Algorithmic explorations in a Partial Information Game

Paolo Ciancarini - University of Bologna
Joint works with my students A.Bolognesi, G.Favini, A. Gasparro

Paris, February 15, 2013
Université Paris 13 - Villetaneuse
Agenda

• Partial Information Games
• Chess as a wargame: Kriegspiel
• Algorithmic issues in Kriegspiel
• Designing a Kriegspiel program
• Planning in Kriegspiel
Partial Information Games (PIGs)

• PIGs are games in which players know only partially the game status

• Examples:
  – Card games, like Poker or Bridge
  – Board games, like Stratego or Risk
  – Wargames for military training
  – Some PIGs are Chess variants: eg. Kriegspiel, Dark Chess, others
Perfect vs. imperfect

- Perfect information games: all players know the full state of the game and see all the moves
  - Checkers, Chess, Go
- Imperfect information games: the players have partial and different knowledge about the state of the game, but see the moves
  - Poker, Stratego, Risk, Battleship
- Imperfect information games where also the moves are unknown ("fog of war"), hence a referee is necessary:
  - Kriegspiel, Phantom Go
Fog of war

• The **fog of war** is the uncertainty in situational awareness experienced by participants in military operations

• The term concerns uncertainty regarding one's own capability, adversary capability, and adversary intent

• Concept introduced by Carl von Clausewitz in his posthumously published book, *Vom Kriege* (1837)

(definition from Wikipedia)
Partial information is hard

- Zermelo’s theorem: in a two-player zero-sum game of perfect information either player has an optimal strategy that guarantees a minimum payoff
- It is the foundation of the Minimax theorem, which is the basis of game tree search
- It does not hold in games with partial information (even if an optimal strategy exists, the player may not be allowed to see it)
Information set

Playing a PIG, an agent:

• Has to make some hypotheses - *beliefs* - about the state of the game (building an *information set*)

• Has to take decisions in uncertainty, possibly assigning some probabilities to the beliefs included in the information set
Poker and Bridge

• These are PIGs because players do not know other players’ hands
• In Poker ignorance is symmetric
• In Bridge there is some asymmetry: Declarer plays both Dummy and his own cards, instead the other two players must cooperate “in the dark”
Stratego

- This is a PIG because the opponent pieces are covered.
Risk (Risiko)

This is a PIG because the players’ goals are secret.
PIG based on Chess

• Kriegspiel – invented in 1890
  – Referee: Eastern or Western rules
  – Derivatives (eg. Invisible Chess)

• Dark Chess
  – Asymmetric knowledge
  – Symmetric knowledge

• Stealth Chess, Cloack Chess
Stealth Chess
Dark Chess
Kriegspiel is a partial information variant of Chess (invented 1890)
Players do not see their opponent’s pieces or moves
They hear some “outcome” of each move from a referee, similarly as in wargames
If a player tries to make an illegal move, he is allowed to try again.
Kriegspiel

Gambit Club, London, 1946
Kriegspiel

• Two players, all rules of normal Chess
• Each player only sees his army and moves
• A referee R sees the complete position and players’ moves
• After a player P makes a move, R accepts it if legal (otherwise P tries another move), and announces captures and checks, if any
Referee's messages

• After each move by a player P, the referee R can
  – Say "illegal": the move is refused and player P tries another
  – Say "check": the move is accepted, the opponent's King is in check
  – Say "capture on square XY": the move is accepted, an opponent piece has been captured
  – Stay silent: the move is accepted and the other player is informed he has to move
Example

White vision after e4

Referee: silent
Example

Black vision after f6

Referee: silent
Example

White vision after e5

Referee: “one pawn try”
Example

Black vision after fxe5

Referee: “pawn captured in e5”
Example

White vision after Qh5

Referee: “check on short diagonal”
Example

Black vision after g6

Referee: silent
Example

White vision after Be2

Referee: “one pawn try”
Example

Black vision after gxh5

Referee: “piece captured in h5”
Example

White moves Bh5

Referee:
“pawn captured in h5; check on short diagonal; checkmate”
An interface for Kriegspiel
Kriegspiel practice

• On Internet Chess Club it is Wild variant 16
• Hundreds of rated players
  – Some are strong chess players (GMs, etc.)
• About 50 games per day
• Some weak computer players
• World championship @ICGA olympics
• Simple applets: www.pathguy.com/chess/Kriegsp2.htm
Why do we study Kriegspiel?

- **Complex**: extremely large belief state makes an explicit representation of it computationally intractable
- **Challenging**: currently, the best humans are still far ahead of computer players at this game
- **Convenient**: same rules as Chess: this allows for reuse of a certain amount of game theory and software
Kriegspiel as a scientific problem

• Can the computational knowledge we have on Chess be applied on Kriegspiel?
• Is Kriegspiel better than Chess as a “Drosophila for AI”?
• Which problems are similar to Kriegspiel?
Kriegspiel and Game Theory

• Kriegspiel was invented to make Chess more similar to a wargame
• Studied by Von Neumann under the name of Blind Chess (in the book *Theory of Games and Economic Behavior*)
• Played by Nash and others researchers at RAND Corporation
• Papers by Shapley, Nau, Russell, and others
Algorithmic explorations

@UniBo we have developed for Kriegspiel:

- An extended minimax search
- A MonteCarlo Tree Search
- A retrograde algorithm to play some endgames perfectly
- A fully working program (Darkboard), world champion since 2006
- An experimental priority planner
Metapositions

• Our first program played building a tree of metapositions
• A metaposition groups several game states together to provide the illusion of complete information
• The states with the same strategy space (set of moves available to the player) may be merged together and a game tree can be built
• Concept introduced in [Sakuta 2001] to deal with a Shogi equivalent of Kriegspiel, used to solve endgame positions
KR vs K in Kriegspiel

The Black King is alone, somewhere. Can White give checkmate?
Metaposition
States as metapositions

- **Belief state**: the set of states compatible with our observations so far
- Basic idea: we represent a belief state as a metaposition
- A metaposition groups several game states together to provide the illusion of perfect information
- The goal is to apply a variant of minimax including the referee’s messages
What it looks like

• Metapositions are simpler to represent if only the enemy king is left on the board.

• Example taken from a King and Rook vs. King endgame, White to move, maximum uncertainty
Updating metapositions

- White plays Kc3.
- In this example, Kc3 can be **silent, rank check** or **illegal**.
- The first two cases include Black’s move (the enemy king spreads)
Metaposition evaluation

• We developed an evaluation function to decorate minimax trees, as in chess
• The algorithm will choose the child with the highest value (MAX)
• If the opponent is omniscient, it will pick the referee’s message with the lowest value (MIN)
Size of KRK in Kriegspiel

- 630 ways to place WK e WR;
- BK can in up to 52 positions
- $2^{52} \times 630 \sim 10^{17}$ metapositions
- Many are irrelevant: indistinguishable after two plys
Retrograde solution

- Referee R can give a number of answers
- Eg.: Kc3 can get **silent**, **Check on file** or **illegal**
- A retrograde algorithm can exploit these answers to reconstruct the starting positions
The tablebase

- If a given position is not in the tablebase we answer the largest metaposition with the related solution (=distance from checkmate)
- $10^6$ positions, meaning that only one out $10^{11}$ is meaningful
- Longest sequence: 37 moves
Solving KRK

Mate in 37

(Boyce) Mate in 26

(Magari) Mate in 30

A Kriegspiel problem

Petkovic,
*Chess and Mathematics*,
Dover 1997

Checkmate in 14
The longest checkmate

• 37 moves
• Save the WR: Rf4, Kc2, Rf8, Kd3, Rg8, Rh8, Rh1, Rd1, Kc2
• Then “squeeze” the BK towards the border
Infinite board

*Figure 6.1:* KRK on an infinite chessboard; the actual King is only one (this problem and its solution is due to Lloyd Shapley).

Tablebase results

• We created tablebases for 4 endgames: KRK, KQK, KBBK and KBNK
• Tablebases range from 600,000 entries for KRK to 18 million entries for KBNK
• We finally solved KBNK, finding that it is always won in a fixed number of moves (89 at most). This problem was debated from the 1920s

MonteCarlo search for PIGs

• In a PIG, the game state is not fully known to the players, and minimax is computationally intractable

• In the last decade some progress has been obtained using MonteCarlo search techniques, that are very effective

• In some domains approaches based on MonteCarlo search are not enough to match the best humans
Progress

• In Chess or Go “progress” is crucial: programs rely upon minimax and evaluations which compare White and Black positions

• How can we measure the progress to victory if we do not see the opponent’s position?
MonteCarlo Tree Search

- Monte Carlo Tree Search (MCTS) has been used to play successfully complex board games such as Go.
- It has also been used to play PIGs such as Phantom Go and Kriegspiel (Parker & Nau).
- MCTS builds a game tree in four steps that are repeated iteratively as long as time allows.
Monte Carlo Tree Search (1/4)

a) Selection: the algorithm selects a leaf node from the tree according to some policy (e.g., UCT: Upper Confidence-bound applied to Trees); it has similarities to an exploration/exploitation problem.
Monte Carlo Tree Search (2/4)

b) **Expansion**: optionally, the algorithm expands a leaf to the next depth level (for example, after $x$ visits)
Monte Carlo Tree Search (3/4)

c) **Simulation**: A full game (or several) is simulated from a leaf node in the tree. Moves are random, but preferably guided by some heuristics
d) **Backpropagation**: The value of the simulated game(s) is propagated to the node’s ancestors up to the root, usually by averaging it. This will affect subsequent selection steps.
MonteCarlo search in Kriegspiel

- We adapted MCTS to play Kriegspiel better than our first program based on minimaxing a tree of metapositions.
Three approaches

• How does MCTS perform in Kriegspiel?
• We have developed three different MCTS programs for Kriegspiel, that we call A, B and C
• Program A is textbook MCTS for Kriegspiel: for each simulation we create a random layout of the enemy pieces we believe are “onboard”
Problems with approach A

- It is difficult to create good random layouts for the opponent’s pieces
- It is also time-consuming, which is very harmful to a Monte Carlo method
- Simulating the game with random moves makes for very long games that usually result in a draw regardless of the starting position
- Except in very specific scenarios, approach A turns out to be about as strong as the random player
Approach B (1/2)

- Approach B tries neither to generate layouts for the enemy pieces, nor to move them on the board.
- Only the player’s moves and their consequences are simulated.
- **The referee’s messages are simulated**: the algorithm estimates the probability of each message being given by the referee (can be upgraded with opponent modeling data).
- Even if the probabilities are not very accurate, they are more reliable than generating random layouts - not to mention much faster.
Spreading probabilities

- Every horizontal, vertical or diagonal sequence of 2 or more squares is considered.
- For each sequence, the total probability of a piece (other than King and Pawn) being there is unchanged, but the probabilities for the individual squares are adjusted so they are closer to the average.
- We ignore Knight moves for performance reasons.
Approach B (2/2)

- After estimating that, for example, Bb2-a1 has a 20% chance of capturing something…

  ... we run our Monte Carlo simulations as before, but we simulate our own moves and update the board according to the referee’s simulated messages (as defined by a probability board)

  There is one more addition from approach A: a *simulation cutoff* after \( k \) moves
Simulation cutoff

• Achieving checkmate is approach A’s major problem. Progress to checkmate happens very seldom with random moves, adding too much noise to the evaluation.
• To remedy this, we add a little game-specific knowledge to the algorithm
• Instead of running each simulation to the end, we stop it after $k$ moves and adjudicate the game to the player that seems to be winning
• This function is much simpler than a true “evaluation function” and just counts the number of pieces each player has
Approach C

- The final approach is the same as approach B, with $k = 1$.
- Simulation is stopped after only one move.
- Since there is only one move to simulate, the result can be computed as a weighed average of the possible referee messages for that move.
- Every node is computed only once, saving time. Also, simulation is very accurate in the short range, though short-sighted (but the algorithm can use quiescence).
The three approaches

A

Umpire is silent

Pawn try

Illegal move

Full game simulations

B

Value of b4-b5?

Silent (35%)

Pawn try (30%)

Illegal (35%)

k-move simulations

C

value = 0.35*v(silent) + 0.3*v(pawn_try) + 0.35*v(illegal)

Weighed average of B (k=1)
Experimental results

- We test our B and C programs against an existing Kriegspiel player based on minimax search.
- We test a 1, 2, 4 seconds per move.
- Surprisingly, short-sighted C performs best.
Results

• C performs better because it simulates better in the short range and can explore more nodes, but…
• … on higher time settings, B seems to be catching up. Eventually we expect B to be able to beat C
• Longer simulations perform better as soon as they can explore enough nodes.
• The minimax player is clearly defeated by the MonteCarlo approach
Results

• We call program C Darkboard 2.0.
• It plays better than Darkboard 1.0 (0.65-0.35 or about +120 Elo in direct matches)
• It won the 14th Computer Olympiad with a perfect score
• Consistently among top 20 players on the Internet Chess Club
• Still vulnerable to high-level human players
## ICC Elo comparison

<table>
<thead>
<tr>
<th></th>
<th>Darkboard 1.0</th>
<th>Darkboard 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Games played</td>
<td>2442</td>
<td>7121</td>
</tr>
<tr>
<td>Unique opponents</td>
<td>384 (6.36 games each)</td>
<td>589 (12.09 each)</td>
</tr>
<tr>
<td>Avg. Opponent Elo</td>
<td>1534</td>
<td>1646</td>
</tr>
<tr>
<td>Avg. Score</td>
<td>0.512</td>
<td>0.470</td>
</tr>
<tr>
<td>Avg. Elo</td>
<td><strong>1543</strong></td>
<td><strong>1626</strong></td>
</tr>
<tr>
<td>% of games vs higher Elo</td>
<td>47.0%</td>
<td>60.3%</td>
</tr>
<tr>
<td>Games vs Top 20 players</td>
<td>792 (32%)</td>
<td>2777 (39%)</td>
</tr>
<tr>
<td>Avg. Score vs Top 20 players</td>
<td><strong>0.171</strong></td>
<td><strong>0.26</strong></td>
</tr>
</tbody>
</table>
Some comments

• This scoring system is conservative by about 70-90 Elo due to ICC mechanics.
• Darkboard 1.0 played its games in 2006 and 2007, 2.0 played in 2008 and 2009.
• DB 1.0 faced and defeated many weak opponents, DB 2.0 played against the stronger players more often.
• The Top 20 players considered were the same for both programs, so the confrontation is direct.
Applicability

• Our Monte Carlo approaches are applicable to any scenario with high uncertainty and referee-mediated knowledge (e.g. sensors).

• Imperfect information tablebases are likewise applicable to any situation in which one player can force a win.
The future

• Some more improvements are necessary before we can really challenge the best humans:
  – Planning
  – More learning (currently only rote learning)
  – Recursive opponent models (“he knows that I know that he knows…”)
  – Better simulation policies
Planning

- Some PIG can be modeled with a game tree whose branching factor is very huge and even grows along the game
- **Planning**: "a selective approach only searching "important" branches as the most promising ones" (C. Shannon)
- Problem: how to use planning in a search based on MonteCarlo search
Planning

Human players do not build a complete game tree when they choose a move: a common approach in many games is to identify states that verify a set of "known relationships" as a favorable state.

Such "relationships" may be identified as "strategical elements", and we can use them as goals for a long term plan.
Conclusions

• PIGs are especially interesting for Game Theorists, because their mathematical modeling is still incomplete (eg. wrt planning)

• Kriegspiel is interesting as model of warfare and in general for taking decisions in condition of uncertainty
References

• Von Neumann and Morgenstern, *Theory of Games and Economic Behavior*, 1944
• Smith and Nau, Strategic Planning for Imperfect-Information Games, *AAAI Symposium on Games: Planning and Learning*, 1993
Research works on Kriegspiel

• Definition of game-theoretic algorithms for simple endings (Shapley, Boyce, Ferguson, Ciancarini and others)
• Planning based on MonteCarlo sampling (Parker Nau & Subrahmanian, IJCAI 2005)
• AND-OR search of belief-state trees (Russell & Wolfe, IJCAI 2005)
• Reasoning about partially observed actions (Rance Vogel & Amir AAAI 2006)
• Towards strategic Kriegspiel play with opponent modeling (Del Giudice Gmytrasiewicz & Bryan 2009)
• Parallelizing Information Set Generation for Game Tree Search Applications (Richards et al. 2012)
Unibo Publications

• A.Bolognesi, P.Ciancarini: Searching over Metapositions in Kriegspiel. *Computers and Games* 246-261, 2004
• P.Ciancarini, G.Favini: Monte Carlo Tree Search Techniques in the Game of Kriegspiel, *IJCAI*, 474-479, 2009
• A.Bolognesi, P.Ciancarini, G.Favini: Progress through uncertainty in some Kriegspiel endings, *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2):99-110, 2010
Question time

• Thank you for your attention!