Depth-Limited Endgame solving, and

*Pluribus*, the state of the art for multi-player no-limit Texas hold’em

Tuomas Sandholm

CS 15-888
DEPTH-LIMITED SEARCH FOR IMPERFECT-INFORMATION GAMES

[BROWN, SANDHOLM & AMOS, NEURIPS-18]
Perfect-information games and single-agent search

Remaining game is too large
Perfect-information games and single-agent search

Value substituted at leaf node is estimate of both players playing perfectly thereafter

If estimate is perfect, limited-lookahead search finds optimal policy (equilibrium)

But state values are not well defined in imperfect-information games!
Depth-limited solving

[Brown, Sandholm & Amos NeurIPS-18e]

Rock-Paper-Scissors+

Depth-Limited Rock-Paper-Scissors+
Depth-limited solving

[Brown, Sandholm & Amos NeurIPS-18e]
**How to tackle this issue?**

- **Libratus**: When solving a subgame, solves it to the end of the game
- **DeepStack**: Solves depth-limited subgames, but is very expensive and relies on certain structure
- **Our new approach**: Solves depth-limited subgames, and is very cheap and general
Depth-limited solving

[Brown, Sandholm & Amos NeurIPS-18e]

- At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game
- Step 1: Solve subgame with current set of $P_2$ leaf-node policies
- Step 2: Calculate a $P_2$ best response
- Step 3: Add $P_2$ best response to set of leaf-node policies
- Repeat
Depth-limited solving

[Brown, Sandholm & Amos NeurIPS-18e]

At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game

**Step 1:** Solve subgame with current set of $P_2$ leaf-node policies

**Step 2:** Calculate a $P_2$ best response

**Step 3:** Add $P_2$ best response to set of leaf-node policies

**Repeat**
Depth-limited solving
[Brown, Sandholm & Amos NeurIPS-18e]

At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game

Step 1: Solve subgame with current set of $P_2$ leaf-node policies

**Step 2: Calculate a $P_2$ best response**

Step 3: Add $P_2$ best response to set of leaf-node policies

Repeat
Depth-limited solving
[Brown, Sandholm & Amos NeurIPS-18e]

- At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game
- Step 1: Solve subgame with current set of $P_2$ leaf-node policies
- **Step 2: Calculate a $P_2$ best response**
- Step 3: Add $P_2$ best response to set of leaf-node policies
- Repeat
Depth-limited solving

[Brown, Sandholm & Amos NeurIPS-18e]

- At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game
- Step 1: Solve subgame with current set of $P_2$ leaf-node policies
- Step 2: Calculate a $P_2$ best response
- Step 3: Add $P_2$ best response to set of leaf-node policies
- Repeat
Depth-limited solving
[Brown, Sandholm & Amos NeurIPS-18e]

- At leaf nodes, allow other player(s) one final action choosing among multiple policies for the remaining game
- **Step 1:** Solve subgame with current set of $P_2$ leaf-node policies
- **Step 2:** Calculate a $P_2$ best response
- **Step 3:** Add $P_2$ best response to set of leaf-node policies
- Repeat

**Theorem.** Converges to Nash equilibrium.
In practice, reaches very low exploitability in a small number of iterations.

There are also other ways to generate continuation policies for the opponent.
Safe depth-limited solving starting later than the root [Brown, Sandholm & Amos NeurIPS-18e]

- In imperfect-information games, “subgames” are not independent
- However, techniques from Libratus’s endgame solving can be applied, but now the endgames are midgames that end in continuation strategy choices
  - Have a blueprint strategy for the whole game
    - E.g., via abstraction+equilibrium computation, Deep CFR [Brown, Lerer, Gross & Sandholm, ICML-19c], or manual
  - When determining our strategy for an endgame, give opponent the choice of model: blueprint or endgame model [Burch et al. AAAI-14; Jackson AAAI-14; Moravcik et al. AAAI-16; Brown & Sandholm NIPS-17; Moravcik et al. Science 2017; Brown & Sandholm Science 2018]
    - Want to solve for our endgame strategy such that opponent isn’t better off choosing endgame model for any private type she may have => Theorem: safe
    - Allow opponent to get back in the endgame the gifts she has given so far => Theorem: safe [Brown & Sandholm NIPS-17 Best Paper; Science 2018]

- Can apply this recursively
  - Can include the action that the opponent made
  - Can use finer abstraction when endgame starts closer to end of the game
  - Theorem: Safe [Brown & Sandholm NIPS-17 Best Paper; Science 2018]
Head-to-head performance in 2-player no-limit Texas hold’em

[Brown, Sandholm & Amos NeurIPS-18e]

- **Baby Tartanian8**
  [2016 champion]
  - 2 million core hours
  - 18 TB of memory

- **Slumbot**
  [2018 champion]
  - 250,000 core hours
  - 2 TB of memory

- **Modicum**
  - 700 core hours
  - 16 GB of memory
  - Plays in real time with a 4-core CPU in 20 seconds per hand

<table>
<thead>
<tr>
<th></th>
<th>Baby Tartanian8</th>
<th>Slumbot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modicum (no real-time reasoning)</td>
<td>$-57 \pm 13$</td>
<td>$-11 \pm 8$</td>
</tr>
<tr>
<td>Modicum (just one continuation strategy)</td>
<td>$-10 \pm 8$</td>
<td>$-1 \pm 15$</td>
</tr>
<tr>
<td>Modicum (just a few continuation strategies)</td>
<td>$6 \pm 5$</td>
<td>$11 \pm 9$</td>
</tr>
</tbody>
</table>

Unit: milli-big-blinds / game
Key takeaways from this segment

- Planning is important in imperfect-information games, but different

- In real-time planning, you must consider how the opponent can adapt to changes in your strategy
  - Except in perfect-information games and single-agent setting

- States don’t have well-defined values in imperfect-info games

- Our depth-limited solving algorithm:
  - Is sound
  - Enabled 2nd-best AI for heads-up no-limit Texas hold’em poker to be developed on a 4-core CPU with 16 GB of RAM
MULTI-PLAYER GAMES
Multi-player games

• All prior superhuman AI game-playing milestones have been in 2-player games:
  – **Checkers**: Chinook 1994
  – **Othello**: Logistello 1997
  – **Chess**: Deep Blue 1997
  – **2-player limit Texas hold’em**: Polaris 2008
  – **Go**: AlphaGo 2016
  – **2-player no-limit Texas hold’em**: Libratus 2017
  – **Starcraft II**: AlphaStar 2019 and **DOTA 2**: OpenAI Five 2019 (if they are superhuman)

• Our research led to techniques that enabled us to develop a superhuman AI for **multi-player** no-limit Texas hold’em ...
Multi-player poker

- Recognized AI, game theory, and OR milestone that has been open for decades
- Most popular variant in the world: 6-player no-limit Texas hold’em
- Very recently we developed a superhuman AI, *Pluribus*, for this game [Brown & Sandholm, *Science* 2019]
  - Science Breakthrough of the Year runner-up, 2019
2-player 0-sum vs. multi-player games

- All prior superhuman AI game milestones have been in 2-player 0-sum games
- Multi-player games have additional issues (even in normal form):
  - Playing a Nash equilibrium is not safe
  - Finding even an approximate Nash equilibrium is hard
    - In theory [Daskalakis et al. 2009; Chen et al. 2009; Rubinstein 2018]
    - In practice, fastest complete algorithm only scales to 3-5 players and 3-5 strategies per player [Berg & Sandholm AAAI-17]
- **Pluribus** finds superhuman strategies with a novel set of algorithms
  - No guarantee that the solution is a Nash equilibrium (beyond 2-player 0-sum games)
How does *Pluribus* work?

- Developed and runs on a single server, no GPUs
- Doesn’t use any data
- Doesn’t adapt to the opponent
- Offline blueprint computation and real-time depth-limited search
Pluribus

Rules of the game

Abstraction generation
• Information abstraction algorithm [Brown, Ganzfried & Sandholm, AAMAS-15]
• Action abstraction

Coarse abstraction of the game

Blueprint computation (offline)

Blueprint strategy profile

Computing strategy for depth-limited subgame

Finer abstraction of the game

Action

Abstraction generation

Information abstraction algorithm [Brown, Ganzfried & Sandholm, AAMAS-15]

Action abstraction

Coarse abstraction of the game

Blueprint computation (offline)

Blueprint strategy profile

Computing strategy for depth-limited subgame

Finer abstraction of the game

Action
Pluribus

Rules of the game

Abstraction generation
- Information abstraction algorithm [Brown, Ganzfried & Sandholm, AAMAS-15]
- Action abstraction

Coarse abstraction of the game

Blueprint computation (offline)

Blueprint strategy profile

Finer abstraction of the game

Computing strategy for depth-limited subgame

Action
Pluribus’s new form of depth-limited search for imperfect-information games

• All players (not just opponents) pick from k continuation strategies at leaves

• Search starts before current situation (beginning of current betting round)
  – Mitigates exploitability of unsafe search while keeping its advantages
  – Our player’s strategy is kept fixed for the moves already taken
  – As in Libratus, opponents’ actual actions are added to subgame model before the subgame is solved => no need to reverse map actions
Pluribus

Rules of the game

Abstraction generation
- Information abstraction algorithm [Brown, Ganzfried & Sandholm, AAMAS-15]
- Action abstraction

Coarse abstraction of the game

Blueprint computation (offline)

Blueprint strategy profile

Computing strategy for depth-limited subgame

Action
Pluribus

Rules of the game

Abstraction generation
- **Information abstraction algorithm** [Brown, Ganzfried & Sandholm, AAMAS-15]
- Action abstraction

Coarse abstraction of the game

Blueprint computation (offline)

Blueprint strategy profile

Computing strategy for depth-limited subgame

Finer abstraction of the game

Action
Pluribus’s new equilibrium-finding algorithm

- Used for blueprint computation and for solving depth-limited subgames
- Significant improvement over MCCFR [Lanctot et al., NeurIPS-09]
  - Pluribus uses linear weighting for both regrets and for averaging the strategies
  - => “Linear MCCFR”
- New form of dynamic pruning in early part of the run
  - Not in last two steps of the game
- Saving memory: sequences allocated in RAM only if encountered
At play time, *Pluribus*:

- Runs on a regular computer using
  - 2 CPUs
  - Less than 128 GB RAM
  - No GPUs
- Plays twice as fast as human pros (20 sec / hand)
Performance against top human pros

• AIVAT [Burch et al. AAAI-18] was used in the evaluation for variance reduction

• **Experiment 1:** 1 human pro, 5 copies of *Pluribus*
  – Independent copies of *Pluribus*; didn’t know even seat of others
  – Each of Chris Ferguson and Darren Elias played 5,000 hands (also, monetary incentive to play as well as they can)
  – *Pluribus* beat each opponent with statistical significance
  – In a later identical experiment, *Pluribus* also beat Linus Loeliger

• **Experiment 2:** 5 human pros, 1 *Pluribus*
  – 10,000 hands
  – For each 6-player session, 5 humans were selected based on availability from 13 human pros
  • Each has won over $1M playing poker, many have won over $10M
  • Linus Loeliger, Jimmy Chou, Seth Davies, Michael Gagliano, Anthony Gregg, Dong Kim, Jason Les, Daniel McAulay, Nick Petrangelo, Sean Ruane, Trevor Savage, Jake Toole
  – $50,000 divided among human pros to incentivize them to play as well as they can
  – *Pluribus* won with statistical significance (p=0.028)
Improvement of *Pluribus* with training time

- 64-core server, 512 GB RAM, no GPUs
- ~$150 at cloud prices