Distributed Methods for Finding Approximate Equilibria

Artur Czumaj\textsuperscript{1}  Argyrios Deligkas\textsuperscript{2}  Michail Fasoulakis\textsuperscript{1}  
John Fearnley\textsuperscript{2}  Marcin Jurdziński\textsuperscript{1}  Rahul Savani\textsuperscript{2}  

\textsuperscript{1}University of Warwick  
\textsuperscript{2}University of Liverpool
Bimatrix games

Mixed strategy: probability distribution over actions

Best response: action that maximizes payoff

Regret: best response payoff - expected payoff
Bimatrix games

Nash equilibrium: A pair of mixed strategies where

- Both players have regret zero
- Both players only play best responses

eg. row player plays \((0.5, 0.5)\) and column player plays \((0.5, 0.5)\)
Bimatrix games

\[
\begin{array}{cc|cc}
 & 
 & L & R \\
\hline
I \\
\hline
 t & 0 & 1 \\
1 & 1 & 0 \\
 b & 1 & 0 \\
0 & 0 & 1 \\
\end{array}
\]

\( \epsilon \)-Approximate Nash equilibrium (\( \epsilon \)-NE)

- Nash: Both players have regret 0
- \( \epsilon \)-NE: Both players have regret at most \( \epsilon \)

eg. row player plays (0.5, 0.5) and column player plays \( R \)
Bimatrix games

\[ \begin{array}{cc}
  & L & R \\
 t & 0 & 1 \\
 b & 1 & 0 \\
\end{array} \]

\( \epsilon \)-Approximate Well Supported Nash equilibrium (\( \epsilon \)-WSNE)

- Nash: Both players only play best responses
- \( \epsilon \)-WSNE: Both players only play \( \epsilon \)-best responses

eg. row player plays \((0.4, 0.6)\) and column player plays \((0.6, 0.4)\)
Goal: Design an algorithm

- Finds an $\epsilon$-NE or $\epsilon$-WSNE
- Runs in polynomial time
Previous work

$\epsilon$-NE

- 0.75 (Kontogiannis, Panagopoulou, and Spirakis, 2006)
- 0.5 (Daskalakis, Mehta, and Papadimitriou, 2006)
- 0.38 (Kontogiannis, Panagopoulou, and Spirakis, 2007)
- 0.36 (Bosse, Byrka, and Markakis, 2007)
- **0.3393** (Tsaknakis and Spirakis, 2007)
Previous work

$\epsilon$-NE

- 0.75 (Kontogiannis, Panagopoulou, and Spirakis, 2006)
- 0.5 (Daskalakis, Mehta, and Papadimitriou, 2006)
- 0.38 (Kontogiannis, Panagopoulou, and Spirakis, 2007)
- 0.36 (Bosse, Byrka, and Markakis, 2007)
- 0.3393 (Tsaknakis and Spirakis, 2007)

$\epsilon$-WSNE

- 0.6667 (Kontogiannis and Spirakis, 2007)
- 0.6608 (Fearnley, Goldberg, Savani, and Sørensen, 2012)
Previous work

$\epsilon$-NE

- 0.75 (Kontogiannis, Panagopoulou, and Spirakis, 2006)
- 0.5 (Daskalakis, Mehta, and Papadimitriou, 2006)
- 0.38 (Kontogiannis, Panagopoulou, and Spirakis, 2007)
- 0.36 (Bosse, Byrka, and Markakis, 2007)
- 0.3393 (Tsaknakis and Spirakis, 2007)

$\epsilon$-WSNE

- 0.6667 (Kontogiannis and Spirakis, 2007)
- 0.6608 (Fearnley, Goldberg, Savani, and Sørensen, 2012)
- 0.6528 (CDFFJS, 2015)
Our results

We give a polynomial time algorithm for finding a 0.6528-WSNE

Communication complexity

- 0.6528-WSNE with polylogarithmic communication
- 0.3820-NE with polylogarithmic communication
- 0.5-WSNE with polylogarithmic communication in win-lose games

Query complexity

- 0.6528-WSNE with $n \cdot \log n$ queries
1. WSNE results
   ▶ A simple algorithm to get a 2/3-WSNE
   ▶ An improved algorithm to get a 0.6528-WSNE

2. Communication complexity results
   ▶ Communication efficient algorithm for 0.6528-WSNE

3. Query complexity results
   ▶ Query efficient algorithm for 0.6528-WSNE
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3\) against \(x^*\) return \((x^*, y^*)\)
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3\) against \(x^*\) return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - Find a row \(i\) such that \(R_{ij^*} \geq 1/3\) and \(C_{ij^*} \geq 1/3\).
   - Return \((i, j^*)\).
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3\) against \(x^*\) return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - Find a row \(i\) such that \(R_{ij^*} \geq 1/3\) and \(C_{ij^*} \geq 1/3\).
   - Return \((i, j^*)\).

Row \(i\) exists because:
   - \(x^*\) secures payoff \(> 2/3\) against \(j^*\)
   - \(j^*\) secures payoff \(> 2/3\) against \(x^*\)
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3\) against \(x^*\) return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - Find a row \(i\) such that \(R_{ij^*} \geq 1/3\) and \(C_{ij^*} \geq 1/3\).
   - Return \((i, j^*)\).

**Theorem:** This algorithm finds a 2/3-WSNE
An algorithm for 2/3-WSNE

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((-C, C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3\) against \(x^*\) return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - Find a row \(i\) such that \(R_{ij^*} \geq 1/3\) and \(C_{ij^*} \geq 1/3\).
   - Return \((i, j^*)\).

Theorem: This algorithm finds a 2/3-WSNE

We will improve it by replacing Step 3
Improve the algorithm to get a $< 2/3$-WSNE
Improve the algorithm to get a $< 2/3$-WSNE

Idea: Shift probabilities

- $B$ - set of rows with payoff $> 1/3$
- $S$ - set of rows with payoff $\leq 1/3$
- Shift all probability from $S$ to $B$
Improve the algorithm to get a $< 2/3$-WSNE

Idea: Shift probabilities

- $B$ - set of rows with payoff $> 1/3$
- $S$ - set of rows with payoff $\leq 1/3$
- Shift all probability from $S$ to $B$

Row player will be happy, column player...
Improve the algorithm to get a $< \frac{2}{3}$-WSNE

Initially, $j^*$ is a best response, as probability is shifted:

- Payoff of $j^*$ may fall
- Payoff of another row $j'$ may rise
- We may not have a $< \frac{2}{3}$-WSNE...
Improve the algorithm to get a $< \frac{2}{3}$-WSNE

This can only happen if the game looks like this:
Improve the algorithm to get a < 2/3-WSNE

This can only happen if the game looks like this:

```
<table>
<thead>
<tr>
<th></th>
<th>j*</th>
<th>j'</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1/3</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>S</td>
<td>1/3</td>
<td>1</td>
</tr>
</tbody>
</table>
```

Because:

- $x^*$ secures payoff $\geq 2/3$ against $j^*$
- $x^*$ secures payoff $\geq 2/3$ against $j'$
- $j^*$ has payoff $\geq 2/3$ against $x^*$, and $< 1/3$ against $B$
- The payoff of $j'$ is $\geq 2/3$ more than the payoff of $j^*$ against $B$
- The game contains no pure 2/3-WSNE
Improve the algorithm to get a $< \frac{2}{3}$-WSNE

This can only happen if the game looks like this:

\begin{align*}
\begin{array}{c|c|c}
\hline
& b & s \\
\hline
I & \approx \frac{1}{3} & \approx 1 \\
\hline
II & \approx 1 & \approx \frac{1}{3} \\
\hline
\end{array}
\end{align*}

In fact we can select two rows $b$ and $s$ with:

\begin{align*}
\begin{array}{c|c|c}
\hline
& b & s \\
\hline
I & 0 & \approx 1 \\
\hline
II & \approx 1 & 0 \\
\hline
\end{array}
\end{align*}

When both players mix uniformly this is a 0.5-WSNE
The algorithm

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((C, -C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3 - z\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3 - z\) against \(x^*\), return \((x^*, y^*)\)
The algorithm

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((C, -C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3 - z\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3 - z\) against \(x^*\), return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - **Shift probability** from the rows of \(j^*\) with payoff \(< 1/3 + z\)
   - If the result is a \((2/3 - z)\)-WSNE, return it
The algorithm

1. Solve the zero-sum games \((R, -R)\) and \((-C, C)\).
   - \((x^*, y^*)\) is a NE of \((R, -R)\) that secures \(v_R\)
   - \((\hat{x}, \hat{y})\) is a NE of \((C, -C)\) that secures \(v_C\)
   - Assume that \(v_R \geq v_C\)
   - If \(v_R \leq 2/3 - z\), return \((\hat{x}, y^*)\)

2. If all columns have payoff \(\leq 2/3 - z\) against \(x^*\), return \((x^*, y^*)\)

3. Otherwise:
   - Let \(j^*\) be a pure best response to \(x^*\).
   - Shift probability from the rows of \(j^*\) with payoff \(< 1/3 + z\)
   - If the result is a \((2/3 - z)\)-WSNE, return it

4. Otherwise
   - Check that the game has no pure \((2/3 - z)\)-WSNE
   - Find a \(2 \times 2\) matching pennies subgame and return it
   - Return a strategy profile where both players mix uniformly
Analysis

What is the best $z$ we can achieve?

- Steps 1 - 3 return a $(2/3 - z)$-WSNE
- Step 4 returns a

$$(1 - \frac{1 - 39z + 360z^2}{2 - 33z - 117z^2})\text{-WSNE}$$
Analysis

What is the best $z$ we can achieve?

- Steps 1 - 3 return a $(2/3 - z)$-WSNE
- Step 4 returns a

$$(1 - \frac{1 - 39z + 360z^2}{2 - 33z - 117z^2})$\text{-WSNE}$$

So we want the largest value of $z$ such that

$$1 - \frac{1 - 39z + 360z^2}{2 - 33z - 117z^2} \leq \frac{2}{3} - z$$
Analysis

\[ z = \frac{1}{117} \sqrt{3} \left( \sqrt{2434} \sqrt{3} \cos \left( \frac{1}{3} \arctan \left( \frac{39}{240073} \sqrt{9749} \sqrt{3} \right) \right) \right) \]

\[ - 3 \sqrt{2434} \sin \left( \frac{1}{3} \arctan \left( \frac{39}{240073} \sqrt{9749} \sqrt{3} \right) \right) - 48 \sqrt{3} \]
Analysis

\[ z = \frac{1}{117} \sqrt{3} \left( \sqrt{2434} \sqrt{3} \cos \left( \frac{1}{3} \arctan \left( \frac{39}{240073} \sqrt{9749} \sqrt{3} \right) \right) \right) \]

\[ - 3 \sqrt{2434} \sin \left( \frac{1}{3} \arctan \left( \frac{39}{240073} \sqrt{9749} \sqrt{3} \right) \right) \]

\[ - 48 \sqrt{3} \]

\[ z \approx 0.013906376 \]

So we get a 0.6528-WSNE
Outline

1. WSNE results
   - A simple algorithm to get a 2/3-WSNE
   - An improved algorithm to get a 0.6528-WSNE

2. Communication complexity results
   - Communication efficient algorithm for 0.6528-WSNE

3. Query complexity results
   - Query efficient algorithm for 0.6528-WSNE
Communication Complexity

The setup:

- Both players know their own payoff matrix
- They do not know their opponent’s
- They can communicate with each other
- They eventually output a strategy profile

Goal:

- Use poly-logarithmic communication
- Output an $\epsilon$-NE or $\epsilon$-WSNE
Communication Complexity

The setup:

▶ Both players know their own payoff matrix
▶ They do not know their opponent’s
▶ They can communicate with each other
▶ They eventually output a strategy profile

Goal:

▶ Use poly-logarithmic communication
▶ Output an $\epsilon$-NE or $\epsilon$-WSNE

Previous best: (Goldberg & Pastink, 2012)

▶ 0.438-NE
▶ 0.732-WSNE
Communication Complexity

\((R, -R)\) and \((-C, C)\) can be solved with no communication.
Communication Complexity

$(R, -R)$ and $(-C, C)$ can be solved with no communication

A strategy can be transmitted with $O(\log^2 n/\epsilon^2)$ communication

(Goldberg & Pastink, 2012)

- First sample $\log n/\epsilon^2$ pure strategies
- Transmit the samples
- whp. the transmitted strategy is within $\epsilon$ of the original
Communication Complexity

\((R, -R)\) and \((-C, C)\) can be solved with no communication

A strategy can be transmitted with \(O(\log^2 \frac{n}{\epsilon^2})\) communication

(Goldberg & Pastink, 2012)

- First sample \(\log \frac{n}{\epsilon^2}\) pure strategies
- Transmit the samples
- whp. the transmitted strategy is within \(\epsilon\) of the original

All other steps can be carried out using \(O\left(\frac{\log^2 n}{\epsilon^2}\right)\) communication
Communication Complexity

We have $O(\log^2 n/\epsilon^2)$ communication algorithms for:

- Finding a 0.6528-WSNE
We have $O(\log^2 n/\epsilon^2)$ communication algorithms for:

- Finding a 0.6528-WSNE
- Finding a 0.3820-NE
- Finding a 0.5-WSNE in a win-lose game
Outline

1. WSNE results
   ▶ A simple algorithm to get a $2/3$-WSNE
   ▶ An improved algorithm to get a $0.6528$-WSNE

2. Communication complexity results
   ▶ Communication efficient algorithm for $0.6528$-WSNE

3. Query complexity results
   ▶ Query efficient algorithm for $0.6528$-WSNE
Query complexity

The setup:

- The algorithm knows the size of the game, but not the payoffs
- It can make **payoff queries**
  - Propose a pure strategy profile \((i, j)\)
  - Receive \(R_{ij}\) and \(C_{ij}\)

Goal:

- Use \(< n^2\) queries
- Output an \(\epsilon\)-NE or \(\epsilon\)-WSNE
Query complexity

The setup:

- The algorithm knows the size of the game, but not the payoffs
- It can make payoff queries
  - Propose a pure strategy profile \((i, j)\)
  - Receive \(R_{ij}\) and \(C_{ij}\)

Goal:

- Use \(< n^2\) queries
- Output an \(\epsilon\)-NE or \(\epsilon\)-WSNE

Previous best: (Fearnley & Savani, 2014)

- 0.3820-NE using \(O(n \cdot \log n / \epsilon^2)\) queries
- 0.6667-WSNE using \(O(n \cdot \log n / \epsilon^4)\) queries
Query complexity

A zero-sum game can be solved using $O(n \cdot \log n/\epsilon^2)$ queries
(Goldberg & Roth, 2013)
Query complexity

A zero-sum game can be solved using $O(n \cdot \log n / \epsilon^2)$ queries (Goldberg & Roth, 2013)

So:

- We solve $(R, -R)$ and $(-C, C)$ using $O(n \cdot \log n / \epsilon^2)$ queries
- We query $j^*$ and $j'$ using $2 \cdot n$ queries
- Carry out our algorithm
Query complexity

A zero-sum game can be solved using $O(n \cdot \log n/\epsilon^2)$ queries
(Goldberg & Roth, 2013)

So:

- We solve $(R, -R)$ and $(-C, C)$ using $O(n \cdot \log n/\epsilon^2)$ queries
- We query $j^*$ and $j'$ using $2 \cdot n$ queries
- Carry out our algorithm

We obtain a 0.6528-WSNE using $O(n \cdot \log n/\epsilon^2)$ queries
Conclusion

We gave a polynomial time algorithm for finding a 0.6528-WSNE

Communication complexity

- 0.6528-WSNE with polylogarithmic communication
- 0.3820-NE with polylogarithmic communication
- 0.5-WSNE with polylogarithmic communication in win-lose games

Query complexity

- 0.6528-WSNE with $n \cdot \log n$ queries