Adversarial Search Cont…

Properties of α–β

- Pruning does not affect final result
- However, effectiveness of pruning affected by…?

Resource limits

Suppose we have 100 secs, explore 10^4 nodes/sec
→ 10^6 nodes per move

Standard approach (Shannon, 1950):

- evaluation function
  = estimated desirability of position
- cutoff test:
  e.g., depth limit

Cutting off search

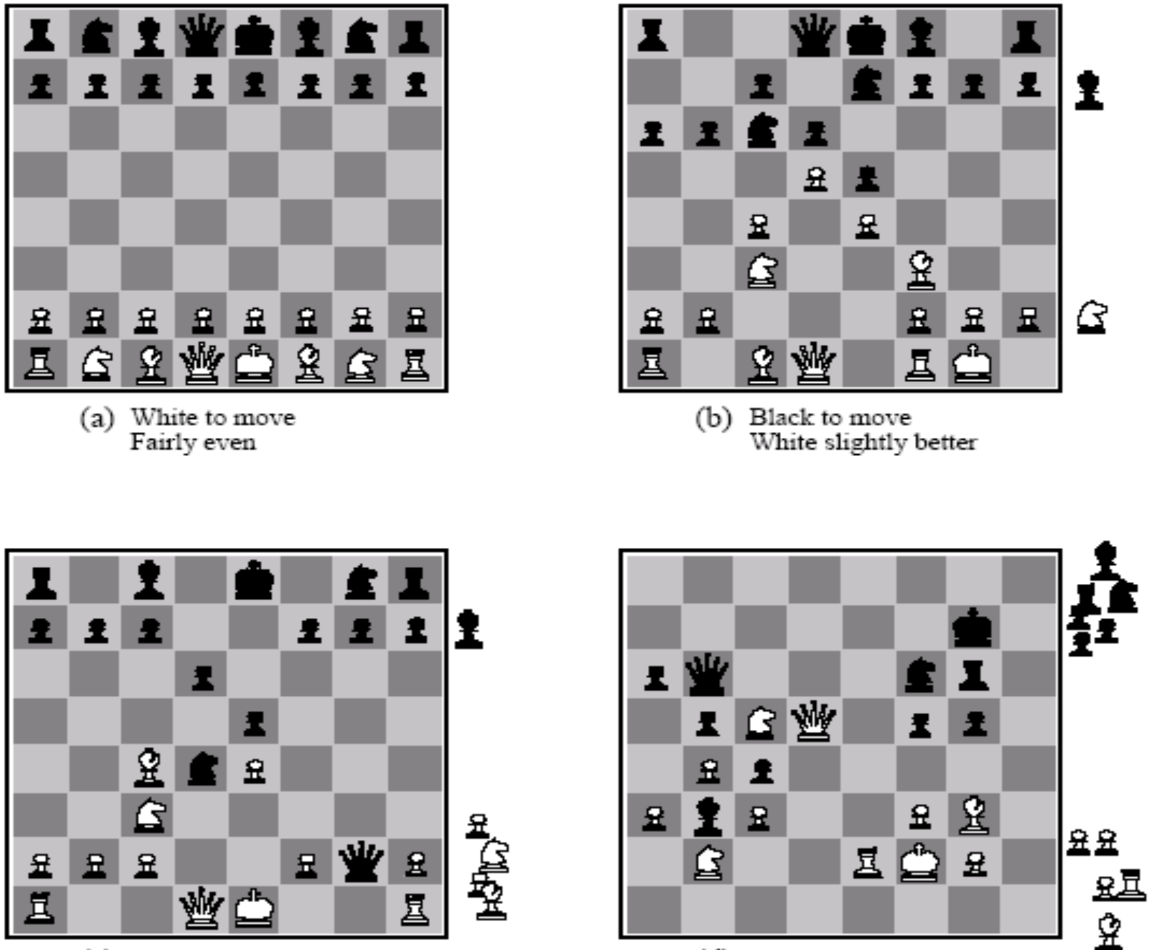
- Change:
  - if TERMINAL-TEST(state) then return UTILITY(state)
- into
  - if CUTOFF-TEST(state,depth) then return EVAL(state)
- Introduces a fixed-depth limit
  - Is selected so that the amount of time will not exceed what the rules of the game allow.
- When cuttoff occurs, the evaluation is performed.

Heuristic EVAL

- Idea: produce an estimate of the expected utility of the game from a given position.
- Performance depends on quality of EVAL.
- Requirements:
  - EVAL should order terminal-nodes in the same way as UTILITY.
  - Computation may not take too long.
  - For non-terminal states the EVAL should be strongly correlated with the actual chance of winning.

Simple Mancala Heuristic: Goodness of board = # stones in my Mancala minus the number of stones in my opponents.

Heuristic EVAL example



(a) White to move
Fairly even

(b) Black to move
White slightly better

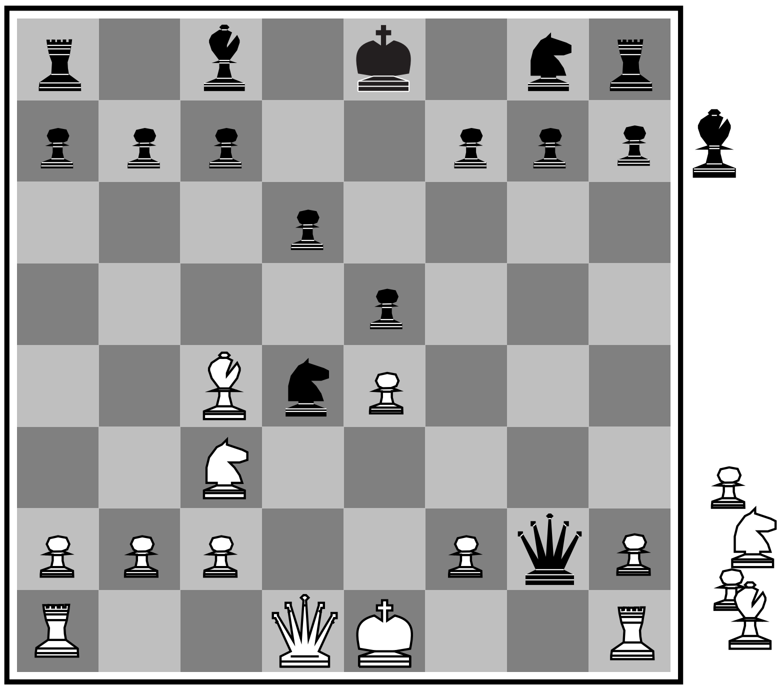
(c) White to move
Black winning

(d) Black to move
White about to lose

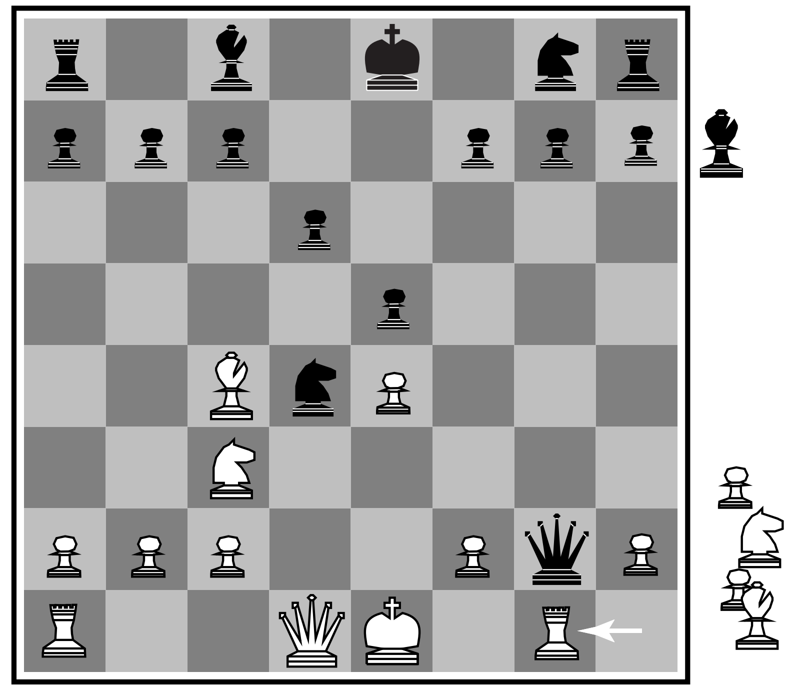
$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

Heuristic difficulties

Simple heuristic - weighing the pieces by material value



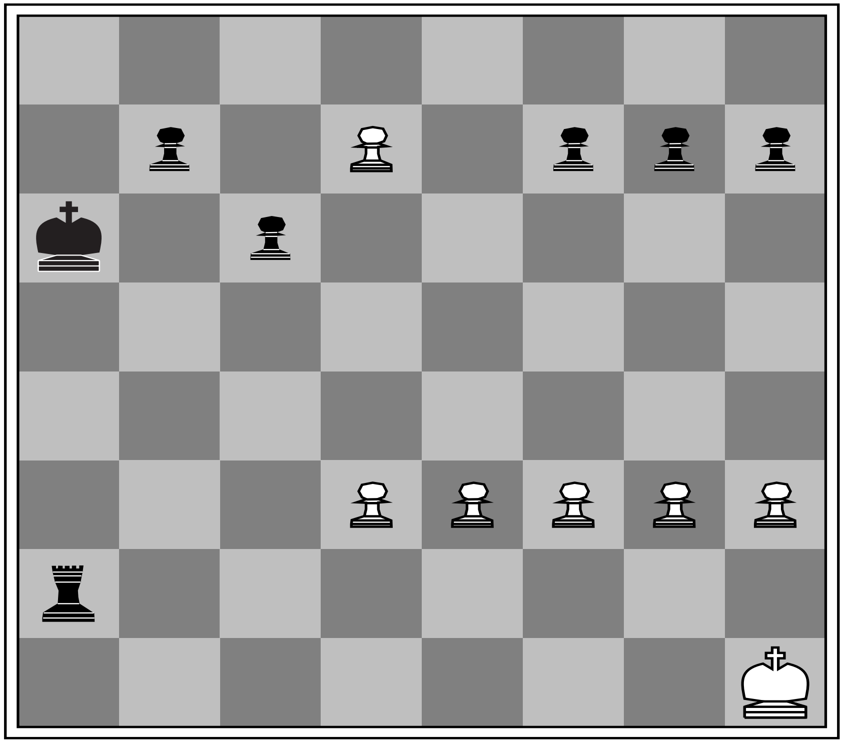
(a) White to move



(b) White to move

Horizon effect

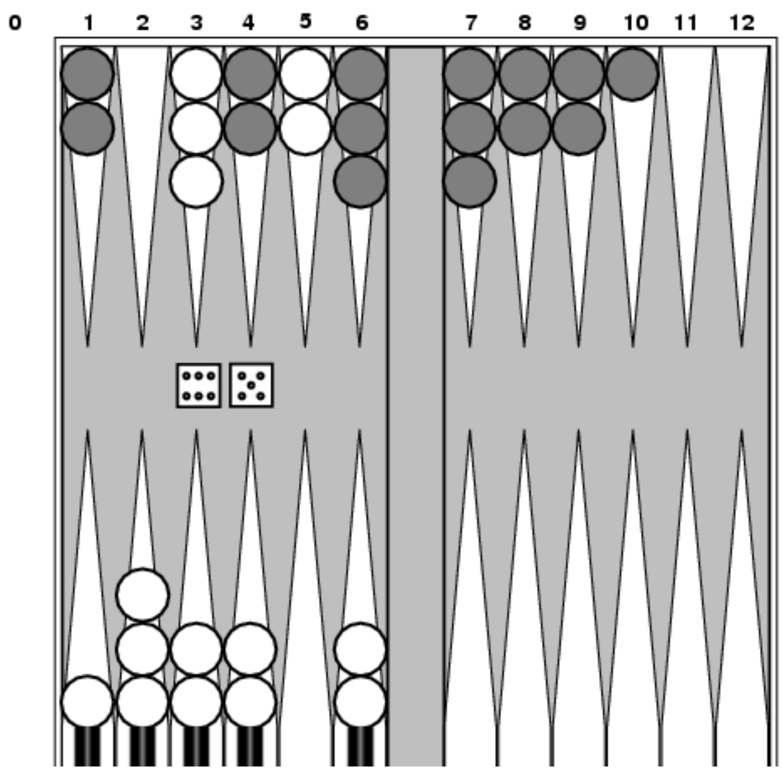
Fixed depth search
thinks it can avoid
the queening move

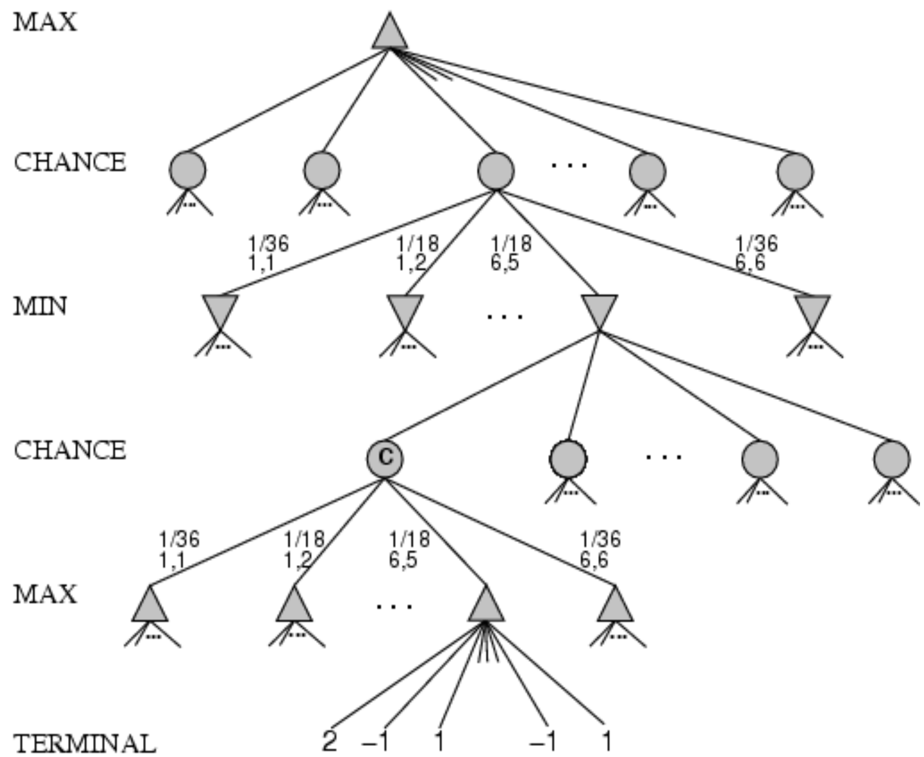


Black to move


0
64
0

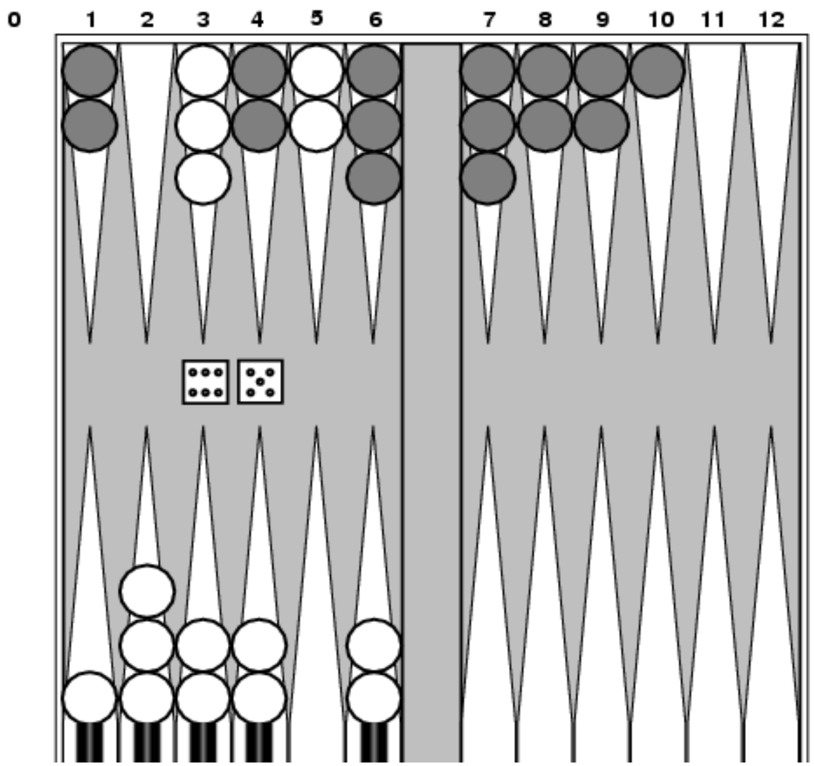
Games that include chance

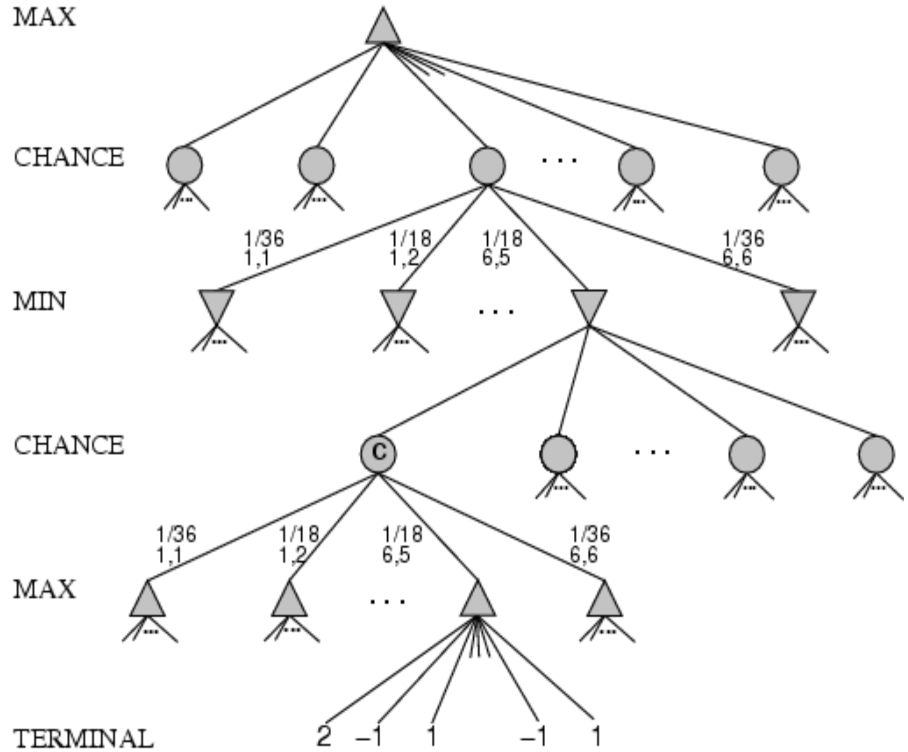




- Whites turn, After rolling a 5 and a 6
- Possible moves (5–10,5–11), (5–11,19–24), (5–10,10–16) and (5–11,11–16)

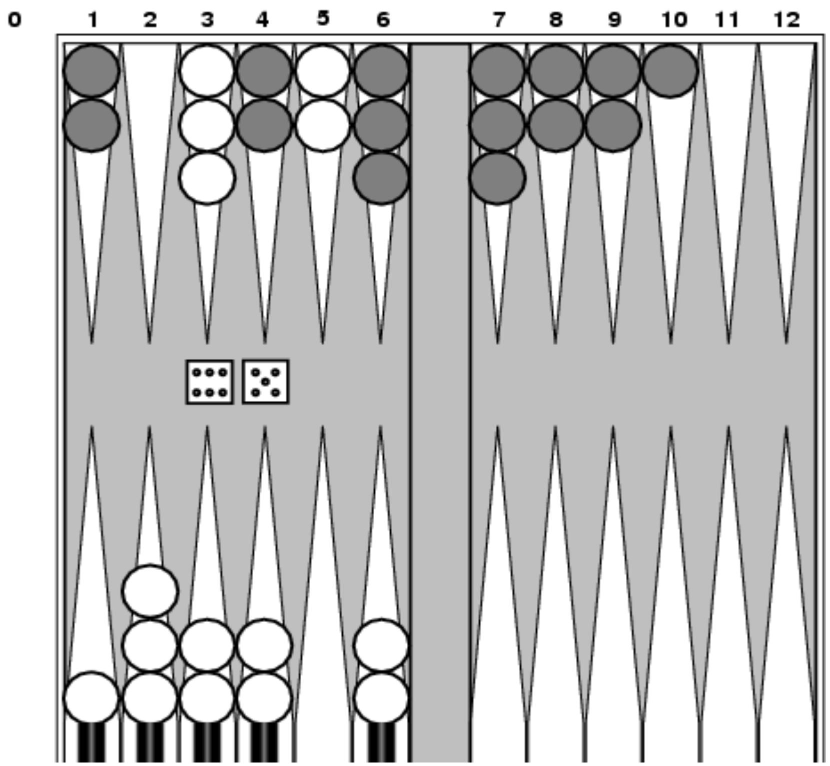
Games that include chance

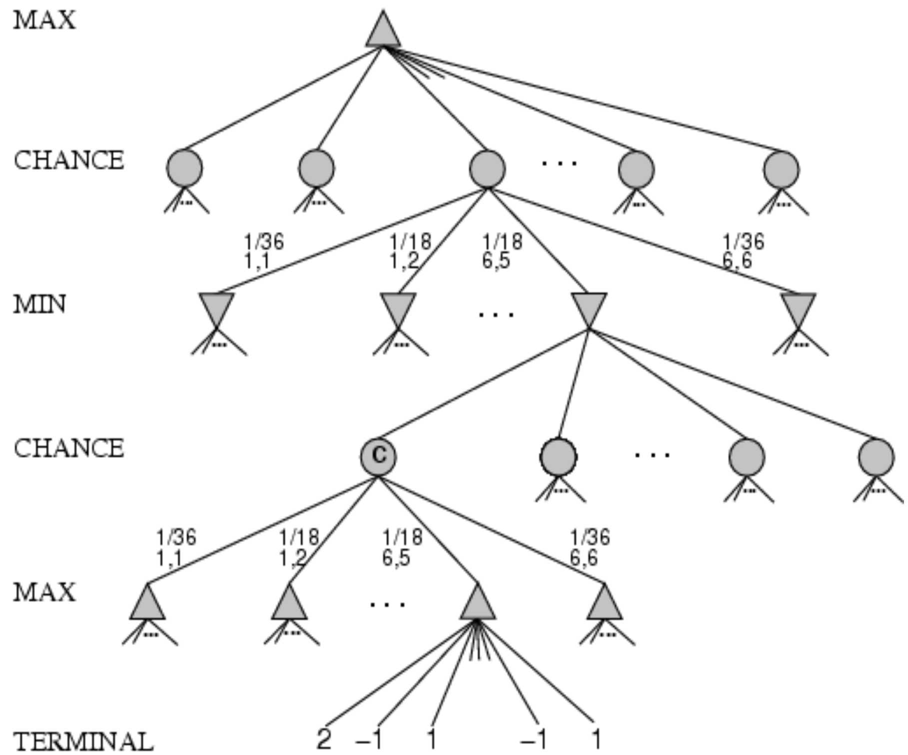




- Possible moves (5–10,5–11), (5–11,19–24), (5–10,10–16) and (5–11,11–16)

Games that include chance





- [1,1], [6,6] chance 1/36, all other chance 1/18
- Can not calculate definite minimax value, only *expected* value

Expecti minimax value

EXPECTI-MINIMAX-VALUE(n)=

UTILITY(n) If n is a terminal

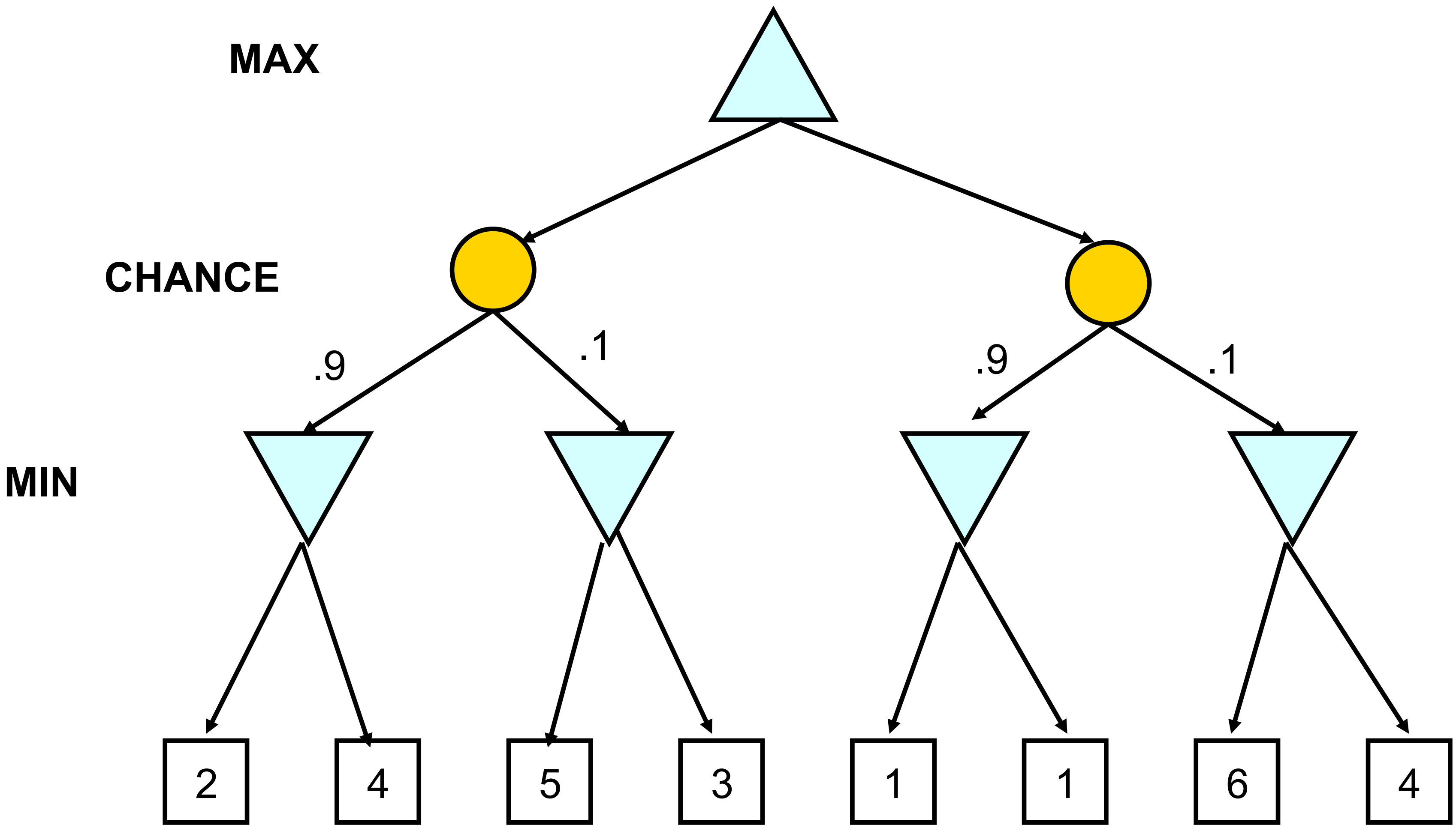
$\max_{s \in successors(n)}$ MINIMAX-VALUE(s) If n is a max node

$\min_{s \in successors(n)}$ MINIMAX-VALUE(s) If n is a min node

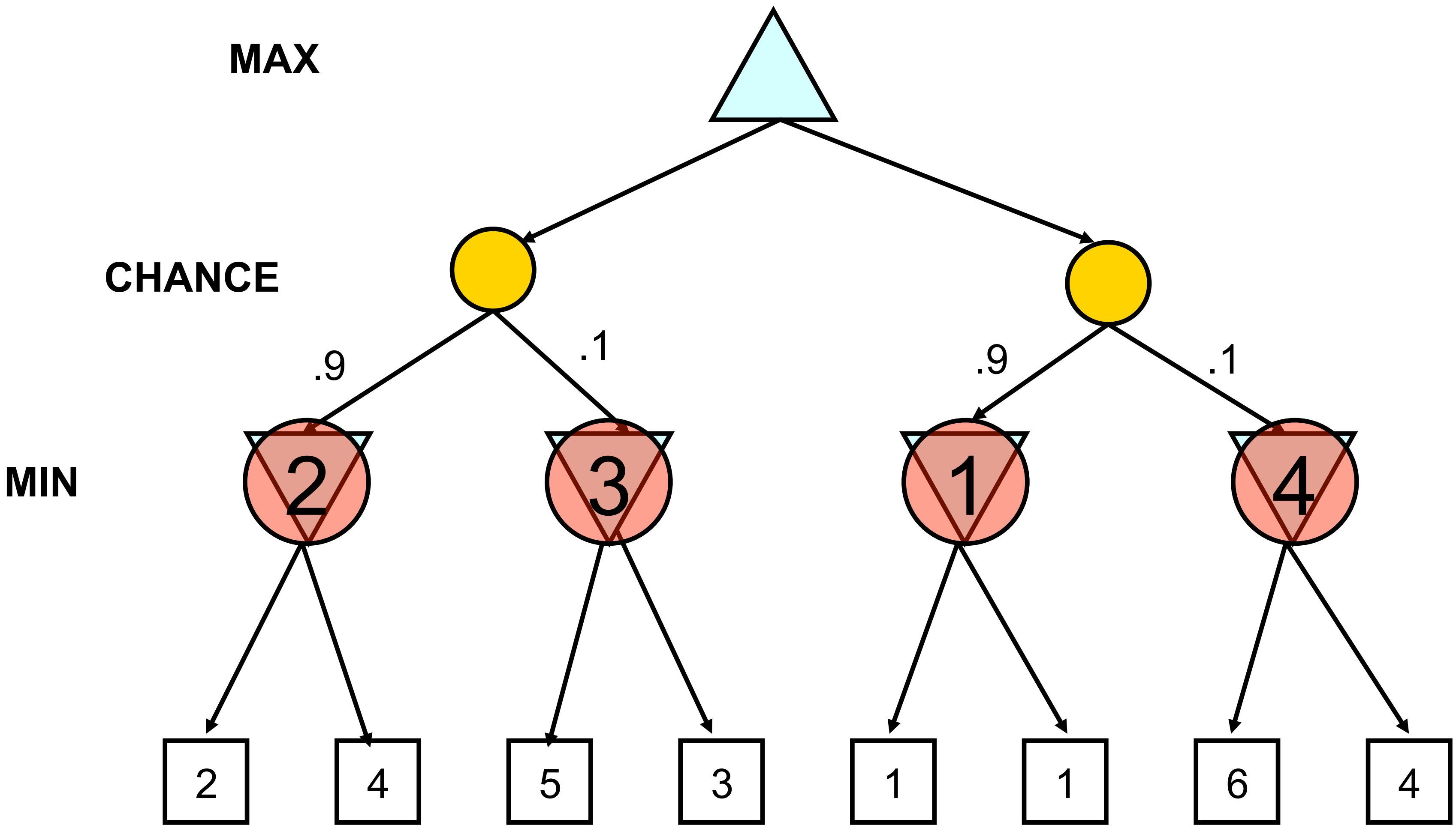
$\sum_{s \in successors(n)} P(s)$. EXPECTIMINIMAX(s) If n is a chance node

These equations can be backed-up recursively all the way to the root of the game tree.

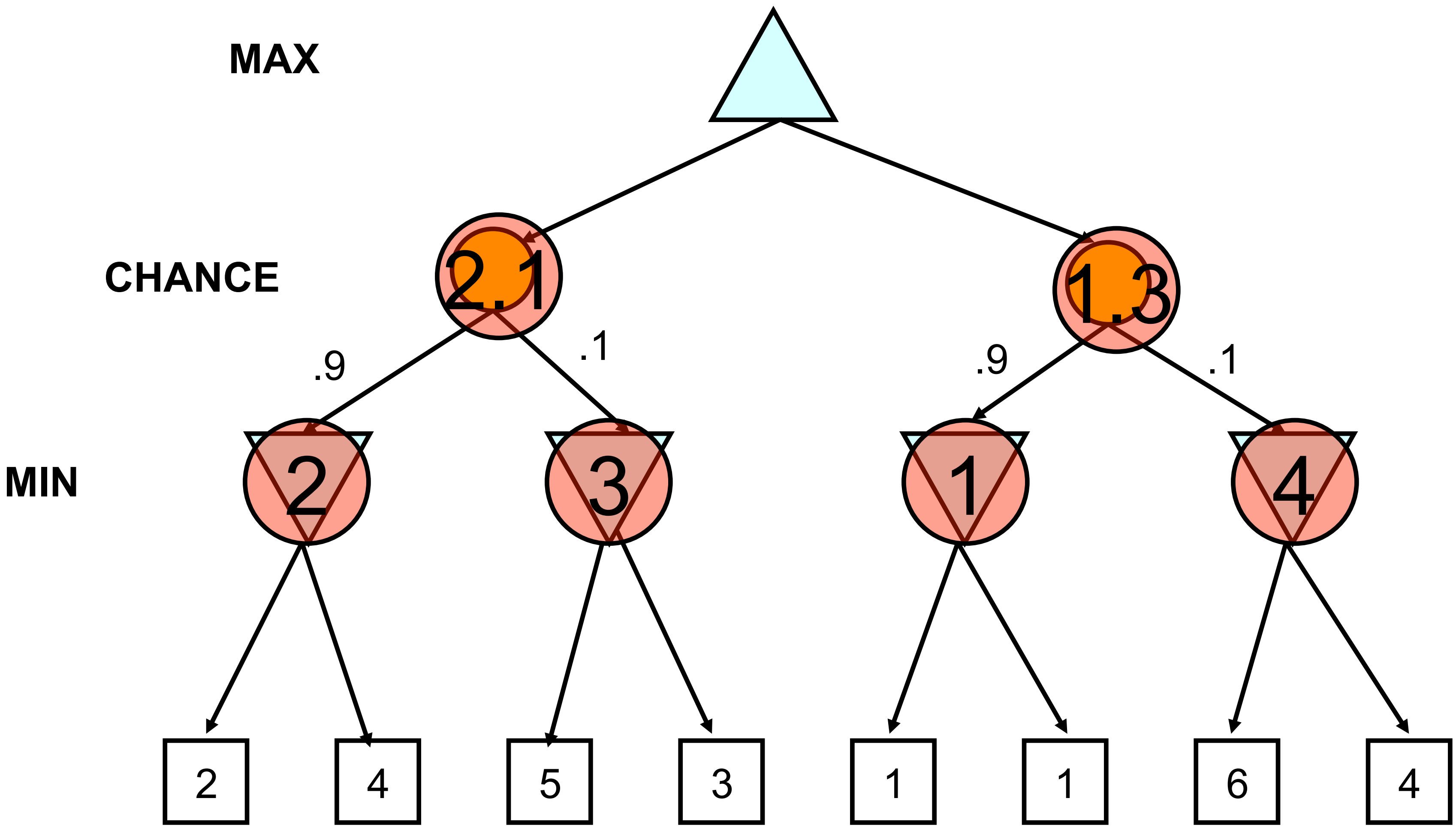
EXPECTEDMINIMAX example



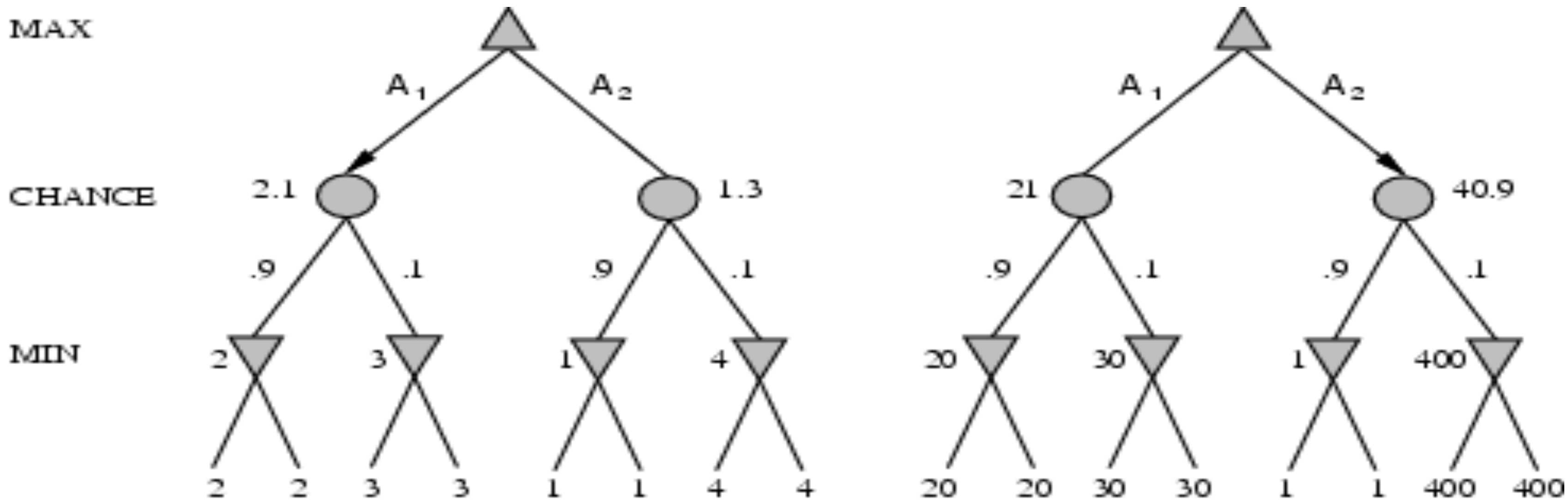
EXPECTIMINIMAX example

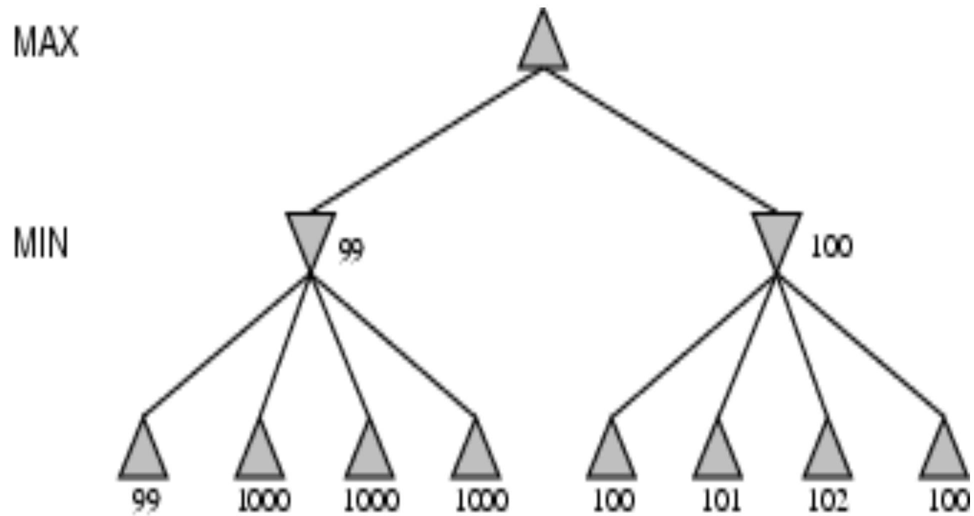


EXPECTIMINIMAX example



Position evaluation with chance nodes





- What will minimax do here?
- Is that OK?
- What might you do instead?

Learning Types

- Supervised learning:
  - (Input, output) pairs of the function to be learned can be perceived or are given.
- Unsupervised Learning:
  - No information about desired outcomes given
- Reinforcement learning:
  - Reward or punishment for actions

AUSTRALIAN VETS