Game Playing

Why is game playing an active research area in artificial intelligence?

agreement that it requires intelligence

well-defined, yet large, game spaces

easy to evaluate success

public interest
Game Playing

status of computer players

Chess

has beaten world champion

Checkers

current world champion is Chinook
checkers has been solved

Othello

computers beat human experts

Backgammon

a neural net learning system
is in top 3 in the world...
plays against itself continuously

Go

large prize offered for success
The game of checkers has roughly 500 billion billion possible positions ($5 \times 10^{20}$). The task of solving the game, determining the final result in a game with no mistakes made by either player, is daunting. Since 1989, almost continuously, dozens of computers have been working on solving checkers, applying state-of-the-art artificial intelligence techniques to the proving process. This paper announces that checkers is now solved: Perfect play by both sides leads to a draw. This is the most challenging popular game to be solved to date, roughly one million times as complex as Connect Four. Artificial intelligence technology has been used to generate strong heuristic-based game-playing programs, such as Deep Blue for chess. Solving a game takes this to the next level by replacing the heuristics with perfection.
Game Playing

How is it similar to different from problem solving?

similar

moves like actions

solution path from initial to winning situation

different

solution can not pass through losing situation

do not control every move (every other move is worst possible)
Game Playing

game solution

Can we solve a game?

What would a solution look like?

solution strategy

move selection criteria:

no matter the opponent's move,
we remain on a winning (drawing) path

Do solutions exist?

Tic-Tac-Toe, Nim(m,n)
Checkers, Go, Chess
Backgammon, Poker
Game Playing

How can we play games with no known solution strategy?

move selection based on partial search heuristic evaluation

Important Notions

static evaluation function
mini-max procedure
alpha-beta procedure plausability ordering horizon effect quiescence patterns/chunks
Game Playing

static evaluation function

measure of goodness of game situation
distance to win
likelihood of win

1-ply search

a b c d e

[] [] [] [] []
12 15 5 30 18

as a heuristic
static evaluation tries to look ahead,
but is not as good as looking
because every move is not ours to make.

How can we look ahead?
Game Playing

mini-max procedure

assume every other move is worst
(minimizes the evaluation function)

depth-first search strategy

search to ply depth desired,
alternate maximization / minimization

max

min

max

12 10 15 17 22 10 11 19 21 15 16 20 21 2
Game Playing

alpha-beta procedure

- reduction of search
- mini-max consistency

- depth first search,
  - partial results to prune unnecessary sub-trees

- associate values with nodes
  - as search progresses

with every max-level node (our choice)

alpha

- the lowest value that the node
  - will ever have to accept

with every min-level node (other's choice)

beta

- the highest value that the node
  - will ever have to accept
Game Playing

alpha-beta procedure

alpha-cutoff

if alpha of parent of a min-level node
is greater than or equal to beta of the node
then return to parent

\[
\text{alpha} \quad 25 - \quad 0 \quad \text{max}
\]
\[
0 \quad 0 \quad 0 - \quad \text{beta} \quad 19 \quad 0 \quad 0 \quad \text{min}
\]

beta-cutoff

if beta of parent of a max-level node
is less than or equal to alpha of the node
then return to parent

\[
\text{beta} \quad 25 - \quad 0 \quad \text{min}
\]
\[
0 \quad 0 \quad 0 - \quad \text{alpha} \quad 39 \quad 0 \quad 0 \quad \text{max}
\]
Game Playing

alpha-beta procedure

select-move(board, depth)
moves = all moves for player from board
alpha = -infinity
beta = infinity
for each move
val = alpha-beta(min, move(board), alpha, beta, depth-1)
if ( val > alpha )
    alpha = val
    bestmove = move
return bestmove

alpha-beta(player, board, alpha, beta, depth)
if (game over in current board position)
    return eval(board)
if (depth == 0) return eval(board)
children = all boards realized by legal moves
    for player from this board
if (player == max)
    for each child
        score = alpha-beta(min, child, alpha, beta, depth-1)
        if score > alpha then alpha = score
        // we have found a better move
        if alpha >= beta then return alpha // cut off
        return alpha // this is our best move
else // player == min
    for each child
        score = alpha-beta(max, child, alpha, beta, depth-1)
        if score < beta then beta = score
        // opponent has found a better move
        if alpha >= beta then return beta // cut off
        return beta // this is the opponent's best move
Game Playing

alpha-beta procedure

element
Game Playing

alpha-beta procedure

What is the change in search complexity that results from application of alpha-beta to mini-max search?

The branching factor is reduced to (approximately) square-root of the mini-max factor.
(Knuth and Moore, 1975)

Effect

Chess

$10^{120}$ game positions to search
(more than estimated number of atoms in universe)

mid-game branching factor of 35
12-ply look-ahead
without alpha-beta
6 million trillion positions
200 years to select move at 1/nanosecond

with alpha-beta --- branching factor of 6
23 billion board positions
23 seconds on our computer
Game Playing

alpha-beta procedure

power improved if can pick likely "best" move to allow more pruning

plausability ordering

SSS* search

use static evaluation of "internal" nodes to order expansion
Game Playing

horizon effects

partial search has a horizon
  beyond which it can not see

positive horizon effect
  believe positive outcome possible
    when soon to be undone
  accept small, quick gain,
    precluding eventual win

negative horizon effect
  deny negative outcome,
    though still heading for it
    (adding to losses)

How can we reduce the impact of horizon effects?
Game Playing

horizon effects

quiescence

search on selected nodes
to a quiet, stable position
(no large changes in evaluation
are likely)

conspiracy search

controls search based on number
of leaf nodes that must change
in order to change best move

considered stable when this number
is above some threshold
Game Playing

games with chance

poker

backgammon

bridge

introduce chance levels into game tree

backgammon

Monte Carlo techniques

bridge

use expected values

poker
Game Playing

expertise

Can we characterize aspects of human expertise in game playing?

de Groot

Thought and Choice in Chess, 1965

experiments contrasting
master
beginner / medium

experimental tasks

move-selection protocols

nothing in gross properties of process
not more or deeper search
master selects right moves

quiescence search
in difficult situations
# Game Playing

**expertise**

de Groot (1965)

experimental tasks

position acquisition/memory

5 second exposures to mid-game positions

masters remember much better almost 90% of 20 pieces

beginners only 30%

Chase and Simon (1973)

reproduce/refine deGroot results

<table>
<thead>
<tr>
<th></th>
<th>(busy)</th>
<th>(quiet)</th>
</tr>
</thead>
<tbody>
<tr>
<td>master</td>
<td>65</td>
<td>85</td>
</tr>
<tr>
<td>class A</td>
<td>35</td>
<td>55</td>
</tr>
<tr>
<td>beginner</td>
<td>20</td>
<td>35</td>
</tr>
</tbody>
</table>
Game Playing

expertise

Chase and Simon

memory (continued)

random positions
  master  20  25
  class A  20  35
  beginner 20  30

position reconstruction

interpiece latencies
  inverse of interpiece relation density
    (shape, color, adjacency,
     attack, defend)

pauses
  stable groupings
Game Playing

expertise

Reitman

Go
similar tasks / similar results

Eisenstadt and Kareev

Go/Gomoku

same position recalls
different game expertise
different pieces recalled

\[
\begin{array}{cc}
0 & x \\
x & 0 \\
x & 0 \\
\end{array}
\]
Game Playing

expertise

representation of piece patterns
patterns reflect spatial
and functional structure

chunks

refer to these specialized,
expert patterns

masters have === 70,000 + 20,000
chunks in a domain
Game Playing

expertise

PARADISE

large set of plans

goal-based, planning approach

PIONEER

chess master's method

pattern-based

What are basic aspects of human chess expertise?
Game Playing

new developments

interactive video games

Use AI to control individual characters
provide strategic direction to character groups
dynamically change parameters
to make the game challenging,
produce play-by-play commentary

Video games offer an inexpensive, reliable, and surprisingly accessible research environment, often with built-in AI interfaces.

Research

real-time heuristic search
learning.. adaptive characters
RoboCup

soccer as a task domain

simulation

robots

play-by-play

http://www.youtube.com/user/BotSportTV#p/c/0/XLKKbz2mNyo