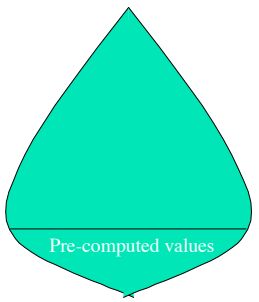
9. Odds and Ends

Jonathan Schaeffer
jonathan@cs.ualberta.ca
www.cs.ualberta.ca/~jonathan

Endgame Databases

- Pre-compute known values in the search
- Start with the goal state(s)
- Work backwards as far as time and space permits
- Retrograde analysis [1,2]

Using an Endgame Database



Normal search here until the search reaches the endgame databases.

Whenever search reaches here, can treat it as a terminal node.

Advantages

- Many more terminal nodes
- Terminal nodes are now closer to the root
- Reduces the amount of heuristic error in the search
- Effective search depth is deeper along some lines of play
- Can be used to solve a game

Endgame Databases

- Search benefits
  - Improves the quality of the evaluations
  - Perfect information with no error
  - Cuts off the search
- Less search, more accuracy
  - Cost in storage
  - Program slows down because of I/O accesses
- Benefits usually far out-weigh the costs
- Unfortunately, not possible for all games (e.g., Ataxx)

Databases in Chinook

- Pre-computed all positions with 8 or fewer pieces on the board
  - 444,000,000,000 positions
  - compressed into 6 gigabytes of data organized for real-time decompression
- Most game-playing programs are compute-bound; Chinook is I/O-bound
- Made a massive difference in quality of play
- Working on the 10-piece databases
  - 12 trillion positions.

Opening Databases

- Most games have a single starting state
- Pre-compute the initial moves of the game, saving time for later moves
- In the past, this has been done using expert knowledge
  - Belle, Chinook books constructed from human literature
  - Human literature is full of errors!

Opening Databases

- Want to automatically build an opening database
  - Save a tree on disk
  - Have processors expand leaf nodes
  - Update tree with new values
  - Search tree identifying leaf nodes to expand
  - Buro [3] ; Lincke [4]

Alpha-Beta Alternatives

- Many algorithms have been proposed as replacements to alpha-beta
- Most look good on paper, but fail in practice; usually have far too much overhead
- Min-max approximation…
- Mega-greedy search…
- BPIP…

B* [5]

- Prove that a move is best at the root, but not necessarily its value
- Evaluation function has two values: an upper bound and a lower bound on the true value
- Two search strategies: prove best or disprove rest

B*

- Prove best
  - Try to raise one move's lower bound to be >= than the upper bound of alternatives
- Disprove rest
  - Lower the upper bound of alternatives to <= that of the best move

B* Reality

- Need to have accurate upper/lower bounds for the algorithm to converge
  - In practice, this is very hard to do
- PBA* -- evaluation function is a (simple) probability distribution [6]
  - Values obtained using null-move searches
- Only one known application where B* has been effective (Scrabble)

Conspiracy Numbers [7]

- Alpha-beta returns a value, but no measure of our confidence in that value
- Conspiracy number -- the number of child nodes that have to change their value ("conspire") to cause a change in the root parent

Conspiracy Numbers

- Assume a range of values from 1 to 6
- Leaf node with value 4
  - (1, 1, 1, 0, 1, 1)
  - 0 nodes have to conspire to achieve 4
  - 1 node must conspire to achieve any other value
- Terminal node with value 4
  - (∞, ∞, ∞, 0, ∞, ∞)

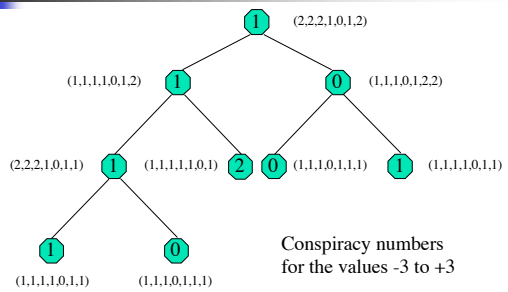
Conspiracy Numbers

- Max node: to increase a value requires only one child to change its value; to decrease requires all children with higher values to lower theirs
- $\uparrow(T,v) = 0$ for all $v <= m$ and
  MIN $\uparrow(T,v)$ for all $v > m$
- $\downarrow(T,v) = 0$ for all $v >= m$ and
  $\sum \downarrow(T,v)$ for all $v < m$

Conspiracy Numbers

- Min node: to decrease a value requires only one child to change its value; to increase requires all children with lower values to raise theirs
- $\uparrow(T,v) = 0$ for all $v <= m$ and
  $\sum \uparrow(T,v)$ for all $v > m$
- $\downarrow(T,v) = 0$ for all $v >= m$ and
  MIN $\downarrow(T,v)$ for all $v < m$

Example



Conspiracy Numbers

- Search to increase "confidence" in the root value
- Expand node that promises to give the most information on the root value
- Use iterative search
  - Prove root with a CN of 1
  - Then try and prove with CN of 2
  - Iterate until time expires

Conspiracy Numbers

- Excellent for tactics
  - Automatically searches forced lines of play (low conspiracy numbers)
- Generally not used in practice
- PN (proof number search) variant [8]
  - Win, loss, unknown values
  - Successfully used to solve several games

References

[1] K. Thompson. "Retrograde Analysis of Certain Endgames", *ICCA Journal*, vol. 9, no.3, pp. 131-139, 1986.

[2] R. Lake, J. Schaeffer and P. Lu. "Solving Large Retrograde Analysis Problems on a Network of Workstations", Advances in Computer Chess VII, pp. 135-162, 1994.

[3] M. Buro. "Toward Opening Book Learning", *ICCA Journal*, vol. 22, no. 2, pp. 98-102, 1999.

[4] T. Lincke. "Strategies for Automatic Construction of Opening Books", *Computers and Games*, T. Marsland and I. Frank (eds), pp. 74-86, Springer Verlag, 2002.

[5] H. Berliner. "The B* Tree Search Algorithm: A Best First Proof Procedure", *Artificial Intelligence*, vol. 12, pp. 23-40, 1979.

[6] A. Palay. "The B* Tree Search Algorithm -- New Results", *Artificial Intelligence*, vol. 19, pp. 145-163, 1982,

[7] D. McAllester. "Conspiracy Numbers for Min-Max Search", *Artificial Intelligence*, vol. 35, pp. 287-310, 1988.

[8] V. Allis, M. van der Muellen, and J. van den Herik. "Proof-number Search", *Artificial Intelligence*, vol. 66, pp. 91-124, 1994.