TTIC 31260 Algorithmic Game Theory

04/01/24

More on Nash equilibria: concepts, complexity, and algorithms

Your guide:

Avrim Blum

[Readings: Ch. 2.1-2.4 of AGT book]

First: Completing the proof for FTPL

Recall FTPL and main theorem

FTPL:

- Choose $c_0 \sim \left[0, \frac{2}{\epsilon}\right]^m$.
- Choose $a_t^{FTPL} = \arg\min_a \langle c_0 + c_1 + \cdots + c_{t-1}, a \rangle$

Theorem: Assume $||c||_1 \leq 1$ for all $c \in \mathcal{C}$, and the L_1 diameter of $\mathcal{A}$ is D. If $c_0 \sim \left[0, \frac{2}{\epsilon}\right]^m$, then

$$E[Regret] \leq D\left(\frac{\epsilon T}{2} + \frac{1}{\epsilon}\right).$$

Setting $\epsilon = \sqrt{\frac{2}{T}}$ gives expected regret $\leq D\sqrt{2T}$.

Recall the analysis structure

FTPL:

- Choose $c_0 \sim \left[0, \frac{2}{\epsilon}\right]^m$.
- Choose $a_t^{FTPL} = \arg\min_a \langle c_0 + c_1 + \cdots + c_{t-1}, a \rangle$

BTPL:

- Choose $c_0 \sim \left[0, \frac{2}{\epsilon}\right]^m$.
- Choose $a_t^{BTPL} = \arg\min_a \langle c_0 + c_1 + \cdots + c_t, a \rangle$

Show:

- Difference in expected cost $\leq TD(\epsilon/2)$, and
- Expected regret of BTPL is $\leq D/\epsilon$.

Expected regret of BTPL

Define:

- $a_t^{BTPL} = \arg\min_a \langle c_0 + c_1 + \cdots + c_t, a \rangle$
- $a_t^{BTL} = \arg\min_a \langle c_1 + \cdots + c_t, a \rangle$

By analysis last time, we know that for any c_0:

$$\langle c_0, a_0^{BTPL} \rangle + \langle c_1, a_1^{BTPL} \rangle + \cdots + \langle c_T, a_T^{BTPL} \rangle \le \langle c_0 + \cdots + c_T, a_T^{BTPL} \rangle$$

Also, RHS $\le \langle c_0 + \cdots + c_T, a_T^{BTL} \rangle$.

Move c_0 terms to RHS, get:

$$\langle c_1, a_1^{BTPL} \rangle + \cdots + \langle c_T, a_T^{BTPL} \rangle \le \langle c_1 + \cdots + c_T, a_T^{BTL} \rangle + \langle c_0, a_T^{BTL} - a_0^{BTPL} \rangle$$

Since $\mathcal{A}$ has L_1-diameter at most D and each coordinate of c_0 has expected value $1/\epsilon$, we get $E[regret] \le D/\epsilon$.

Now to today's material

One more interesting game

"Ultimatum game":

- Two players "Splitter" and "Chooser"
- 3rd party puts $10 on table.
- Splitter gets to decide how to split between himself and Chooser.
- Chooser can accept or reject.
- If reject, money is burned.

One more interesting game

"Ultimatum game": E.g., with $4

Splitter: how much to offer chooser

Chooser: how much to accept

	1	2	3
1	(1,3)	(2,2)	(3,1)
2	(0,0)	(2,2)	(3,1)
3	(0,0)	(0,0)	(3,1)

Stackelberg leader strategies

Strategy such that if you announce it and opponent best-responds to you, you are best off.

Splitter: how much to offer chooser

Chooser: how much to accept

	1	2	3
1	(1,3)	(2,2)	(3,1)
2	(0,0)	(2,2)	(3,1)
3	(0,0)	(0,0)	(3,1)

Stackelberg leader strategies

Strategy such that if you announce it and opponent best-responds to you, you are best off.

Need not be a Nash equilibrium.

	Compete	Leave
Price high	(3,3)	(6,1)
Price low	(2,0)	(4,1)

Stackelberg leader strategies

Can solve efficiently. Say we're row player:

- For each column j, solve for p to maximize our expected gain s.t. j is best-response.
- Choose best.

	Compete	Leave
Price high	(3,3)	(6,1)
Price low	(2,0)	(4,1)

Stackelberg leader strategies

Can solve efficiently. Say we're row player:

- For each column j, solve for p to maximize our expected gain s.t. j is best-response.
- Choose best.
  - For each j, solve for $p_1, \ldots, p_n \geq 0, \sum_i p_i = 1$, to maximize our gain $\sum_i p_i R_{ij}$ subject to:
    - For each j', $\sum_i p_i C_{ij} \geq \sum_i p_i C_{ij'}$ (the column player prefers j)

Hardness of computing Nash equilibria

Looking at 2-player n-action games.

2 types of results:

- NP-hardness for NE with special properties [Gilboa-Zemel] [Conitzer-Sandholm]
  - Is there one with payoff at least v for row?
  - Is there one using row #1?
  - Is there more than one?
  - …
- PPAD-hardness for finding any NE. [Chen-Deng][Daskalakis-Goldberg-Papadimitriou]

Hardness of computing Nash equilibria

NP-hardness for NE with special properties

Basic idea:

- Given 3-SAT formula F, create a game with one row for each literal, variable, & clause.
- Also a default attractor action f. $C = R^T$.
- Somehow set things up so that except for (f,f), all NE must correspond to satisfying assignments.

What about just finding some NE?

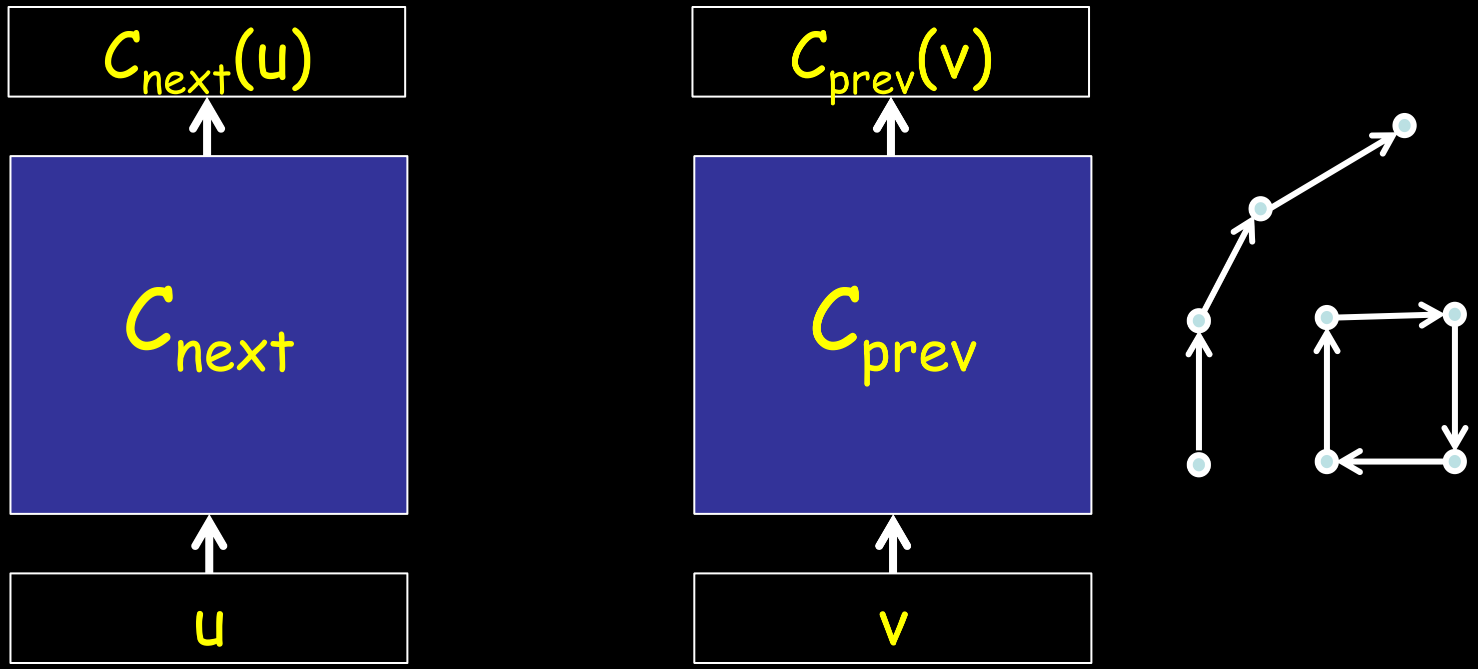
This is "PPAD" hard.

What's that?

What about just finding some NE?

Consider the following problem:

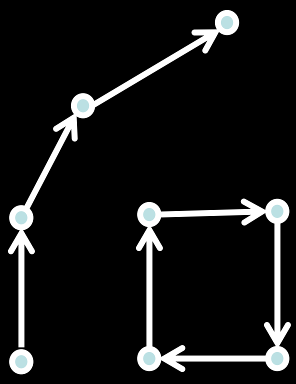
- Given two circuits C_{next} and C_{prev}, each with n-bit input, n-bit output.
- View as defining directed graph G:
  $u \rightarrow v$ iff $C_{next}(u)=v$ and $C_{prev}(v)=u$. (indeg ≤ 1, outdeg ≤ 1)



What about just finding some NE?

Consider the following problem:

- Given two circuits C_{next} and C_{prev}, each with n-bit input, n-bit output.
- View as defining directed graph G:
  $u \to v$ iff $C_{next}(u)=v$ and $C_{prev}(v)=u$. (indeg ≤ 1, outdeg ≤ 1)
- Say v "unbalanced" if indeg(v) $\neq$ outdeg(v).
- If 0^n is unbalanced, then find another unbalanced node. (must exist)



This is PPAD

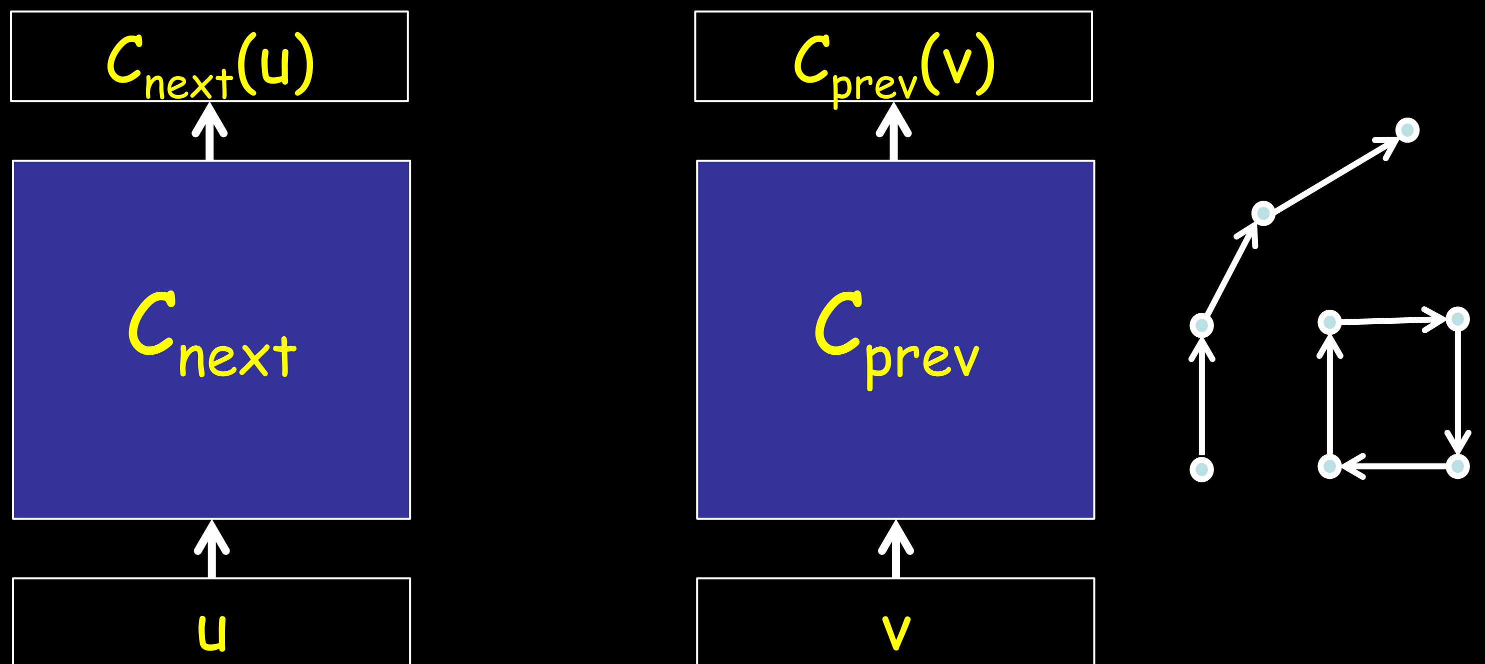
"END OF THE LINE"

What about just finding some NE?

Why isn't this problem trivial? Say $outdeg(0^n)=1$.

- for($u = 0^n$; $u == C_{prev}(C_{next}(u))$; $u = C_{next}(u)$);

Unfortunately, the path might be exponentially long.

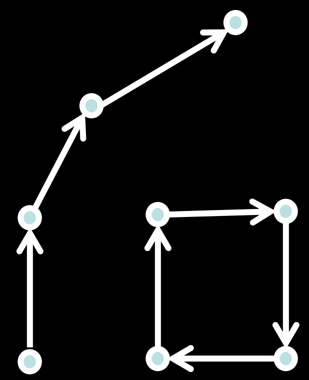


What about just finding some NE?

Not going to give proof that Nash is PPAD-hard.

Instead, give algorithm to show why Nash is in PPAD.

Also another proof of existence of NE



Lemke-Howson algorithm (1964)

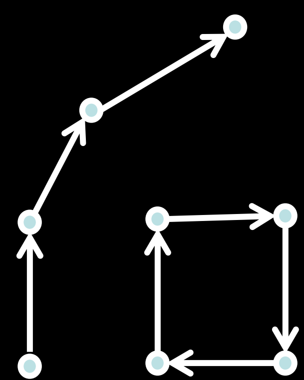
Preliminaries: [following discussion in Ch 2]

Given: matrices R,C with positive entries.

- For simplicity, convert to symmetric game (A,A^T): $A = \begin{bmatrix} 0 & R \\ C^T & 0 \end{bmatrix}$

Claim: If $([x,y],[x,y])$ is a symmetric equilib in (A,A^T), then $(x/X,y/Y)$ is an equilib in (R,C).

Use $X = \sum_i x_i$, $Y = \sum_i y_i$



Pf: Each player getting payoff $x^TRy + y^TC^Tx$ with no incentive to deviate.

Lemke-Howson algorithm (1964)

Given nxn symmetric game A, find symm equil.

Consider the 2n linear constraints on n vars:

- $A_i z \leq 1$ for all i.
- $z_j \geq 0$ for all j.

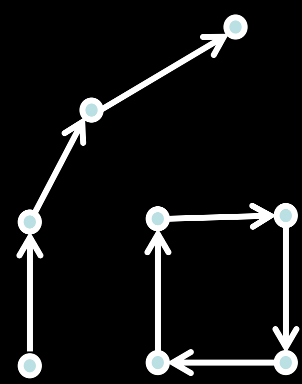
$(A_i x \leq 1/Z$ where $x_i = z_i/Z)$

$z = (z_1, z_2, \ldots, z_n)$

If not zero…

Assume A is full rank, all A_{ij} non-neg.

- Implies have a bounded polytope.
- And all vertices have n tight constraints (at equality).



Alg will start at the origin (a vertex) and move along edges to a NE.

Lemke-Howson algorithm (1964)

Given nxn symmetric game A, find symm equil.

Consider the 2n linear constraints on n vars:

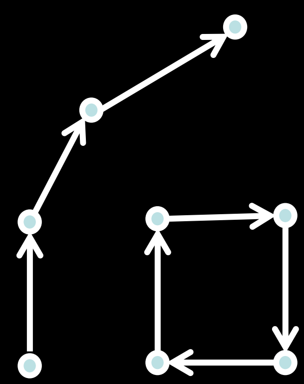
- $A_i z \leq 1$ for all i. ($A_i x \leq 1/Z$ where $x_i = z_i/Z$)
- $z_j \geq 0$ for all j. $z = (z_1, z_2, \ldots, z_n)$

If not zero…

Strategy i is "represented" if $A_i z=1$ or $z_i=0$ (or both)

What if all strategies represented?

- Either $z=(0,\ldots,0)$ or (x,x) is a symmetric Nash.



Lemke-Howson algorithm (1964)

Alg: start at (0,…,0), move along edge.

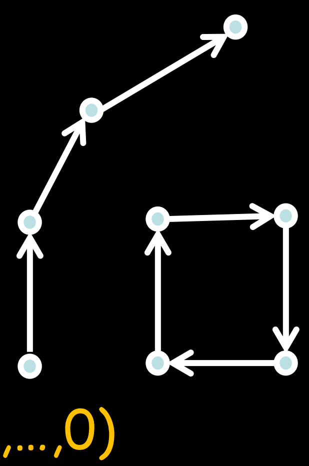
(Relax one of z_j=0 and move until hit some $A_i z$=1)

- If i=j, then all strategies represented!
- Else i is represented twice.

Strategy i is "represented" if $A_i z=1$ or $z_i=0$ (or both)

What if all strategies represented?

- Either z=(0,…,0) or (x,x) is a symmetric Nash.



Lemke-Howson algorithm (1964)

Alg: start at (0,…,0), move along edge.

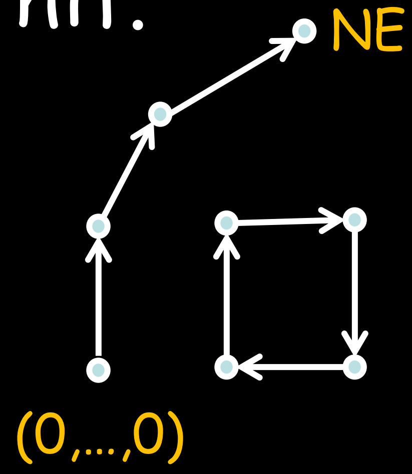
(Relax one of $z_j=0$ and move until hit some $A_i z=1$)

- If i=j, then all strategies represented!
- Else i is represented twice.

In general, take strategy represented twice and relax constraint you didn't just hit.

Claim: can't cycle or reach (0,…,0).

End is a Nash equilibrium.



Lemke-Howson algorithm (1964)

Example:

0	0	0

		*
-	0	0

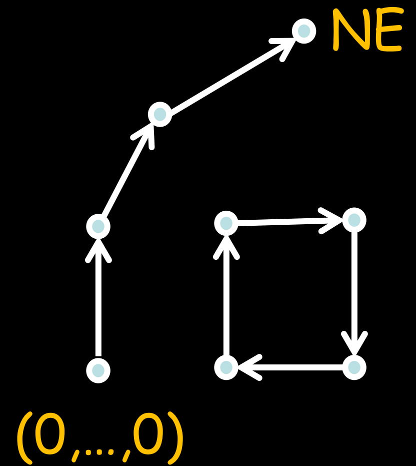
	*	*
-	0	-

	*	*
-	-	0

	*	
0	-	0

	*	
-	0	0

0	0	0



Lemke-Howson algorithm (1964)

One implication: every non-degenerate game has an odd number of Nash equilibria.