Computer Chess & Computer Go

The Quest for Machine Intelligence

Aske Plaat
6 April 2018
Overview

• AI & Games

• From Chess to Go, Minimax & Reinforcement Learning

• Future
AI: Understand Ourselves
Games: Thinking Machines
State Space Complexity

- Tic Tac Toe: $10^3$
- Checkers: $10^{18}$
- Backgammon: $10^{20}$
- Othello: $10^{28}$
- Chess: $10^{47}$
- Shogi: $10^{71}$
- Gomoku: $10^{105}$
- Go: $10^{170}$
2 Drosophila’s of AI

- The Dream of building Thinking Machines
- 1770 Wolfgang von Kempelen: The Turk
- Charles Babbage was defeated twice by The Turk
Chess to Go: From Human Ingenuity to Machine Learning

Chess game tree

- Initial position
- Opening stage: Databases for opening moves usually cover the first 5-15 moves
- Endgame stage
- Middlegame stage: Moves in the middlegame are selected by carrying out a large search guided by the minimax algorithm. The search tree tends out at an average of 30-40 moves at each position in the tree.

Input

hidden Layer1

hidden

Output
1950 Minimax + Eval

- 1945 Konrad Zuse
- 1949 Claude Shannon
- 1951 Alan Turing
1956 AlphaBeta

John McCarthy
1962 Arthur Samuel

- IBM 700
- First Human defeat
Two Games
One Framework

game

chess

go

tree search

fixed depth
Minimax/AlphaBeta

adaptive depth
Monte Carlo Tree Search

eval features

hand crafted

Deep Reinforcement Learning/Self Play
Chess Methods
Eval

- Hand crafted features
  - material
  - mobility
  - center
  - pawn structure
  - king safety
- polynomial, tuning of weights
<table>
<thead>
<tr>
<th>Technology</th>
<th>search depth</th>
<th>level of play</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimax, evaluation function, quiescence search, transposition table, 1 million nodes/sec</td>
<td>5 ply</td>
<td>novice</td>
</tr>
<tr>
<td>plus alpha-beta search</td>
<td>10 ply</td>
<td>expert</td>
</tr>
<tr>
<td>plus heuristic pruning (null-move, history heuristic), opening book, end game database, parallel search (large computer)</td>
<td>14 ply</td>
<td>grand master</td>
</tr>
</tbody>
</table>
AlphaBeta

- Function definition
- Tree-reducer
- AlphaBeta cutoffs

- Reduce search effort to square root in best case
- Search twice as deep
- Move ordering
- knowledge
- null-window
Ordering

• AlphaBeta: efficient if best move is searched first

• Iterative Deepening

• Store best move in transposition table

• History heuristic:
  • Table[piecetype][startswith][destsquare]
  • Increment when move fails high

• Result: 93% of fail highs at first move
Iterative Deepening
Transposition Table
Variable Depth Enhancements

• Quiescence
  Eval only quiet positions. If not, extend search

• Null move
  Is passing + shallow search (cheap move) strong enough to give cutoff?

Does not work in Zugzwang
**Opening Book**

<table>
<thead>
<tr>
<th>Next Move</th>
<th>Number Games</th>
<th>White Won</th>
<th>White Win Percentage</th>
<th>Win Chart</th>
</tr>
</thead>
<tbody>
<tr>
<td>e4</td>
<td>53,035,837</td>
<td>26,970,905</td>
<td>50.9</td>
<td></td>
</tr>
<tr>
<td>d4</td>
<td>22,631,163</td>
<td>11,889,102</td>
<td>52.5</td>
<td></td>
</tr>
<tr>
<td>c4</td>
<td>3,879,308</td>
<td>2,096,086</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Nf3</td>
<td>3,618,841</td>
<td>1,938,420</td>
<td>53.6</td>
<td></td>
</tr>
<tr>
<td>f4</td>
<td>1,526,600</td>
<td>707,787</td>
<td>51.6</td>
<td></td>
</tr>
<tr>
<td>g3</td>
<td>1,394,442</td>
<td>642,471</td>
<td>46.4</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td>994,247</td>
<td>508,883</td>
<td>51.2</td>
<td></td>
</tr>
<tr>
<td>b3</td>
<td>925,417</td>
<td>458,735</td>
<td>49.5</td>
<td></td>
</tr>
<tr>
<td>b4</td>
<td>722,009</td>
<td>372,511</td>
<td>51.6</td>
<td></td>
</tr>
<tr>
<td>d3</td>
<td>655,316</td>
<td>318,619</td>
<td>48.6</td>
<td></td>
</tr>
<tr>
<td>Nc3</td>
<td>642,887</td>
<td>345,485</td>
<td>53.7</td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td>255,315</td>
<td>129,351</td>
<td>50.5</td>
<td></td>
</tr>
<tr>
<td>c3</td>
<td>181,194</td>
<td>84,695</td>
<td>46.7</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td>104,132</td>
<td>50,125</td>
<td>48.1</td>
<td></td>
</tr>
<tr>
<td>b4</td>
<td>94,200</td>
<td>40,072</td>
<td>42.5</td>
<td></td>
</tr>
<tr>
<td>f3</td>
<td>83,648</td>
<td>41,232</td>
<td>49.3</td>
<td></td>
</tr>
<tr>
<td>h3</td>
<td>62,776</td>
<td>27,679</td>
<td>44.1</td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td>62,522</td>
<td>29,626</td>
<td>47.4</td>
<td></td>
</tr>
<tr>
<td>Nh3</td>
<td>38,919</td>
<td>19,012</td>
<td>48.9</td>
<td></td>
</tr>
<tr>
<td>Na3</td>
<td>13,073</td>
<td>5,737</td>
<td>43.9</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>Opening</th>
<th>Black Win %</th>
<th>Draw %</th>
<th>Points per 100 games</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sicilian Defence</td>
<td>34</td>
<td>29</td>
<td>48.5</td>
</tr>
<tr>
<td>1</td>
<td>Nimzo Indian</td>
<td>30</td>
<td>37</td>
<td>48.5</td>
</tr>
<tr>
<td>3</td>
<td>Robatsch Defence</td>
<td>33</td>
<td>29</td>
<td>47.5</td>
</tr>
<tr>
<td>4</td>
<td>Alekhine Defence</td>
<td>32</td>
<td>30</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>Nimzowitsch Defence</td>
<td>34</td>
<td>26</td>
<td>46.5</td>
</tr>
<tr>
<td>5</td>
<td>Rat</td>
<td>30</td>
<td>33</td>
<td>46.5</td>
</tr>
<tr>
<td>7</td>
<td>Benko Gambit</td>
<td>32</td>
<td>28</td>
<td>46</td>
</tr>
<tr>
<td>7</td>
<td>Modern Defence</td>
<td>31</td>
<td>30</td>
<td>46</td>
</tr>
<tr>
<td>7</td>
<td>Queen’s Indian Defence</td>
<td>25</td>
<td>42</td>
<td>46</td>
</tr>
<tr>
<td>10</td>
<td>Pseudo King’s Indian</td>
<td>29</td>
<td>33</td>
<td>45.5</td>
</tr>
</tbody>
</table>
Endgame Database
Parallel

- Communication overhead
- Synchronization overhead
- Search overhead

The null window \((a, a+1)\) is used to search the remaining subtrees in parallel after the bound \(a\) is obtained from the leftmost subtree.
1997 Deep Blue

- 30 node, RS/6000
- 120 MHz P2SC, 480 VLSI chips
- 200 Million positions/sec
- 8,000 part eval
- 700,000 GM game Book
- Controversy. Bug or superior intelligence?
Chess vs Go: The Next Drosophila

- efficient search
- efficient eval

<table>
<thead>
<tr>
<th></th>
<th>branching factor</th>
<th>state space</th>
</tr>
</thead>
<tbody>
<tr>
<td>chess</td>
<td>20</td>
<td>$10^{47}$</td>
</tr>
<tr>
<td>go</td>
<td>200</td>
<td>$10^{170}$</td>
</tr>
</tbody>
</table>
Minimax + Eval: GNU Go

- amateur level
- 5 kyu
Chess vs Go

• tactics - dynamic
• reasoning?

• go - strategic
• visual?
1996 Neuro Go

- Markus Enzenberger

- Evaluate features such as strings

<table>
<thead>
<tr>
<th>black or white string</th>
<th>empty intersection</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of liberties</td>
<td>liberties of Black if he plays here</td>
</tr>
<tr>
<td>(1, 2, 3, 4, ≥5)</td>
<td>(1, 2, 3, 4, ≥5)</td>
</tr>
<tr>
<td>number of stones</td>
<td>liberties of White if he plays here</td>
</tr>
<tr>
<td>(1 or 2, 3, ≥4)</td>
<td>(1, 2, 3, 4, ≥5)</td>
</tr>
<tr>
<td>can be captured in a ladder</td>
<td>Black can be captured in a ladder,</td>
</tr>
<tr>
<td>if string colour plays first</td>
<td>if he plays here</td>
</tr>
<tr>
<td>can be captured in a ladder</td>
<td>White can be captured in a ladder,</td>
</tr>
<tr>
<td>if string colour plays second</td>
<td>if he plays here</td>
</tr>
<tr>
<td>cycle for Black, n moves missing</td>
<td>(n = 0, 1, 2, ≥3)</td>
</tr>
<tr>
<td>cycle for White, n moves missing</td>
<td>(n = 0, 1, 2, ≥3)</td>
</tr>
</tbody>
</table>
1992 TD Gammon

- Gerald Tesauro
- 1 hidden layer, 80 units
- NeuroGammon: Supervised Learning
- TD Gammon: Reinforcement Learning

Goal: learn to choose actions that maximize:
\[ r_0 + \gamma r_1 + \gamma^2 r_2 + ..., \text{ where } 0 \leq \gamma < 1 \]
1993 Monte Carlo Go

• Bernd Brügmann
• Game Tree vs Game Path
• Play game paths to the end
• Best move is average of paths
2006: Adaptive methods

MCTS (+RL NN)
2006 Monte Carlo Tree Search

- Remi Coulom MCTS:
  - Tree: Divide & Conquer
  - Avg over Paths: Monte Carlo Rollouts
- Kocsis & Szepesvari UCT selection rule
- Exploration/Exploitation
Monte Carlo Tree Search

- 2006 Coulom MCTS:
  - Tree: Divide & Conquer
  - Avg over Paths: Monte Carlo Rollouts

- 2006 Kocsis & Szepesvari UCT
  - Exploration/Exploitation

1. Exponential Explosion
2. Best-First
3. Eval when no features
Monte Carlo Tree Search

Selection

Expansion

Simulation

Backpropagation
Upper Confidence Bound applied to Trees

Selecting a child node $c$ which

$$c \in \arg\max_{i \in I} (v_i + C_p \times \sqrt{\frac{\ln n_p}{n_i}})$$

- $p$: $c$’s parent node
- $I$: the set of $p$’s children
- $v_i$: $i$’s approximate utility
- $n_i$: $i$’s visit count
- $n_p$: $p$’s visit count
- $C_p$: a tunable constant
Monte Carlo Tree Search

- MCTS balances Expected value and Uncertainty
- Exploitation: Anytime algorithm: best move has most visits (high certainty)
- Exploration: Visit nodes with high uncertainty, reducing it
Monte Carlo Tree Search
2013
DeepMind, Facebook
Deep Reinforcement Learning

state $s_t$

action $a_t$

reward $r_t$
Deep Reinforcement Learning

Agent

DNN

policy $\pi_\theta(s, a)$

parameter $\theta$

State

Reward

Environment

Take action

Observe state
Deep Reinforcement Learning

Convolutional Agent

input image

possible actions

convolutional neural net

run?
jump?
Planning & Generalization

- **MCTS: Planning.** Explore merit of individual actions

- **DQN: Generalization.** Learn features common to positions
2016 AlphaGo

• David Silver, Aja Huang, ...

• MCTS+NN

• Supervised Learning Roll-out Policy (Grandmaster Games)

• Reinforcement Learning Policy Network (Self Play)

• Reinforcement Learning Value Network (Self Play)
The Four Deep Networks of AlphaGO
Deep Convolutional Networks

Forward: Filter, nonlinearity, subsample, filter, nonlinearity, subsample, ..., predict

Backward: backpropagation (determines how you need to turn the parameter knobs)

• [Max Welling 2016]
MCTS+DQN

- Better selection
- Better Rollouts
- Better Values
Networks

Results

Performance with different combinations of AlphaGo components
Compute Power

![Bar chart showing the relationship between number of threads, GPUs, and computing power for single machine and distributed systems.]
2015 Fan Hui

- European Champion 2013/4/5
- Oct 2015: 2p
- London
- Informal 5-0
Through continued development, AlphaGo has created a unique and extremely powerful approach to the game of Go. To articulate its innovations as fully as possible, I enlisted the help of world champions Gu Li 9p and Zhou Ruiyang 9p. Together, we conducted an exhaustive analysis of the five games between AlphaGo and Lee Sedol, and of three games AlphaGo played against itself shortly before the match. We found its ideas both exciting and inspiring, and it became clear to us that AlphaGo represents not only a scientific and technological advancement, but also a milestone in human understanding of Go. Unconstrained by human biases, and free to experiment with radical new approaches, AlphaGo has demonstrated great open-mindedness and invigorated the game with creative new strategies. Of course, no one strategy can guarantee a player's success, but learning from these games is sure to have a positive, enlightening impact on one's Go strength and style.
2016 Lee Sedol

- 9-15 March
- 9p
- Seoul
- Publicity
- 4-1
- $ 1 million
- move 78 in game 4
2017 Ke Jie

- 23-25-27 May
- 9p
- number 1
- single machine
- 4 TPU
- Elo 4858
- 3-0
- $1.5/0.3 million
2017
AI Summer
Reinforcement Learning

- Learning from feedback
Deep RL Success Stories

**DQN** Mnih et al, NIPS 2013 / Nature 2015

**MCTS** Guo et al, NIPS 2014; **TRPO** Schulman, Levine, Moritz, Jordan, Abbeel, ICML 2015; **A3C** Mnih et al, ICML 2016; **Dueling DQN** Wang et al ICML 2016; **Double DQN** van Hasselt et al, AAAI 2016; **Prioritized Experience Replay** Schaul et al, ICLR 2016; **Bootstrapped DQN** Osband et al, 2016; **Q-Ensembles** Chen et al, 2017; **Rainbow** Hessel et al, 2017; ...
OpenAI Gym

A toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Go.

Read the launch blog post ➔
View documentation ➔
View on GitHub ➔
Future
Future Reinforcement Learning

- Metalearning
- Hierarchical RL
- Learning from sparse inputs
- Self-Reflection
- Planning/Generalization
Future Games

- Chess and Go
  - perfect information
  - 2 player
  - zero sum

<table>
<thead>
<tr>
<th></th>
<th>branching factor</th>
<th>state space</th>
</tr>
</thead>
<tbody>
<tr>
<td>chess</td>
<td>20</td>
<td>$10^{47}$</td>
</tr>
<tr>
<td>go</td>
<td>200</td>
<td>$10^{170}$</td>
</tr>
</tbody>
</table>

- Poker
- Diplomacy
- Starcraft

- Imperfect information
- Mutli-player
- Collaboration
Poker

- imperfect information
- multi-player

• Michael Bowling
Diplomacy

- imperfect information
- multi-player
Starcraft

- imperfect information
- multi-player
- large action space
- delayed credit assignment
Negotiation

An Automated Negotiation Agent for Permission Management

Tim Baarslag, Centrum Wiskunde & Informatica (CWI)
Alper T. Alan, University of Southampton
Richard Gomer, University of Southampton
Muddasser Alam, University of Oxford
Charith Perera, The Open University
Enrico H. Gerding, University of Southampton
M.C. Schraefel, University of Southampton

AAMAS, Sao Paolo, May 2017
Conclusion
Two Games
One Framework

---

game

chess

go

tree search

eval features

---

Position Evaluation in Game of Chess

\[
\text{Eval}(s) = \sum_{i=1}^{n} \lambda_i \cdot v_i(s_i)
\]

- \(\lambda_i\): Number of each kind of piece
- \(v_i\): Value of piece

\[
(1 \cdot 5) + (2 \cdot 3.25) + (2 \cdot 3) + (1 \cdot 6) + (2 \cdot 5) + (1 \cdot 20) = 57.5
\]

\[
(1 \cdot 5) + (2 \cdot 3.25) + (1 \cdot 3) + (1 \cdot 6) + (2 \cdot 5) + (1 \cdot 20) = 56.5
\]
Dual process theory of thought

System 1
Fast / Automatic
Emotional
- Impulses / Drives
- Habits
- Beliefs

System 2
Slow / Effortful
Logical
- Reflection
- Planning
- Problem solving
Conclusion

- Two different 2 player zero sum perfect information games
- Two different methods (minimax+eval/mcts+nn)
- Reinforcement learning architecture of search/features
- Open Source: Crafty, Fuego, Gym
- Focussed efforts yielded effective methods
- Optimization
  - Medical Diagnosis
  - Legal Reasoning
  - Ad targeting
  - Autonomous Vehicles
Machine Intelligence?

- Search. Intelligence?
- Reasoning. Intelligence?
- Knowledge Representation. Intelligence?
- Learning. Intelligence?
- Games. Specific Intelligence.
- General Intelligence?
AlphaGo

- Nature Jan 2016: “Mastering the game of Go with deep neural networks and tree search”

- MCTS
  supervised learning GM games policy net
  reinforcement learning self play policy net
  value net

- Fan Hui: Oct 2015
  Lee Sedol: 9-15 March 2016
  Ke Jie: 23, 25, 27 May 2017
AlphaGo Zero

- Nature Oct 2017: “Mastering the game of Go without human knowledge”

- MCTS
  Reinforcement Learning Self Play policy+value net

- Tabula Rasa

- Defeats AlphaGo 100-0
AlphaZero

- MCTS
  Reinforcement Learning Self Play policy+value net
- Go
  Chess
  Shogi
MCTS+DQN

- Rollout Policy
- SL Policy Network
- Policy gradient
- RL Policy Network
- Value Network

- Classification
- Human expert data

- Policy
- Self-play data

- Regression

---

**MCTS**

- Selection: action choice with UCB
- Expansion: add new child node
- Evaluation: value at a leaf node
- Backup: update action value