An Introduction to Counterfactual Regret Minimization

Todd W. Neller
Gettysburg College

Marc Lanctot
Maastricht University
Motivation

• 2000: Hart and Mas-Colell introduced regret matching algorithm
• 2007: Zinkevich et al. introduced counterfactual regret minimization (CFR)
  – dominant in computer poker competitions
• Perceived need:
  – introductory materials for experiential teaching of regret matching, CFR, and more advanced concepts
  – regret-based game-theory teaching that bypasses traditional path (e.g. dominated strategy elimination, simplex method)
Outline

• Regret
• Counterfactual Regret
• Assignment Handout Outline
• Conclusion
Rock-Paper-Scissors (RPS)

• Rock-Paper-Scissors (RPS)
  – 2 players, 3 possible simultaneous actions: rock (R), paper (P), scissors (S)
  – Win, tie, loss score +1, 0, -1, respectively
Regret

• Suppose you choose rock and your opponent chooses paper. Relative to your choice, how much do you regret not having chosen
  – paper?
  – scissors?

• Regret is the difference in utility between an action and your chosen action.

• Regrets: $R \rightarrow 0$ $P \rightarrow 1$ $S \rightarrow 2$
Regret Matching

• Choose an action with probability proportional to positive regrets.

• Regrets (0, 1, 2) normalized to probabilities: (0, 1/3, 2/3)

• Suppose we now choose S while our opponent chooses R.
  – Regrets: (1, 2, 0)
  – Cumulative regrets: (1, 3, 2)
  – Normalized cumulative regrets: (1/6, 3/6, 2/6)
Regret Minimization

• Regret Matching alone will not minimize regrets in the long run.
• However, the average strategy used over all iterations converges to a **correlated equilibrium**.
• In this example, average the strategies $(1/3, 1/3, 1/3)$, $(0, 1/3, 2/3)$, $(1/6, 3/6, 2/6)$, etc.
Counterfactual Regret Example

- Input: realization weights
- Compute node strategy from normalized positive cumulative regret.
- Update avg. output strategy weighted by player realization weight.
- Recursively evaluate strategy to compute action values and node value.
- Compute counterfactual regret.
- Update cumulative regret weighted by opponent realization weight.
Counterfactual Regret Example

<table>
<thead>
<tr>
<th>Player 1 Node:</th>
<th>p1</th>
<th>p2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1, a2, a3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cumulative Regret</td>
<td>20</td>
<td>-10</td>
</tr>
<tr>
<td>Positive Regret</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>Strategy</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>Cumulative Strategy +=</td>
<td>0.2</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Player 1 Node Actions:</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1'</td>
<td>0.2</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>p2'</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>v1</td>
<td>40</td>
<td>-8</td>
<td>20</td>
</tr>
</tbody>
</table>

| Node Value | 28  |
| Action Regrets | 12  | -36  | -8  |
| Counterfactual Regrets | 3   | -9   | -2  |
| Old Cumulative Regret | 20  | -10  | 30  |
| New Cumulative Regret | 23  | -19  | 28  |

- Input: realization weights
- Compute node strategy from normalized positive cumulative regret.
- Update avg. output strategy weighted by player realization weight.
- Recursively evaluate strategy to compute action values and node value.
- Compute counterfactual regret.
- Update cumulative regret weighted by opponent realization weight.
Materials Provided

- Starter example Java code explained in a 38 page PDF using Knuth’s literate programming style presentation.
- Several tested programming exercises to facilitate experiential learning and deep mastery of material.
Materials Outline

• Regret Matching and Minimization
  – Worked example: RPS regret minimization versus fixed strategy
  – Exercise: RPS equilibrium, Colonel Blotto

• CFR
  – Worked example: Kuhn Poker equilibrium
  – Exercise: 1-die-versus-1-die Dudo

• "Cleaning" strategy results

• FSICFR
  – Worked example: Liar Die
  – Exercise: 1-die-versus-1-die Dudo with 3 claim memory limit

• Exploiting Opponent Mistakes
  – Exercise: Perturbed Liar Die

• Further Challenges (e.g. Minimum Unique Fingers)
Conclusion

• Regret minimization algorithms are an important part of the modern game theory landscape.

• These literate programming materials provide
  – an expedited, experiential introduction to the main concepts.
  – a starting point for many possible advanced undergraduate / graduate research projects.