Artificial Intelligence

Unit-III

Chap-II

Adversarial Search

Alpha–beta Pruning

- In case of standard ALPHA–BETA PRUNING minimax tree, it returns the same move as minimax would, but prunes away branches that cannot possibly influence the final decision.

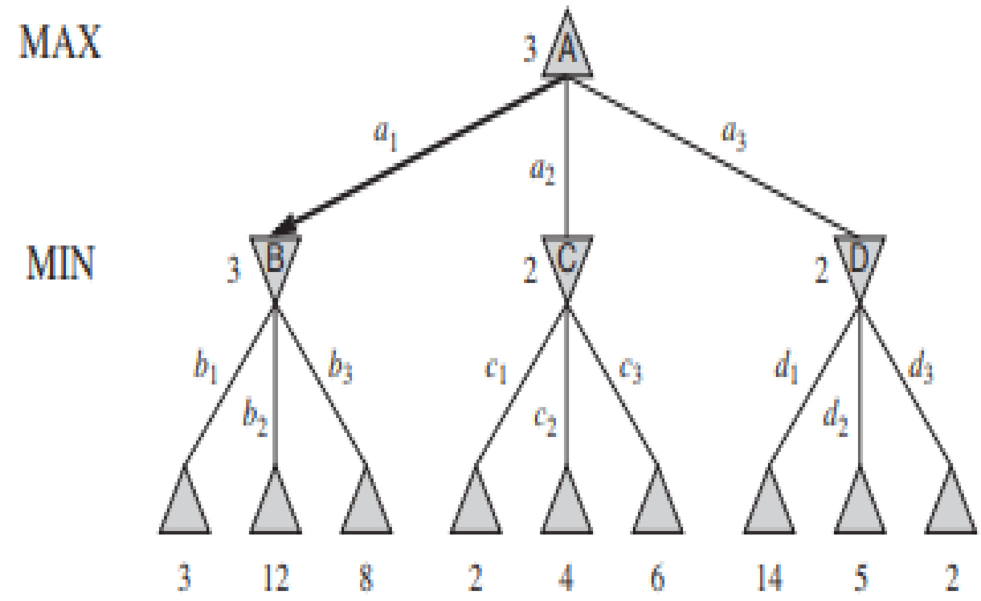


Figure A two-ply game tree. The △ nodes are "MAX nodes," in which it is MAX's turn to move, and the ▽ nodes are "MIN nodes." The terminal nodes show the utility values for MAX; the other nodes are labeled with their minimax values. MAX's best move at the root is a_1, because it leads to the state with the highest minimax value, and MIN's best reply is b_1, because it leads to the state with the lowest minimax value.

Alpha–beta Pruning

- Consider again the two-ply game tree from Figure Let’s go through the calculation of the optimal decision once more, this time paying careful attention to what we know at each point in the process.
- The steps are explained in Figure (P.T.O.).
- The outcome is that we can identify the minimax decision without ever evaluating two of the leaf nodes

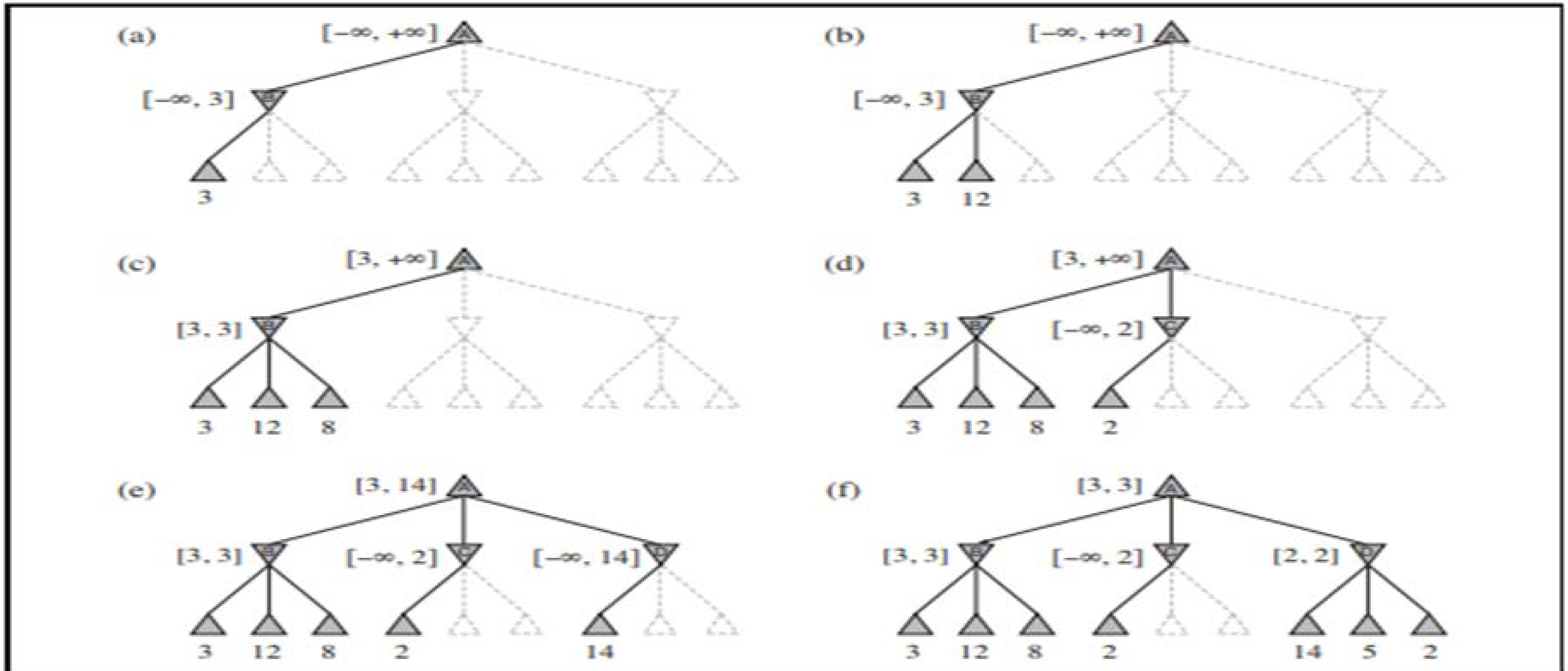


Figure Stages in the calculation of the optimal decision for the game tree in Figure . At each point, we show the range of possible values for each node. (a) The first leaf below *B* has the value 3. Hence, *B*, which is a MIN node, has a value of *at most* 3. (b) The second leaf below *B* has a value of 12; MIN would avoid this move, so the value of *B* is still at most 3. (c) The third leaf below *B* has a value of 8; we have seen all *B*'s successor states, so the value of *B* is exactly 3. Now, we can infer that the value of the root is *at least* 3, because MAX has a choice worth 3 at the root. (d) The first leaf below *C* has the value 2. Hence, *C*, which is a MIN node, has a value of *at most* 2. But we know that *B* is worth 3, so MAX would never choose *C*. Therefore, there is no point in looking at the other successor states of *C*. This is an example of alpha–beta pruning. (e) The first leaf below *D* has the value 14, so *D* is worth *at most* 14. This is still higher than MAX's best alternative (i.e., 3), so we need to keep exploring *D*'s successor states. Notice also that we now have bounds on all of the successors of the root, so the root's value is also at most 14. (f) The second successor of *D* is worth 5, so again we need to keep exploring. The third successor is worth 2, so now *D* is worth exactly 2. MAX's decision at the root is to move to *B*, giving a value of 3.

Alpha–beta Pruning

- Another way to look at this is as a simplification of the formula for MINIMAX. Let the two unevaluated successors of node C in Figure (Refer last figure) have values x and y.
- Then the value of the root node is given by

$$
\begin{aligned}
\text{MINIMAX}(root) &= \max(\min(3,12,8), \min(2,x,y), \min(14,5,2)) \\
&= \max(3, \min(2,x,y), 2) \\
&= \max(3, z, 2) \qquad \text{where } z = \min(2,x,y) \leq 2 \\
&= 3.
\end{aligned}
$$

Alpha–beta Pruning

- In other words, the value of the root and hence the minimax decision are independent of the values of the pruned leaves x and y.
- Alpha–beta pruning can be applied to trees of any depth, and it is often possible to prune entire subtrees rather than just leaves.
- The general principle is this: consider a node n somewhere in the tree (Refer Figure P.T.O.), such that Player has a choice of moving to that node.

Alpha–beta Pruning

- If Player has a better choice m either at the parent node of n or at any choice point further up, then n will never be reached in actual play. So once we have found out enough about n (by examining some of its descendants) to reach this conclusion, we can prune it.
- Alpha–beta pruning gets its name from the following two parameters(Refer Fig) that describe bounds on the backed-up values that appear anywhere along the path.

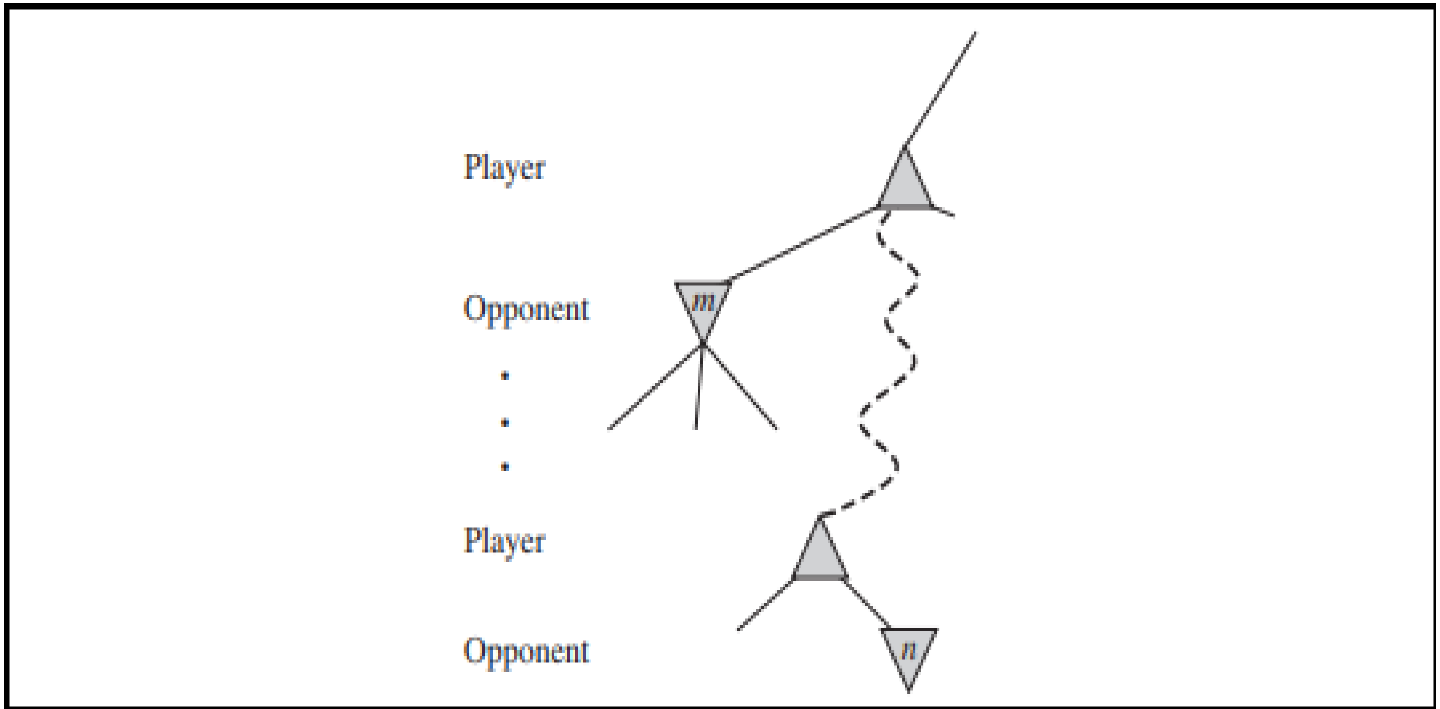


Figure The general case for alpha–beta pruning. If m is better than n for Player, we will never get to n in play.

α = the value of the best (i.e., highest-value) choice we have found so far at any choice point along the path for MAX.

β = the value of the best (i.e., lowest-value) choice we have found so far at any choice point along the path for MIN.

Alpha–beta Pruning

- Alpha–beta search updates the values of α and β as it goes along and prunes the remaining branches at a node (i.e., terminates the recursive call) as soon as the value of the current node is known to be worse than the current α or β value for MAX or MIN, respectively.

Stochastic Strategy

- A **strategy** for an agent is a probability distribution over the actions for this agent. If the agent is acting deterministically, one of the probabilities will be 1 and the rest will be 0; this is called a **pure strategy**.
- If the agent is not following a pure strategy, none of the probabilities will be 1, and more than one action will have a non-zero probability; this is called a **stochastic strategy**.
- The set of actions with a non-zero probability in a strategy is called the **support set** of the strategy.

Stochastic Games

- In real life, many unpredictable external events can put us into unforeseen situations.
- Many games mirror this unpredictability by including a random element, such as the throwing of dice.
- We call these **stochastic games**.
- Backgammon is a typical game that combines luck and skill.
- Dice are rolled at the beginning of a player's turn to determine the legal moves. In the backgammon position of Figure (P.T.O.), for example, White has rolled a 6–5 and has four possible moves.

Stochastic Games

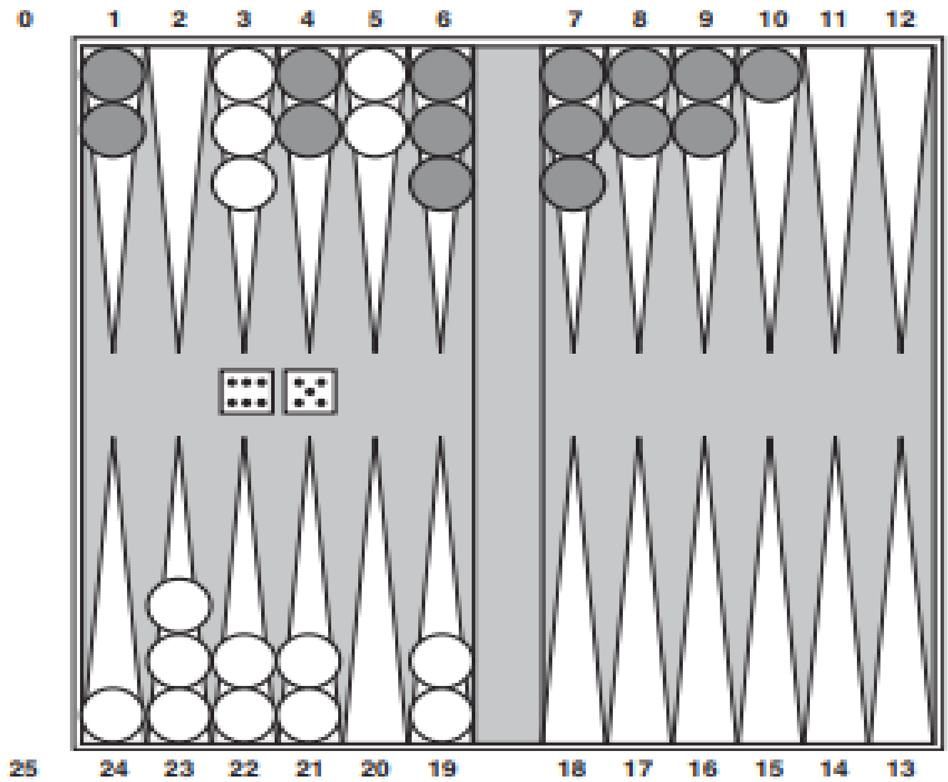


Figure A typical backgammon position. The goal of the game is to move all one's pieces off the board. White moves clockwise toward 25, and Black moves counterclockwise toward 0. A piece can move to any position unless multiple opponent pieces are there; if there is one opponent, it is captured and must start over. In the position shown, White has rolled 6–5 and must choose among four legal moves: (5–10,5–11), (5–11,19–24), (5–10,10–16), and (5–11,11–16), where the notation (5–11,11–16) means move one piece from position 5 to 11, and then move a piece from 11 to 16.

Stochastic Games

- The next step is to understand how to make correct decisions.
- However, positions do not have definite minimax values.
- Instead, we can only calculate the expected value of a position: the average over all possible outcomes of the chance nodes.
- This leads us to generalize the minimax value for deterministic games to an expectiminimax value for games with chance nodes.

Stochastic Games

- Terminal nodes and MAX and MIN nodes (for which the dice roll is known) work exactly the same way as before. For chance nodes we compute the expected value, which is the sum of the value over all outcomes, weighted by the probability of each chance action:

$$\text{EXPECTIMINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if TERMINAL-TEST}(s) \\ \max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MAX} \\ \min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MIN} \\ \sum_r P(r)\text{EXPECTIMINIMAX}(\text{RESULT}(s,r)) & \text{if PLAYER}(s) = \text{CHANCE} \end{cases}$$

where r represents a possible dice roll (or other chance event) and RESULT(s, r) is the same state as s, with the additional fact that the result of the dice roll is r.

Monte Carlo simulation vs. Alpha Beta Purning

- An alternative is to find solution Monte Carlo simulation to evaluate a position.
- Start with MONTE CARLO SIMULATION an alpha–beta (or other) search algorithm.
- From a start position, have the algorithm play thousands of games against itself, using random dice rolls.
- In the case of backgammon, the resulting win percentage has been shown to be a good approximation of the value of the position, even if the algorithm has an imperfect heuristic and is searching only a few plies ROLLOUT (Tesauro, 1995). For games with dice, this type of simulation is called a rollout.

Partially Observable Games

- Partial observability means that an agent does not know the state of the world or that the agents act simultaneously.
- Partial observability for the multiagent case is more complicated than the fully observable multiagent case or the partially observable single-agent case. The following simple examples show some important issues that arise even in the case of two agents, each with a few choices.

Partially Observable Games

- A **partially observable system** is one in which the entire state of the system is not fully visible to an external sensor.
- In a partially observable system the observer may utilise a memory system in order to add information to the observer's understanding of the system.
- An example of a partially observable system would be a card game in which some of the cards are discarded into a pile face down.
- In this case the observer is only able to view their own cards and potentially those of the dealer.

Partially Observable Games

- They are not able to view the face-down (used) cards, nor the cards which will be dealt at some stage in the future.
- A memory system can be used to remember the previously dealt cards that are now on the used pile (large collection arranged one over other).
- This adds to the total sum of knowledge that the observer can use to make decisions.

Partially Observable Games

- In contrast, a fully observable system would be that of chess. In chess (apart from the 'who is moving next' state) the full state of the system is observable at any point in time.
- Partially observable is a term used in a variety of mathematical settings, including that of Artificial Intelligence and Partially observable Markov decision processes.

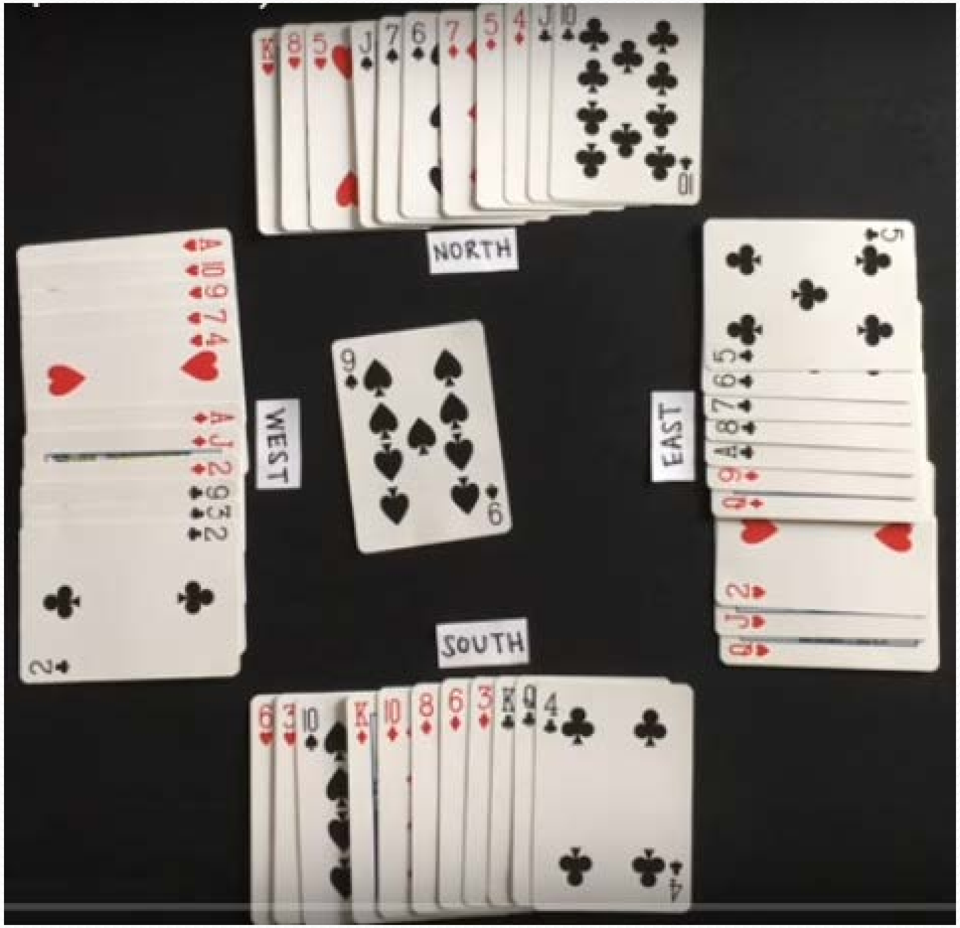
Partially Observable Games

- Chess has often been described as war in miniature, but it lacks at least one major characteristic of real wars, namely, partial observability.
- In the "fog of war," the existence and disposition of enemy units is often unknown until revealed by direct contact.
- As a result, warfare includes the use of scouts and spies togather information and the use of concealment and bluff to confuse the enemy.
- Partially observable games share these characteristics and are thus qualitatively different from other observable games.

Note : Optimal play in games of imperfect information, such as Kriegspiel (Chess) and bridge(See Figure , P.T.O.) , requires reasoning about the current and future belief states of each player.

A simple approximation can be obtained by averaging the value of an action over each possible configuration of missing information.

Bridges using cards in computer



State-of-the-art Game Programs

- State-of-the-art game programs are blindingly fast, highly optimized machines that incorporate the latest engineering advances, but they aren’t much use for doing the shopping or driving off-road.
- Racing and game-playing generate excitement and a steady stream of innovations that have been adopted by the wider community.

State-of-the-art Game Programs

- Various games for explaining state-of-the-art-game programs :

1. Chess:

Chess is a two-player strategy board game played on a chessboard, a checkered gameboard with 64 squares arranged in an 8×8 grid.

The game is played by millions of people worldwide. Chess is believed to have originated in India sometime before the 7th century. The game was derived from the Indian game chaturanga .

Chess: State-of-the-art Game Programs

- Since the second half of the 20th century, computers have been programmed to play chess with increasing success, to the point where the strongest personal computers play at a higher level than the best human players.
- Since the 1990s, computer analysis has contributed significantly to chess theory, particularly in the endgame.
- The IBM computer Deep Blue was the first machine to overcome a reigning World Chess Champion in a match when it defeated Garry Kasparov in 1997.
- The rise of strong chess engines runnable on hand-held devices has led to increasing concerns about cheating during tournaments.

Backgammon : State-of-the-art Game Programs

- Backgammon : Refer Slide no. 12

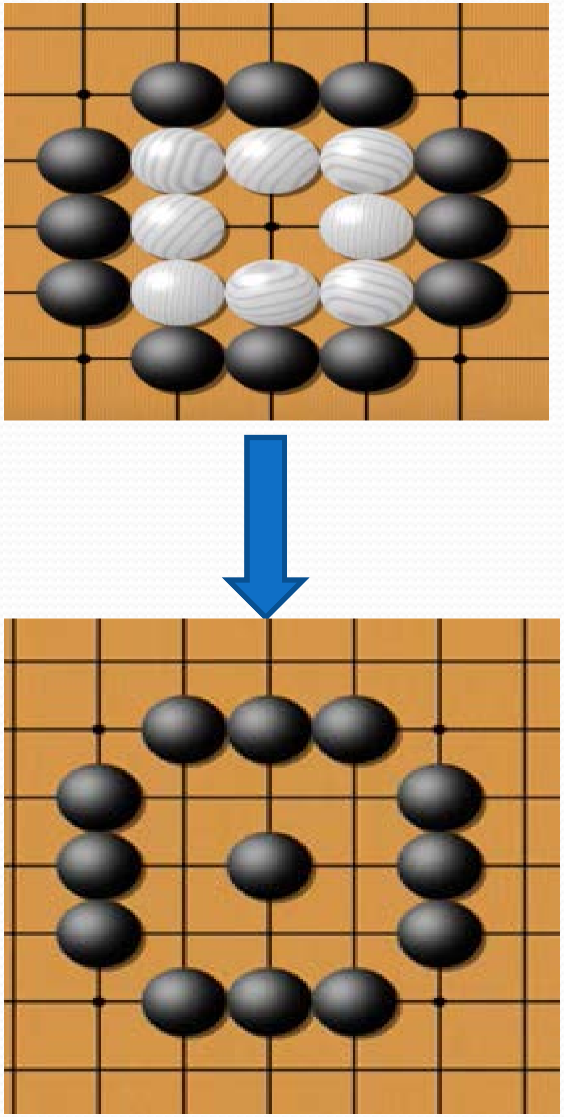
Go : State-of-the-art Game Programs

- Go :

  It is the most popular board game in Asia. Because the board is 19 × 19 and moves are allowed into (almost) every empty square, the branching factor starts at 361, which is too daunting for regular alpha–beta search methods. In addition, it is difficult to write an evaluation function because control of territory is often very unpredictable until the endgame.

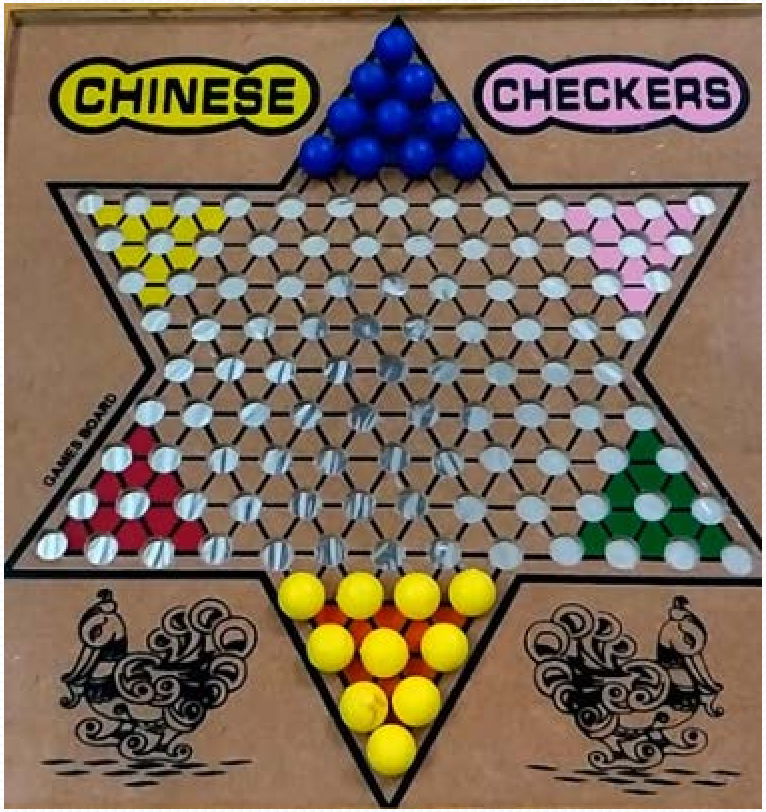
Go : State-of-the-art Game Programs

- Therefore the top programs, such as MOGO, avoid alpha–beta search and instead use Monte Carlo rollouts.
- The trick is to decide what moves to make in the course of the rollout.
- There is no aggressive pruning; all moves are possible.



Checkers : State-of-the-art Game Programs

- Jonathan Schaeffer and colleagues developed CHINOOK, which runs on regular PCs and uses alpha–beta search.
- Chinook defeated the long-running human champion in an abbreviated match in 1990, and since 2007 CHINOOK has been able to play perfectly by using alpha–beta search combined with a database of 39 trillion endgame positions.



Othello : State-of-the-art Game Programs

- Othello, also called Reversi, is probably more popular as a computer game than as a board game. It has a smaller search space than chess, usually 5 to 15 legal moves, but evaluation expertise had to be developed from scratch.

Scrabble

- Most people think the hard part about Scrabble is coming up with good words, but given the official dictionary, it turns out to be rather easy to program a move generator to find the highest-scoring move (Gordon, 1994).
- That doesn't mean the game is solved, however: merely taking the top-scoring move each turn results in a good but not expert player.
- The problem is that Scrabble is both partially observable and stochastic: you don't know what letters the other player has or what letters you will draw next.
- So playing Scrabble well combines the difficulties of backgammon and bridge