Algorithms for computing Nash equilibria of large sequential games

Javier Peña

joint work with
Samid Hoda, Andrew Gilpin, and Tuomas Sandholm at
Carnegie Mellon University

FoCM 2008
Hong Kong

Sequential games

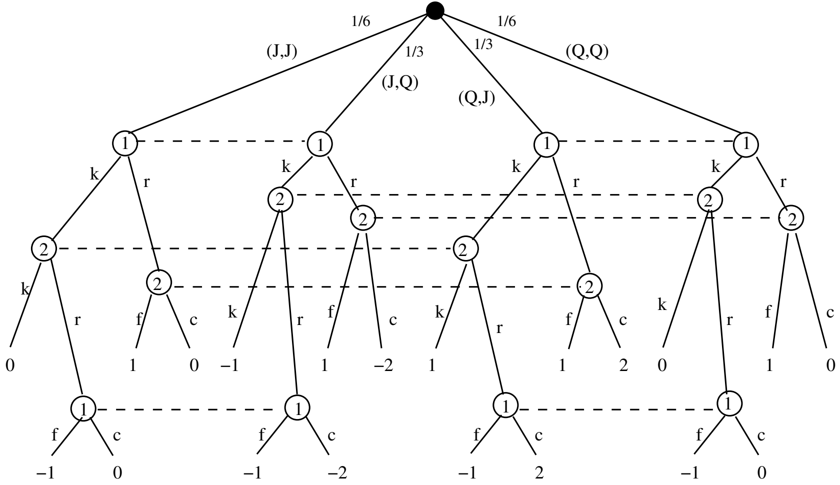
Games that involve turn-taking, chance moves, and imperfect information.

Example (simplified poker)

Card deck with two Js and two Qs

- Opening: players bet $1 each
- One card is dealt to each player
- Player 1 can check or raise
  - If Player 1 checks then Player 2 can check or raise
  - If Player 2 checks there is a showdown (higher card wins)
  - If Player 2 raises then Player 1 can fold, or call (showdown)
- If Player 1 raises then Player 2 can fold, or call (showdown)

Game tree for simplified poker



Nash equilibrium

Simultaneous choice of strategies for all players so that no player has incentive to deviate.

Nash equilibrium formulation (two-person, zero-sum games)

$$\max_{x \in Q_1} \min_{y \in Q_2} \langle x, Ay \rangle = \min_{y \in Q_2} \max_{x \in Q_1} \langle x, Ay \rangle.$$

- Q_1, Q_2: sets of strategies of players 1 and 2 respectively
- A: player 1's payoff matrix
- Games in normal form: Q_1, Q_2 are simplexes.
- **Sequential** games in extensive form: Q_1, Q_2 are **complexes**.

Complexes

Definition

- Any standard simplex $\Delta_m := \{x \in \mathbf{R}_+^m : \sum_{i=1}^m x_i = 1\}$ is a complex
- If $P \subseteq [0,1]^p, Q \subseteq [0,1]^q$ are complexes and $i \in \{1, \ldots, p\}$ then

$$P \boxed{i} Q := \{(x, y) \in \mathbf{R}^{p+q} : x \in P,\ y \in x_i \cdot Q\}$$

  is a complex.

A complex is like a tree whose nodes are simplexes.

Example (simplified poker)

Player 1's sequences:

$$S_1 = \left\{\emptyset, k^J, r^J, k^Q, r^Q, k^J f^J, k^J c^J, k^Q f^Q, k^Q c^Q\right\}$$

Set of realization plans: $Q_1 = \{x : Ex = e,\ x \geq 0\}$, for

$$E = \begin{bmatrix} 1 & & & & & & & & \\ -1 & 1 & 1 & & & & & & \\ -1 & & & 1 & 1 & & & & \\ & -1 & & & & 1 & 1 & & \\ & & & -1 & & & & 1 & 1 \end{bmatrix},\ e = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Example (simplified poker, continued)

Player 2's sequences:

$$S_2 = \left\{ \emptyset, k^J, r^J, k^Q, r^Q, f^J, c^J, f^Q, c^Q \right\}$$

Set of realization plans: $Q_2 = \{ y : Fy = f,\ y \geq 0 \}$, for

$$F = \begin{bmatrix} 1 & & & & & & & & \\ -1 & 1 & 1 & & & & & & \\ -1 & & & 1 & 1 & & & & \\ -1 & & & & & 1 & 1 & & \\ -1 & & & & & & & 1 & 1 \end{bmatrix}, \; f = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Example (simplified poker, continued)

Player 1's Payoff matrix

$$A = \begin{bmatrix} \begin{matrix} 0 & -1/6 \\ 1/3 & 0 \end{matrix} & \begin{matrix} 1/6 & 0 & 1/6 & -1/3 \\ 1/3 & 2/3 & 1/6 & 0 \end{matrix} \\ \begin{matrix} -1/6 & -1/3 \\ 0 & -2/3 \\ -1/3 & -1/6 \\ 2/3 & 0 \end{matrix} & \end{bmatrix}$$

Computation of Nash equilibrium

Nash equilibrium

$$\max_{x \in Q_1} \min_{y \in Q_2} \langle x, Ay \rangle = \min_{y \in Q_2} \max_{x \in Q_1} \langle x, Ay \rangle.$$

Can formulate as the primal-dual pair of linear programs.
However, interesting games lead to enormous instances.

Poker

- Texas Hold'em (with limits): Game tree has $\sim 10^{18}$ nodes.
- Rhode Island Hold'em: simplification of Texas Hold'em.
  Created for AI research (Shi & Littman 2001).
  Game tree has $\sim 10^9$ nodes.
- These problems are too large for general-purpose linear programming solvers.

A first-order approach to computing Nash equilibrium

Nesterov's smoothing technique

Key ingredient: prox-functions for Q_1, Q_2.

Definition

Assume $Q \subseteq \mathbf{R}^n$ is a convex compact set. A function $d : Q \to \mathbf{R}$ is a *prox-function* if it satisfies the following properties

- d is strongly convex in Q, i.e., there exists $\sigma > 0$ such that for all $x, y \in Q$, and $\alpha \in [0, 1]$

  $$d(\alpha x + (1-\alpha)y) \leq \alpha d(x) + (1-\alpha)d(y) - \frac{1}{2}\sigma\alpha(1-\alpha)\|x-y\|^2.$$

- $\min\{d(x) : x \in Q\} = 0$.

Assume

d_1, d_2 are prox-functions for the sets Q_1, Q_2 respectively.

Nesterov's smoothing technique

Theorem (Nesterov)

Algorithm that computes $\bar{x} \in Q_1, \bar{y} \in Q_2$ such that

$$0 \leq \max_{x \in Q_1} \langle x, A\bar{y} \rangle - \min_{y \in Q_2} \langle \bar{x}, Ay \rangle \leq \epsilon$$

in

$$\left\lfloor \frac{4\|A\|}{\epsilon} \sqrt{\frac{D_1 D_2}{\sigma_1 \sigma_2}} \right\rfloor$$

gradient-type iterations.

Main work per iteration: three matrix-vector products involving A, and three subproblems of the form

$$\max_{u \in Q_i} \left\{ \langle g, u \rangle - d_i(u) \right\}. \tag{1}$$

Here $D_i = \max \{ d_i(u) : u \in Q_i \}$.

"Nice" prox-functions

To get a viable algorithm

Prox-functions d_1, d_2 for Q_1, Q_2 should be so that the subproblems (1) can be solved easily.

Definition

Assume $Q \subseteq [0,1]^n$ is convex and compact. A prox-function function $d : Q \rightarrow \mathbf{R}$ is *nice* if for any $s \in \mathbf{R}^n$ the subproblem

$$\max \{\langle s, x\rangle - d(x) : x \in Q\}$$

is *easy*, e.g., it has a closed-form solution.

Challenge

Nice prox-functions are known only for a few simple sets.

Example

For $Q := \Delta_m$, the entropy function

$$d(x) = \ln m + \sum_{i=1}^{m} x_i \ln x_i,$$

is a nice prox-function for Δ_m. In this case, the subproblem

$$\max \{\langle s, x\rangle - d(x) : x \in \Delta_m\}$$

has the closed-form solution

$$x_i = \frac{e^{s_i}}{\sum_{j=1}^{m} e^{s_j}}, \; i = 1, \ldots, m.$$

The Euclidean distance function $d(x) = \frac{1}{2} \sum_{i=1}^{m} (x_i - 1/m)^2$ is also a nice prox-function for Δ_m.

Nice prox-functions for complexes

Theorem (GHP 2007)

Any family of nice prox-functions for simplexes yields a family of nice prox-functions for complexes.

Idea of the proof.

Assume d_m is a nice prox-function for Δ_m.
Define a prox-function d_Q for each complex Q inductively as follows

- If $Q = \Delta_m$, let $d_Q := d_m$
- If $R = P \boxed{i} Q$, let

$$d_R(x, y) := d_P(x) + \bar{d}_Q(x_i, y)$$

where the function $\bar{d}_Q$ is defined as

$$\bar{d}_Q(x_i, y) = \begin{cases} x_i \cdot d_Q\left(\frac{y}{x_i}\right) & \text{if } x_i > 0, \\ 0 & \text{if } x_i = 0. \end{cases}$$

Example.

Consider $Q = \{x : Ex = e,\ x \geq 0\}$, for

$$E = \begin{bmatrix} 1 & & & & & & & & \\ -1 & 1 & 1 & & & & & & \\ -1 & & & 1 & 1 & & & & \\ & -1 & & & & 1 & 1 & & \\ & & & -1 & & & & 1 & 1 \end{bmatrix},\ e = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Entropy prox-function for simplexes yields

$$\begin{aligned} d(x) = x_2 \log x_2 + x_3 \log x_3 + \log 2 \\ + x_4 \log x_4 + x_5 \log x_5 + \log 2 \\ + x_2 \left(\frac{x_6}{x_2} \log \frac{x_6}{x_2} + \frac{x_7}{x_2} \log \frac{x_7}{x_2} + \log 2 \right) \\ + x_4 \left(\frac{x_8}{x_4} \log \frac{x_8}{x_4} + \frac{x_9}{x_4} \log \frac{x_9}{x_4} + \log 2 \right) \end{aligned}$$

To compute $\bar{x} := \operatorname{argmax}\{\langle s, x\rangle - d_Q(x)\}$:

Backward pass:

$$\begin{aligned}
\tilde{s}_i &:= s_i,\ \ i = 3,5,6,7,8,9 \\
\tilde{s}_4 &:= s_4 + \log(e^{\tilde{s}_8} + e^{\tilde{s}_9}) \\
\tilde{s}_2 &:= s_2 + \log(e^{\tilde{s}_6} + e^{\tilde{s}_7}) \\
\tilde{s}_1 &:= s_1 + \log(e^{\tilde{s}_2} + e^{\tilde{s}_3}) + \log(e^{\tilde{s}_4} + e^{\tilde{s}_5})
\end{aligned}$$

Forward pass:

$$\begin{aligned}
\bar{x}_i &= \tfrac{e^{\tilde{s}_i}}{e^{\tilde{s}_2}+e^{\tilde{s}_3}}, & i &= 2,3 \\
\bar{x}_i &= \tfrac{e^{\tilde{s}_i}}{e^{\tilde{s}_4}+e^{\tilde{s}_5}}, & i &= 4,5 \\
\bar{x}_i &= \tfrac{e^{\tilde{s}_2}}{e^{\tilde{s}_2}+e^{\tilde{s}_3}} \cdot \tfrac{e^{\tilde{s}_i}}{e^{\tilde{s}_6}+e^{\tilde{s}_7}}, & i &= 6,7 \\
\bar{x}_i &= \tfrac{e^{\tilde{s}_4}}{e^{\tilde{s}_4}+e^{\tilde{s}_5}} \cdot \tfrac{e^{\tilde{s}_i}}{e^{\tilde{s}_8}+e^{\tilde{s}_9}}, & i &= 8,9
\end{aligned}$$

Complexity results (for uniform games)

Theorem (GHP 2007)

First-order smoothing algorithm that finds $(\bar{x}, \bar{y}) \in Q_1 \times Q_2$ *such that*

$$0 \le \max_{x \in Q_1} \langle x, A\bar{y} \rangle - \min_{y \in Q_2} \langle \bar{x}, Ay \rangle \le \epsilon$$

in $\lfloor (4 n_1 n_2 / \epsilon) \|A\| \rfloor$ *iterations.*
n_i*: number of sequences of Player* i *for* $i = 1, 2$

Theorem (HPS 2008)

First-order smoothing algorithm that finds $(\bar{x}, \bar{y}) \in Q_1 \times Q_2$ *such that*

$$0 \le \max_{x \in Q_1} \langle x, A\bar{y} \rangle - \min_{y \in Q_2} \langle \bar{x}, Ay \rangle \le \epsilon$$

in $\lfloor 4 n_1 n_2 \log(\|A\|/\epsilon)\, \kappa(A, Q_1, Q_2) \rfloor$ *iterations.*
$\kappa(A, Q_1, Q_2)$*: "condition number" of instance* (A, Q_1, Q_2)

Application to poker

Poker

- Central problem in artificial intelligence
- Unlike chess or checkers, it is a game of imperfect information
- Bluffing and other deceptive strategies are necessary to be a good player.
- The development of automatic poker players is a milestone comparable to the development of a chess computer player in the nineties.

Game-theoretic approach to designing poker players

Texas Hold'em with limits

- Main version of poker used in academic research
- Game tree has about 10^{18} nodes.
- Use a sophisticated *abstraction* technique to create smaller games that approximate the original game
- Compute approximate Nash equilibria for the abstractions
- Recover approximate Nash equilibria for the original game
- Main current limitation of this approach: size of the abstractions that can be handled

Computational experience

Instances

- Two lossy abstractions of Texas Hold'em
- Lossless abstraction of Rhode Island Hold'em

Problem sizes

Name	Rows	Columns	Nonzeros
AbsTex2	160,421	160,421	8,684,668
RI	1,237,238	1,237,238	50,428,638
AbsTex3	162,216,751	162,216,766	1,737,852,626,167

Implementation

Main work per iteration

- (Most expensive) matrix-vector products $x \mapsto A^{\mathrm{T}}x,\ y \mapsto Ay$
- Subproblems $\max\limits_{u \in Q_i} \{\langle g, u\rangle - d_i(u)\}$.

Peculiar structure in poker instances

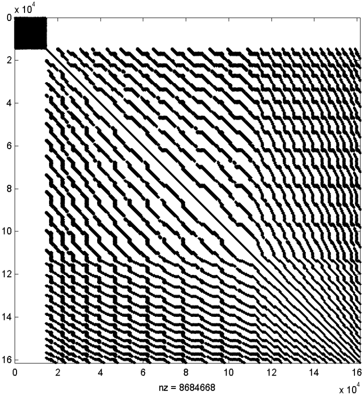
- Payoff matrix in poker games admits a concise representation. For example, for a three-round game

$$A = \begin{bmatrix} A_1 & & \\ & A_2 & \\ & & A_3 \end{bmatrix}$$

  where $A_i = F_i \otimes B_i,\ i = 1, 2$ and $A_3 = F_3 \otimes B_3 + S \otimes W$
- Do not need to form A explicitly.
- Instead have subroutines that compute $x \mapsto A^{\mathrm{T}}x,\ y \mapsto Ay$.

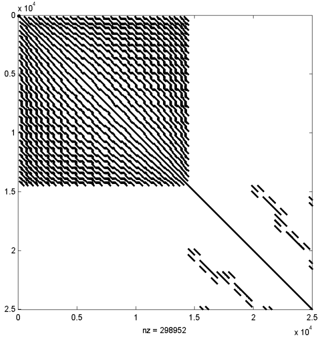
Fractal-like structure in the payoff matrix

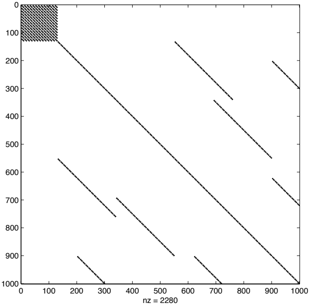
Matrix A for AbsTex2 (a three-round abstraction)



nnz = 8,684,668

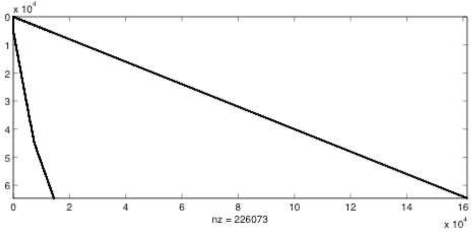
$25k \times 25k$ and $1k \times 1k$ upper-left blocks of A



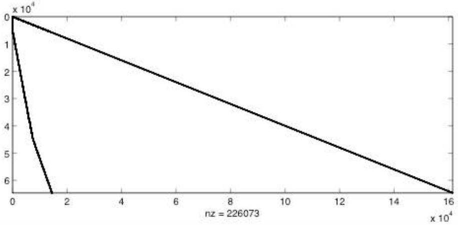


More about the AbsTex2 instance

Matrix E



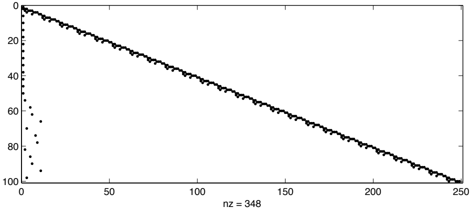
Matrix F

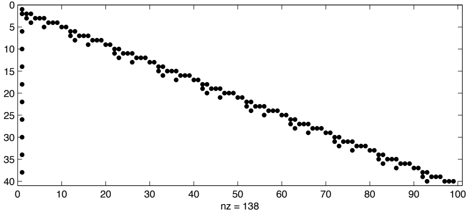


nnz $= 226073$

More about the AbsTex2 instance

Upper-left blocks of E

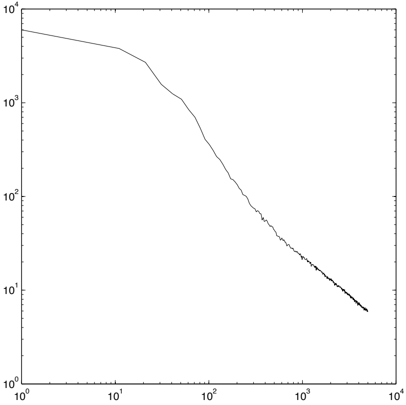




More about the AbsTex2 instance

Path of the iterates' gap

$$\max_{x \in Q_1} \langle x, Ay^k \rangle - \min_{y \in Q_2} \langle x^k, Ay \rangle$$



Do we get useful strategies?

- AAAI Poker Competition July 2007
- 17 teams competed Texas Hold'em with limits
- 14 teams competed Texas Hold'em with no limits
- GS3 & Tartanian players (based on our algorithm)
  - limit: 3rd place (out of 17)
  - no-limit: 2nd place (out of 14)
- Unlike other players, GS3 and Tartanian do not use poker-specific expert knowledge

Concluding remarks

- Nash equilibrium computation of two-person, zero-sum sequential games is amenable to smoothing techniques.
- Crux: construction of nice prox-function for complexes.
- Complexity results: $\frac{4n_1 n_2 \|A\|}{\epsilon}$ or $4n_1 n_2 \log\left(\frac{\|A\|}{\epsilon}\right) \kappa(A, Q_1, Q_2)$ simple iterations to find ϵ-equilibrium
- Promising computational results: have computed near-equilibria for games with $T \sim 10^{12}$.
- Numerical work has been instrumental in the design of competitive poker players.