```
function MINIMAX-SEARCH(game, state) returns an action
  player ← game.TO-MOVE(state)
  value, move ← MAX-VALUE(game, state)
  return move

function MAX-VALUE(game, state) returns a (utility, move) pair
  if game.IS-TERMINAL(state) then return game.UTILITY(state, player), null
  v, move ← $-\infty$, null
  for each a in game.ACTIONS(state) do
    v2, a2 ← MIN-VALUE(game, game.RESULT(state, a))
    if $v2 > v$ then
      v, move ← v2, a
  return v, move

function MIN-VALUE(game, state) returns a (utility, move) pair
  if game.IS-TERMINAL(state) then return game.UTILITY(state, player), null
  v, move ← $+\infty$, null
  for each a in game.ACTIONS(state) do
    v2, a2 ← MAX-VALUE(game, game.RESULT(state, a))
    if $v2 < v$ then
      v, move ← v2, a
  return v, move
```

Figure 5.3 An algorithm for calculating the optimal move using minimax—the move that leads to a terminal state with maximum utility, under the assumption that the opponent plays to minimize utility. The functions MAX-VALUE and MIN-VALUE go through the whole game tree, all the way to the leaves, to determine the backed-up value of a state and the move to get there.

first recurses down to the three bottom-left nodes and uses the UTILITY function on them to discover that their values are 3, 12, and 8, respectively. Then it takes the minimum of these values, 3, and returns it as the backed-up value of node B. A similar process gives the backed-up values of 2 for C and 2 for D. Finally, we take the maximum of 3, 2, and 2 to get the backed-up value of 3 for the root node.

The minimax algorithm performs a complete depth-first exploration of the game tree. If the maximum depth of the tree is m and there are b legal moves at each point, then the time complexity of the minimax algorithm is $O(b^m)$. The space complexity is $O(bm)$ for an algorithm that generates all actions at once, or $O(m)$ for an algorithm that generates actions one at a time (see page 80). The exponential complexity makes MINIMAX impractical for complex games; for example, chess has a branching factor of about 35 and the average game has depth of about 80 ply, and it is not feasible to search $35^{80} \approx 10^{123}$ states. MINIMAX does, however, serve as a basis for the mathematical analysis of games. By approximating the minimax analysis in various ways, we can derive more practical algorithms.