordinate and repeat.

There are at most $\binom{2n}{n}$ total vertices of the polytope that we'll ever reach, so the algorithm definitely terminates in exponential time. We just need to make sure it terminates at a Nash and doesn't cycle back to a previous vertex. But observe the following:

- Each vertex on the path has exactly two neighbors (because the double-covered coordinate has two constraints that can be relaxed). So we cannot repeat a vertex on the path before terminating (except possibly the origin).

- The origin has n neighbors, where each of the n coordinates are uncovered.
- Finally, observe that if we start by uncovering 1, then the path will terminate as soon as we re-cover 1. This is because all coordinates except for 1 remain covered the entire time (perhaps double-covered). So we cannot return to the origin through any of its $n-1$ neighbors, as they cover 1. We also cannot return through its initial neighbor, as this would revisit an internal node.

Therefore, the algorithm must actually terminate in a Nash, and not revisit the origin. Sidenote: this algorithm was the original motivation for the class PPAD.

4 Definition of PPAD

Typically, when we think about complexity classes, we think about classes like NP. For problems in NP, such as independent set, we can ask "does there exist an independent set of size k?" The answer could be yes or no. If the answer is yes, we can also ask questions like "what is an independent set of size k?"

The problem of finding a Nash equilibrium doesn't quite fit into this language. For example, we could start by asking "does this game have a Nash equilibrium?", but the answer is always yes. We can still ask "what is a Nash equilibrium?", but NP doesn't seem to be the right complexity class to capture this.[1] There is a broad class TFNP ("Total Function Non-Deterministic Polynomial") of problems where an answer is guaranteed to exist, and also can be verified in polynomial time. Finding a Nash equilibrium is an example of such a problem.

Within TFNP, several complexity classes have been defined, roughly corresponding to various non-constructive proofs. For example:

- PPAD (every graph with a source has a sink): You are given as input an exponentially large directed graph, G, where each node has in-degree and out-degree at most one. G is defined by two poly-sized circuits p and s. p takes as input a node v and outputs its in-neighbor $p(v)$ in G (if one exists, otherwise it outputs 0). s takes as input a node v and outputs its out-neighbor $s(v)$ in G. You are also given as input a node v, such that $p(v) = 0$. Find another node w such that $p(w) = 0$, or $s(w) = 0$. Or, find two nodes w, u such that $p(w) = u$ but $s(u) \neq w$.
- PPP (pigeonhole principle): You are given as input a circuit, $f : \{0,1\}^n \to \{0,1\}^n$, find an x such that $f(x) = 0$, or an x, y such that $f(x) = f(y)$.
- PLS (every function on a finite domain has a local minimum): You are given as input a function $f : \{0,1\}^n \to \{0, 1, \ldots, 2^n\}$. Find an input v such that $f(v) \leq f(w)$ for all neighbors w of v (w is a neighbor of v if they disagree on exactly one bit).

[1] You could try asking decision problems of the form "does a Nash equilibrium exist where Player 1 plays strategy 1 with probability at least 1/2?" and doing a binary search. Most variants of this problem that I'm aware of are NP-hard. But, because games have multiple Nash equilibria, it's conceivable that one could find an efficient algorithm to find one Nash equilibrium without ever learning whether another Nash equilibrium of a particular form exists.

- Lots of others: pick your favorite non-constructive proof from math.

Bibliography

1. Algorithmic Game Theory. Nisan, Roughgarden, Tardos, Vazirani (eds.), Cambridge University Press 2007.

2. The mathematics of traffic in networks. Frank Kelly. In *Princeton Companion to Mathematics* (T. Gowers, Ed.). PU Press 2008.

3. Settling the Complexity of 2-Player Nash Equilibrium. X. Chen and X. Deng. IEEE FOCS 2006.

4. The Complexity of computing a Nash Equilibrium. C. Daskalakis, P. W. Goldberg, C. H. Papadimitriou. SIAM Journal on Computing, 2009.

5. Settling the Complexity of Computing Two-Player Approximate Nash Equilibria. A. Rubinstein. Foundations of Computer Science, 2016.

6. Playing Large Games Using Simple Strategies. R. J. Lipton, E. Markakis, A. Mehta. Electronic Commerce, 2003.

7. C. E. Lemke, J. T. Howson. Equilibrium Points of Bimatrix Games. SIAM Journal on Applied Mathematics, 1964.