COMP3411 Week 04 Tutorial

Yifan He

z5173587@unsw.edu.au

https://github.com/hharryyf/COMP3411-24T1-tutoring

Alpha-beta pruning algorithm Review

- Two-player zero-sum perfect information game
  - Sequential game (this course)
  - Simultaneous move game (Saffidine et al., 2012)
- Optimization of minimax search
- Can compute the optimal minimax value
- Time complexity: best case $O(b^{\frac{d}{2}})$
- Application
  - Chess, Deep Blue (1997)
  - General Game Playing, FluxPlayer etc. (2006)

Alpha-beta pruning algorithm Review

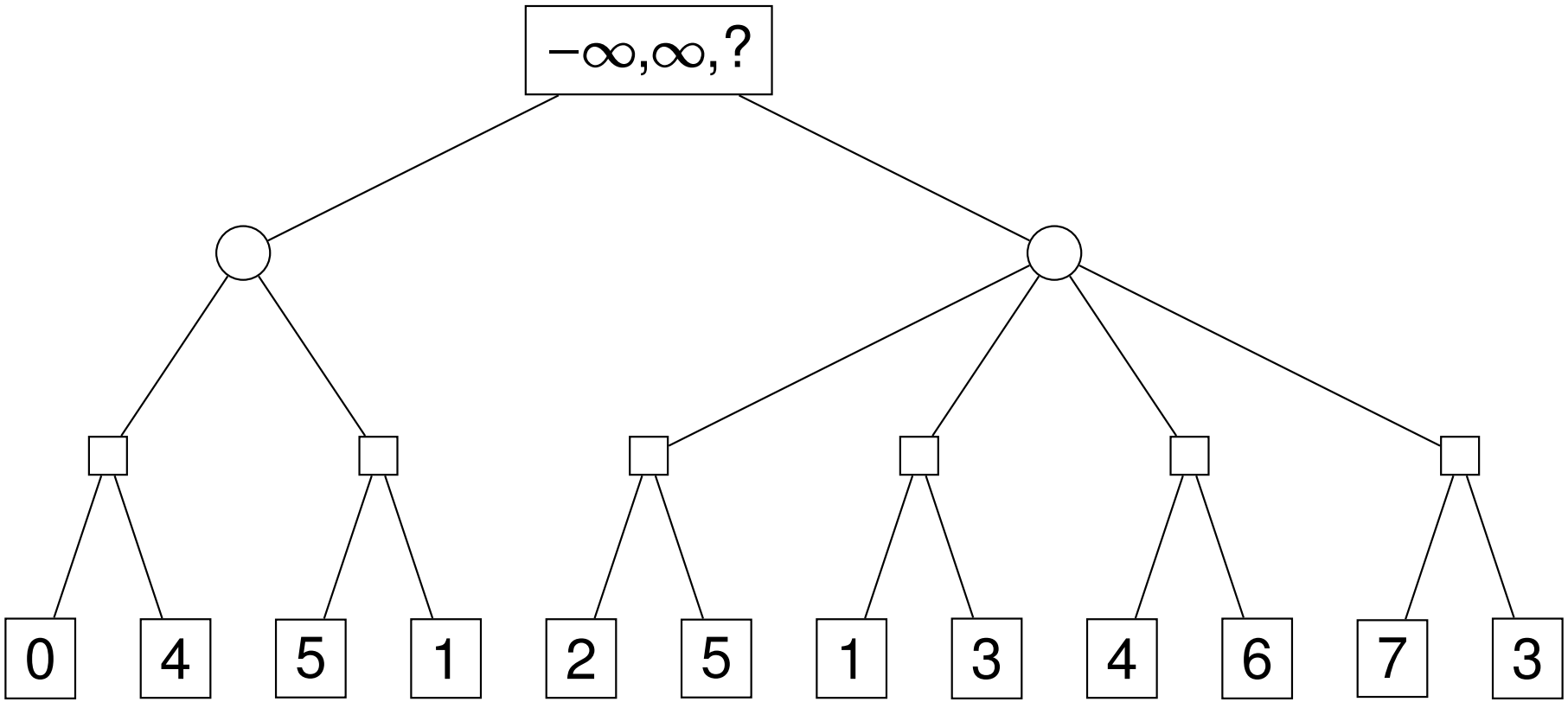
- We review the standard alpha-beta pruning
- In practice, negamax + transposition table
- Each node of the search tree has 3 parameters
  - *value*: the return value of the recursion.
  - α: the minimum possible score of the max player
  - β: the maximum possible score of the min player
- Initially $\alpha = -\infty$ and $\beta = \infty$

Alpha-beta pruning algorithm Review

- $alphabeta(state, \alpha = -\infty, \beta = \infty, depth)$
  1. If *state* is terminal or $depth = 0$, return: heuristic score of the *max* player
  2. If it is the turn of *max*
     - For each *next* state of *state*
     - $\alpha = max(\alpha, alphabeta(next, \alpha, \beta, depth - 1))$
     - If $\alpha \geq \beta$ return α
  3. return α
  4. If it is the turn of *min*
     - For each *next* state of *state*
     - $\beta = min(\beta, alphabeta(next, \alpha, \beta, depth - 1))$
     - If $\alpha \geq \beta$ return β
  5. return β

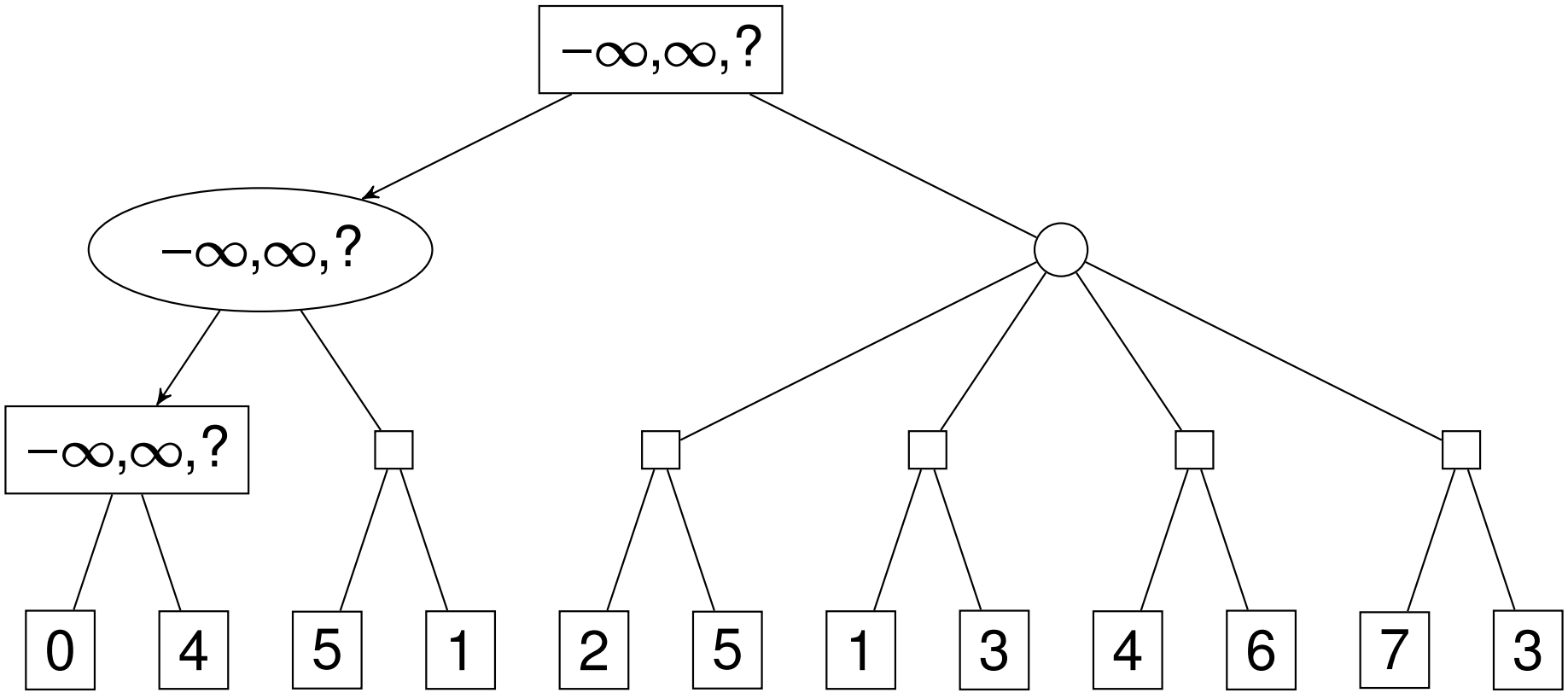
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



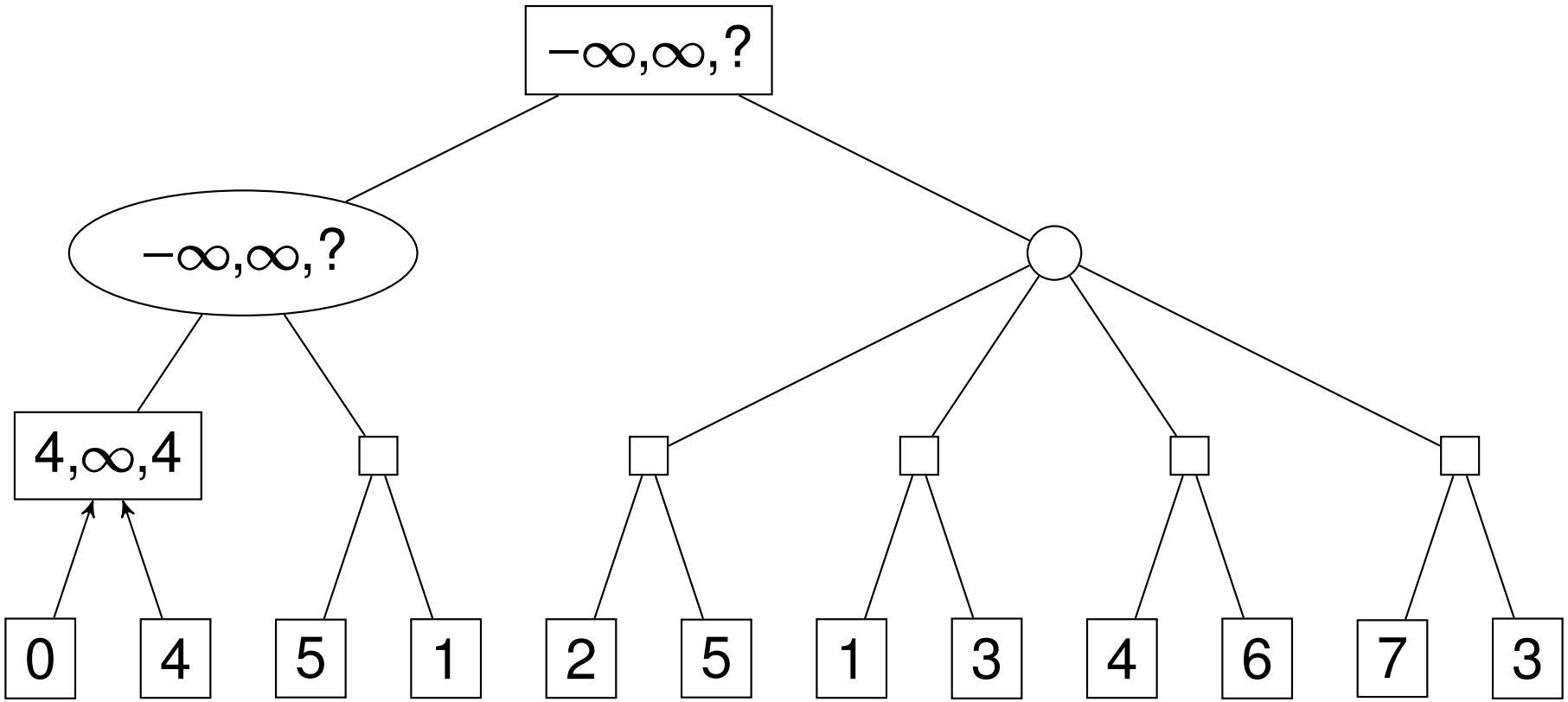
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



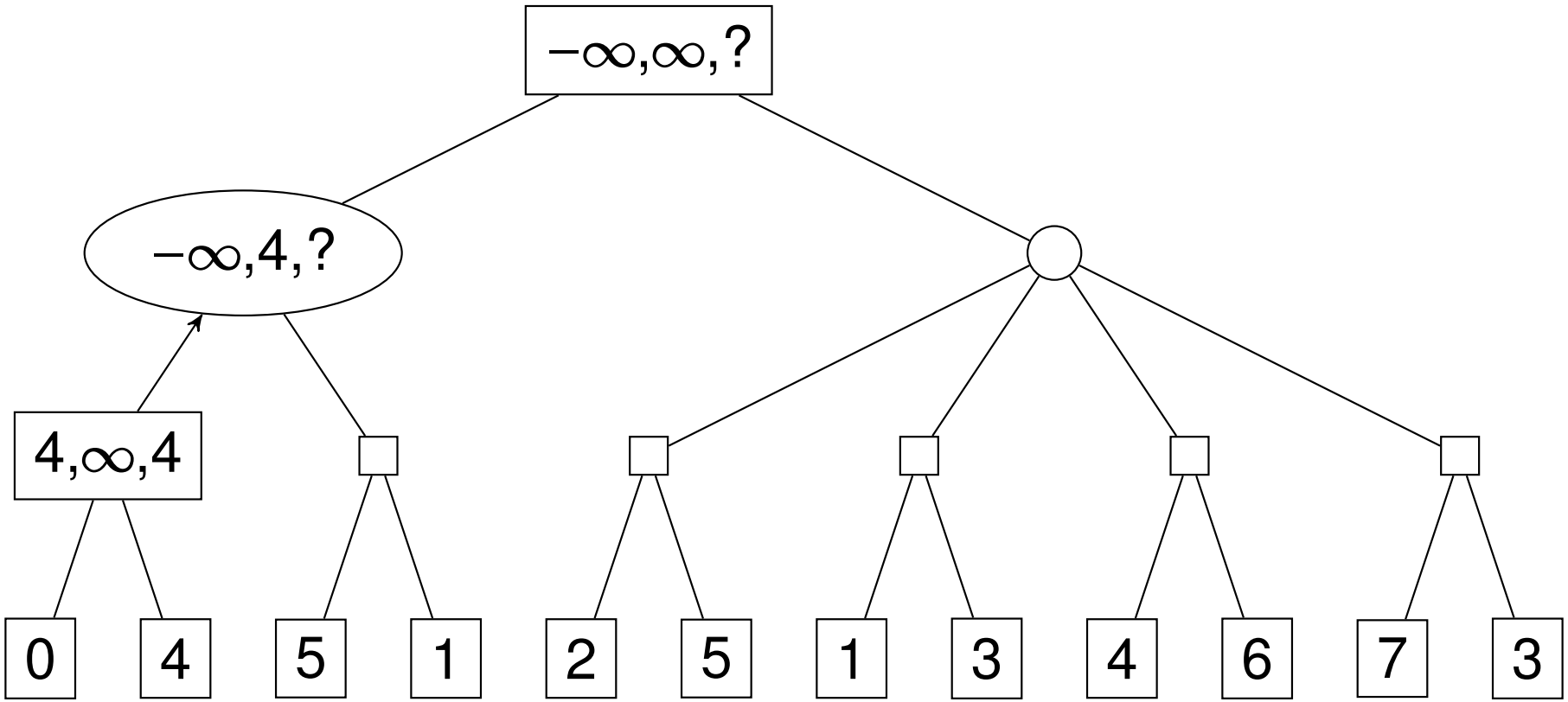
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



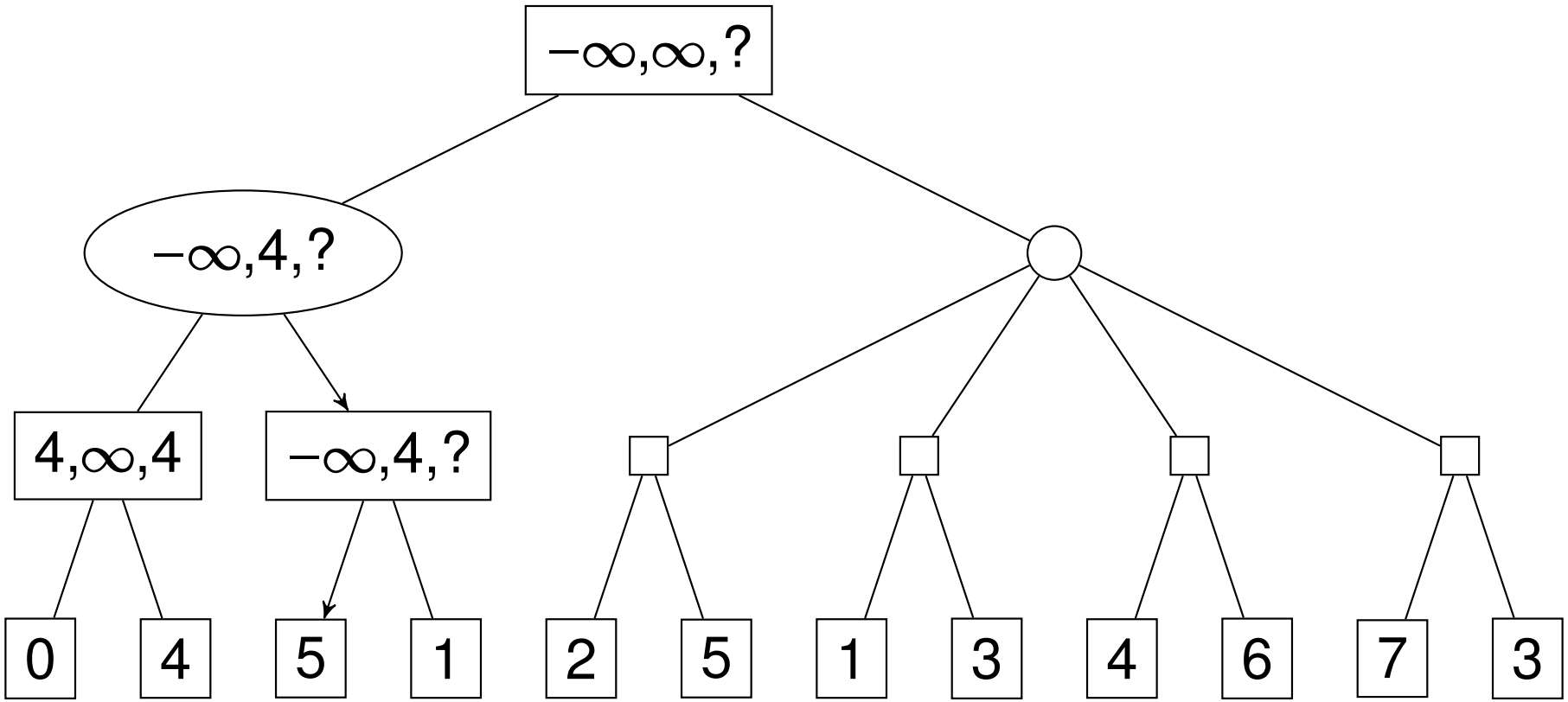
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



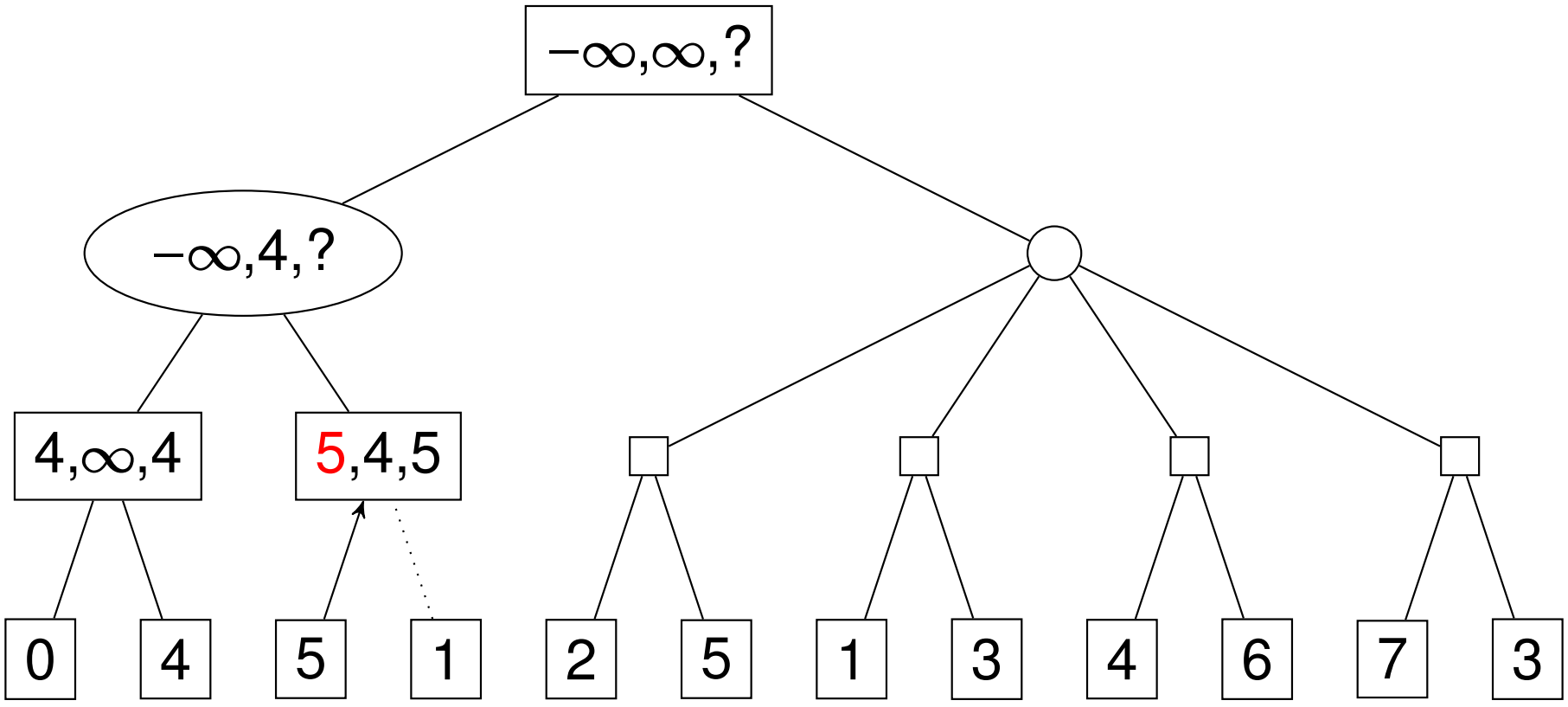
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



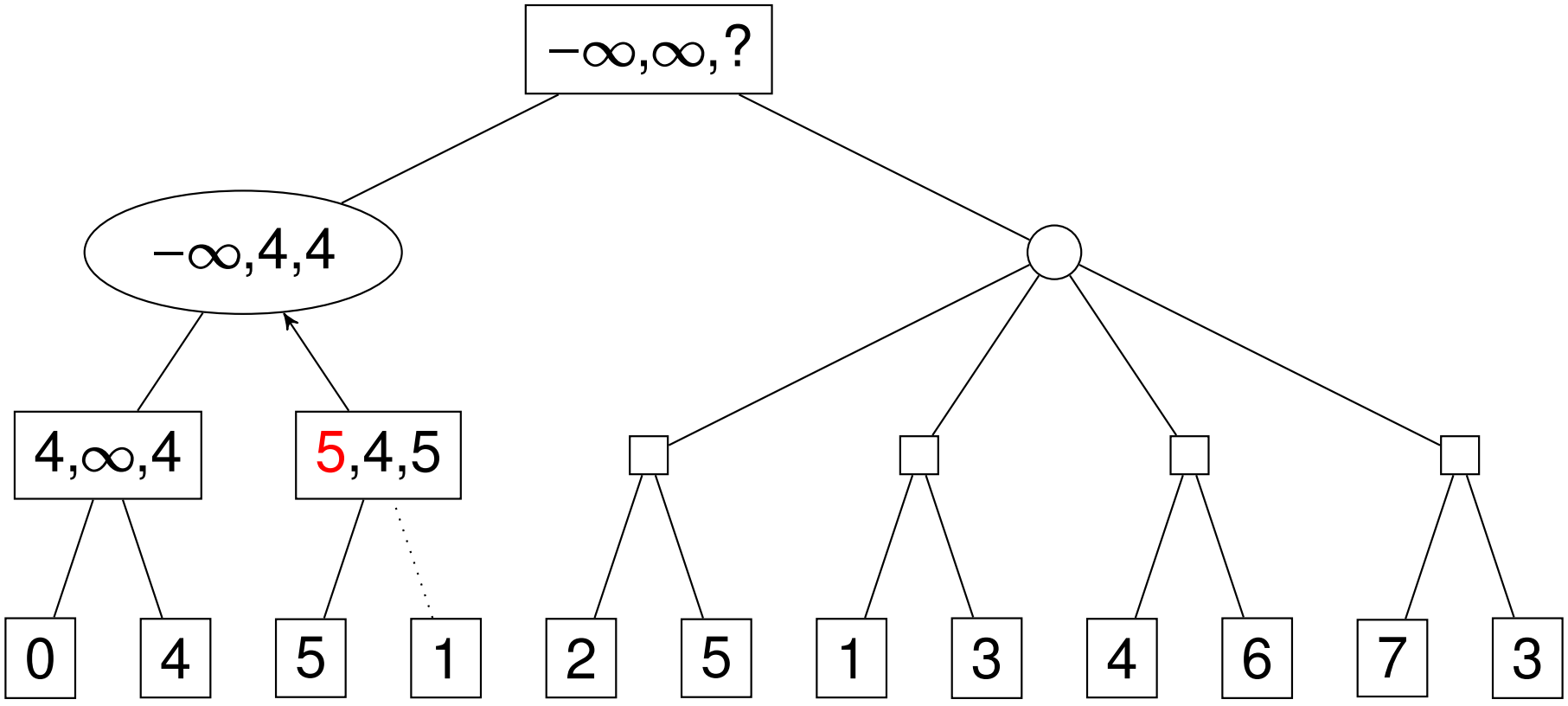
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



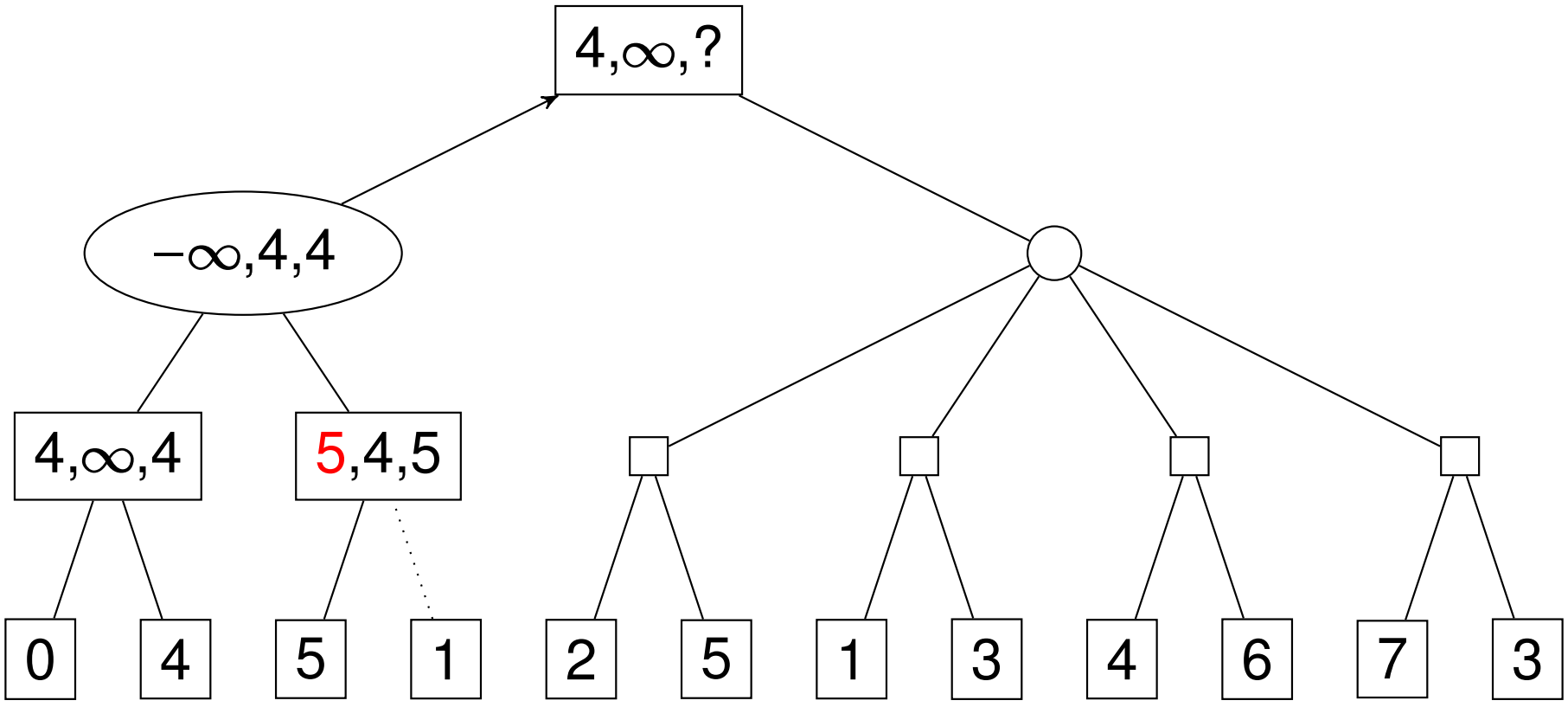
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



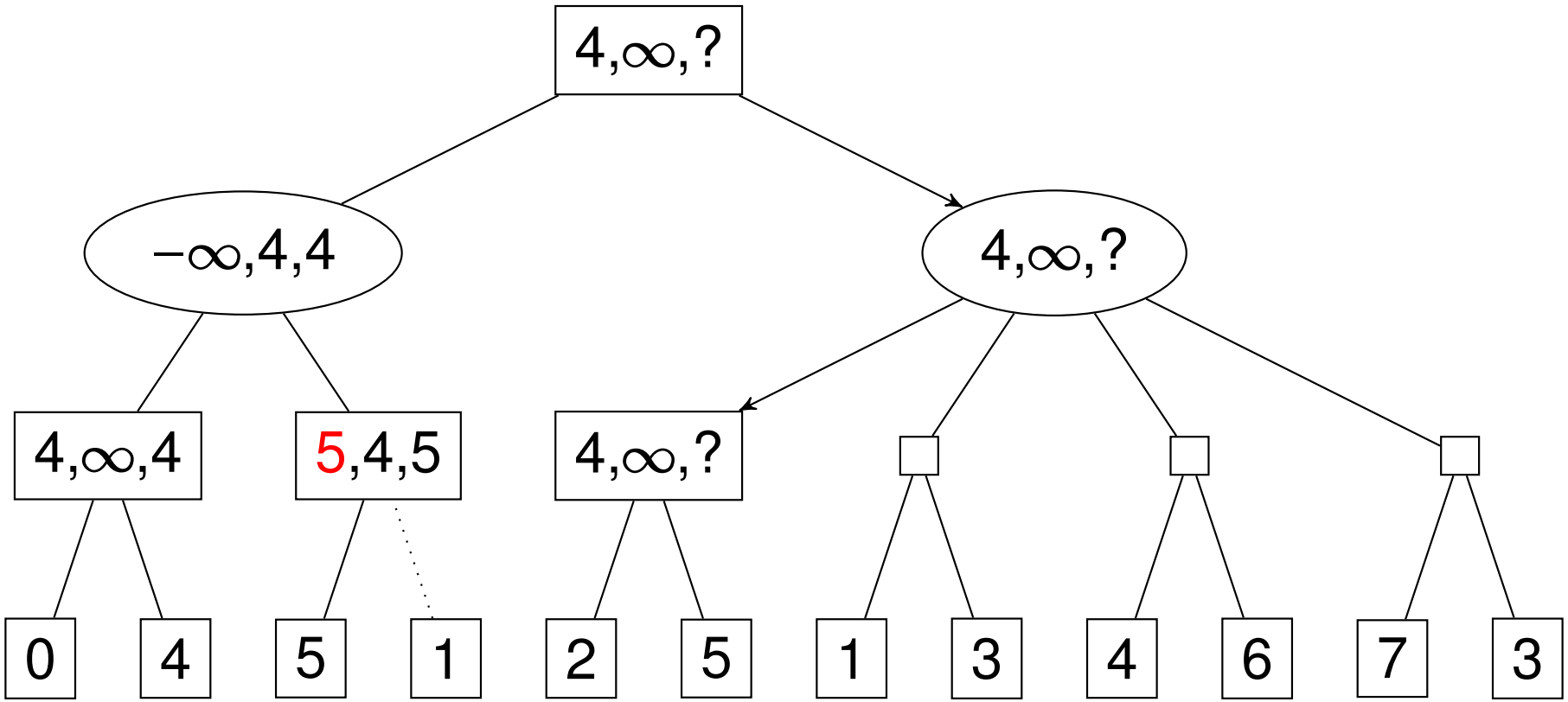
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



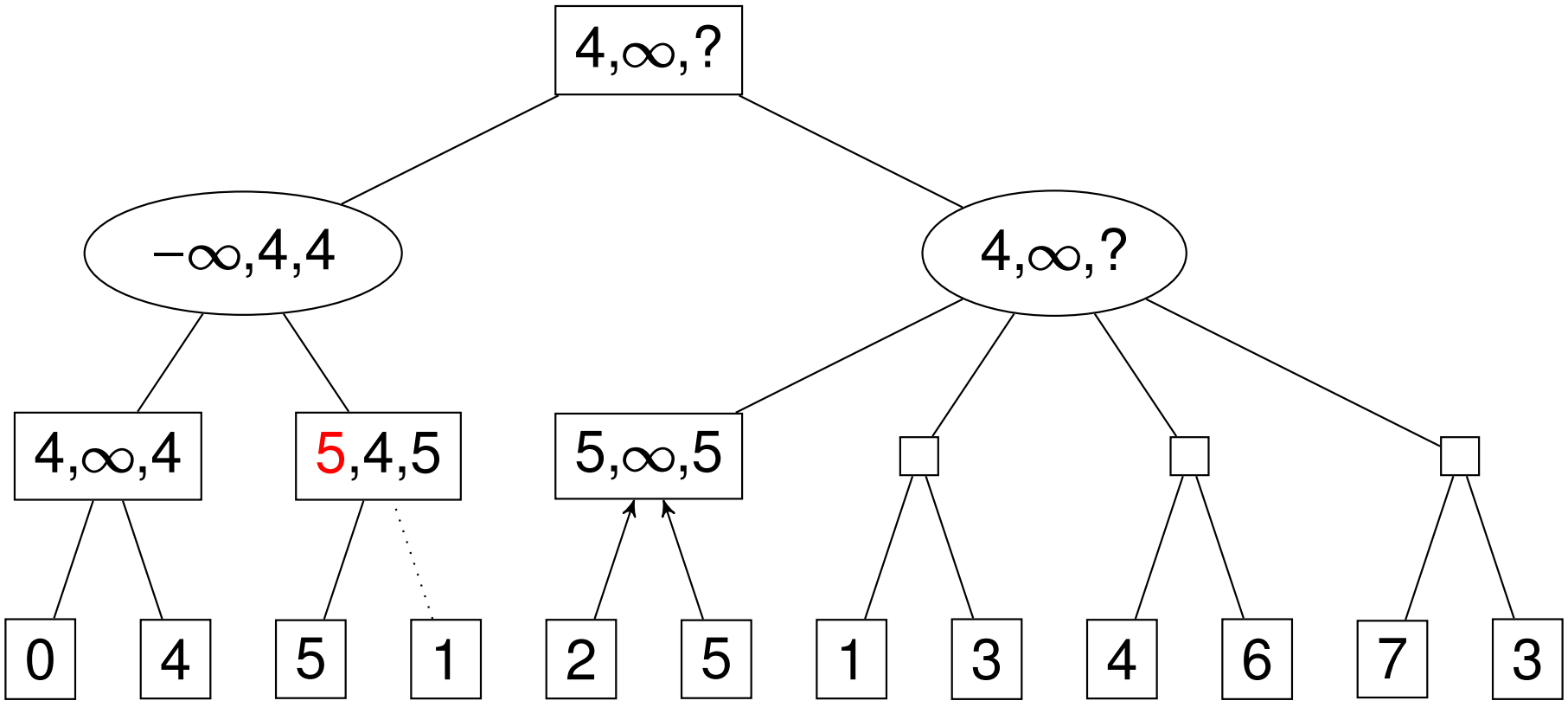
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



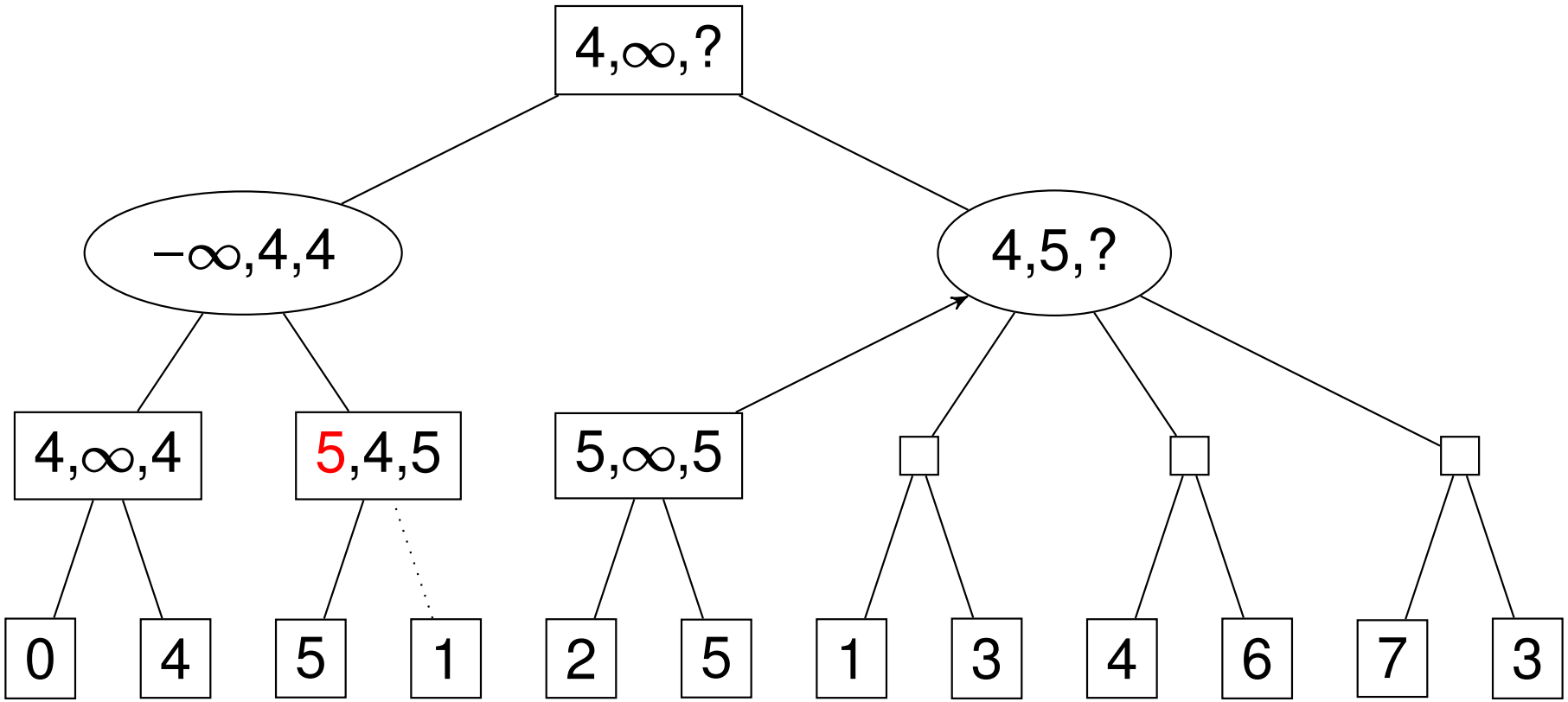
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



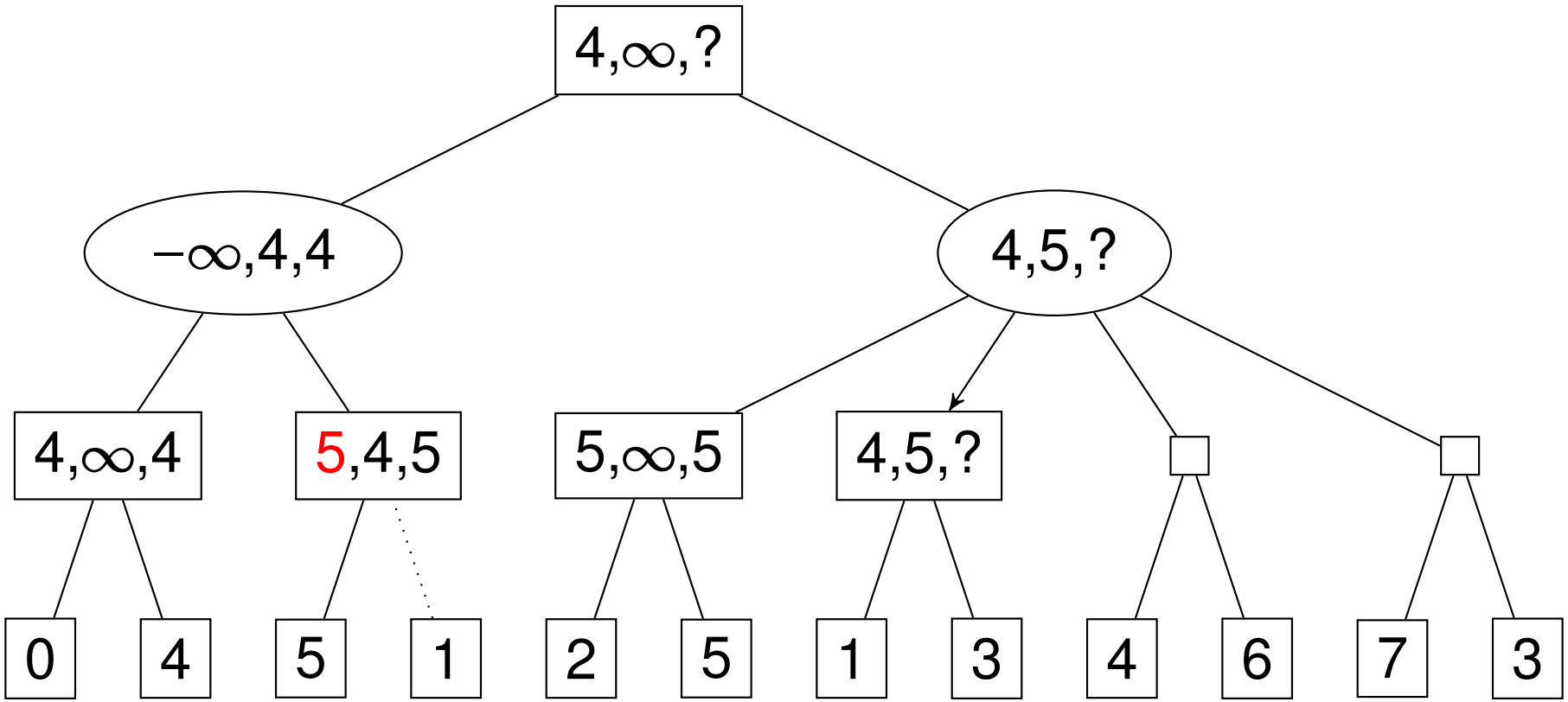
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



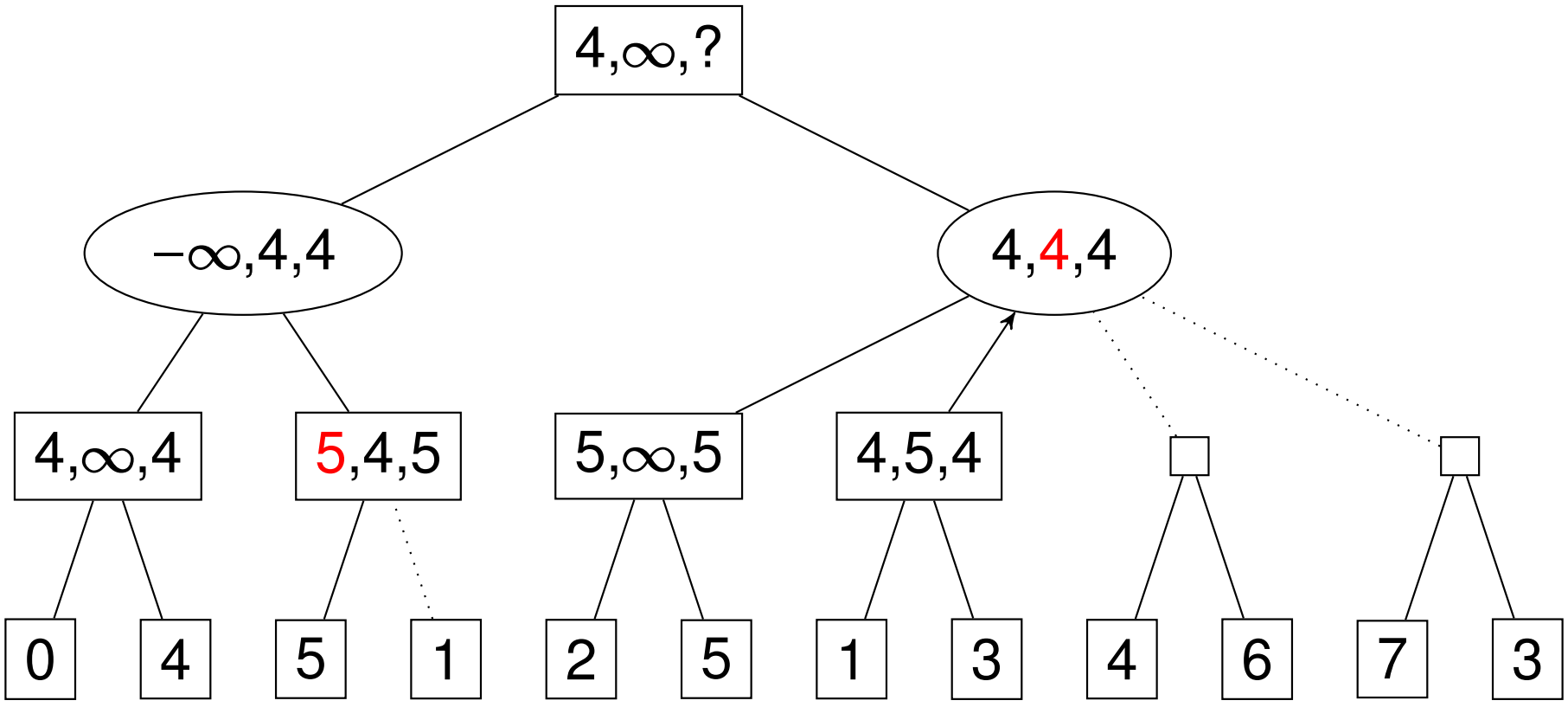
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



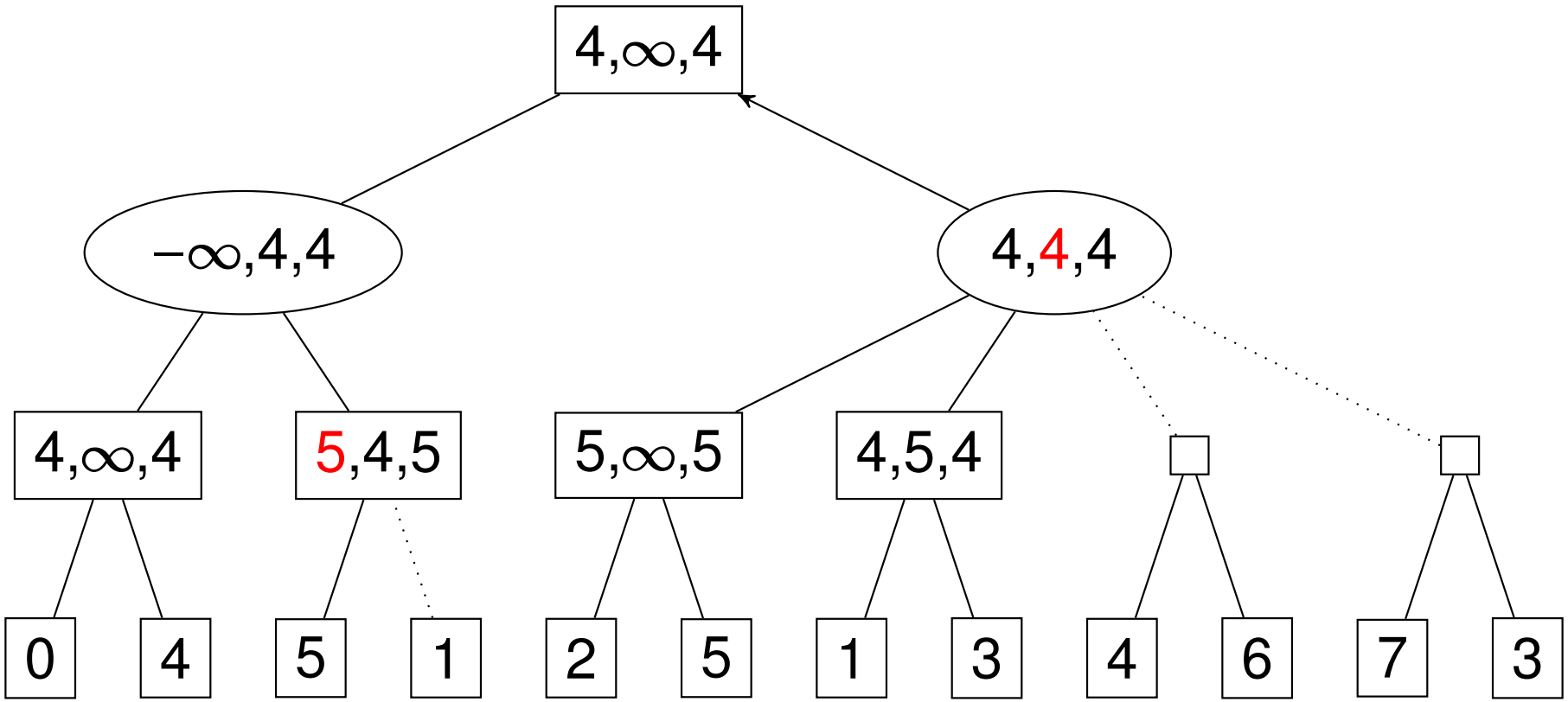
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



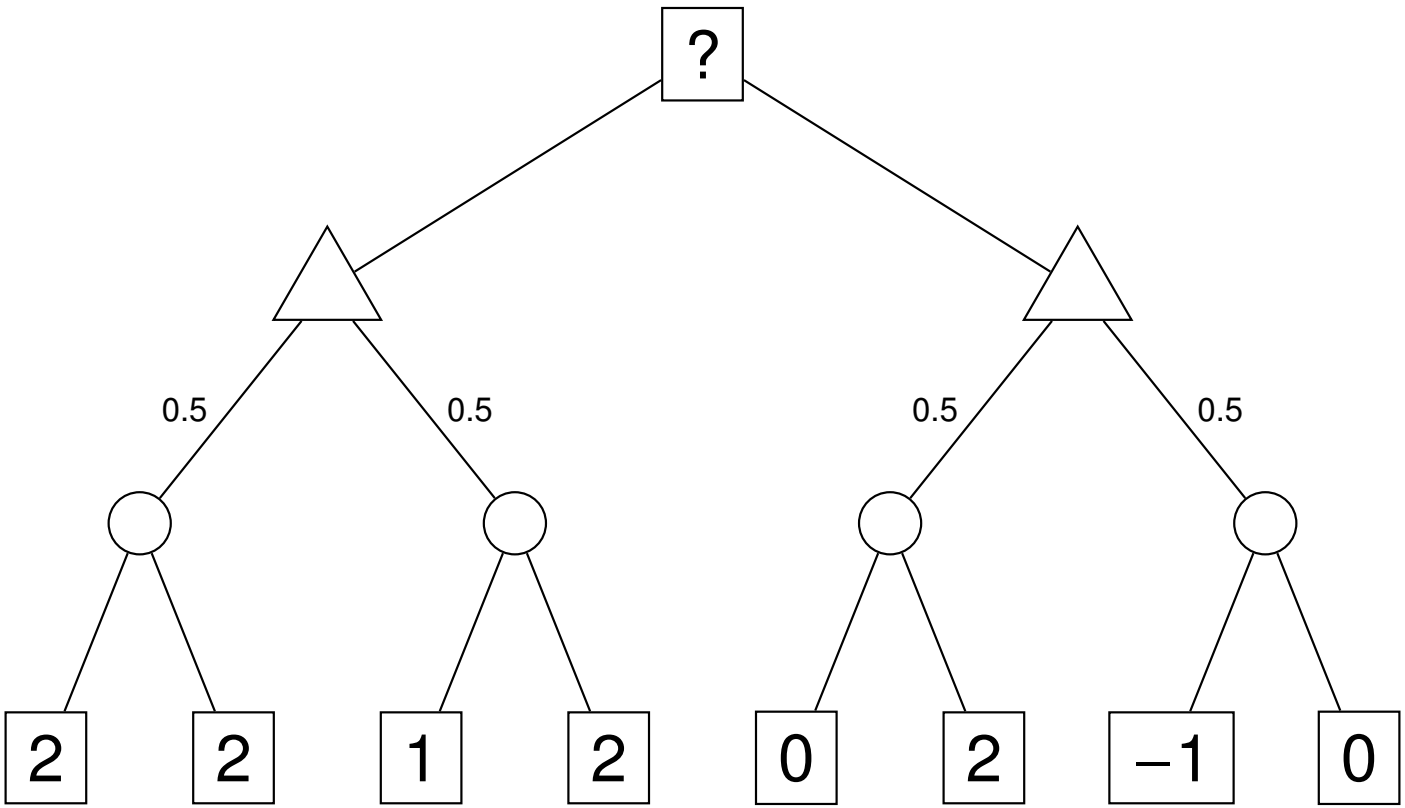
Q1 Trace alpha-beta pruning

- Notation: box for *max*, circle for *min*, $(\alpha, \beta, value)$



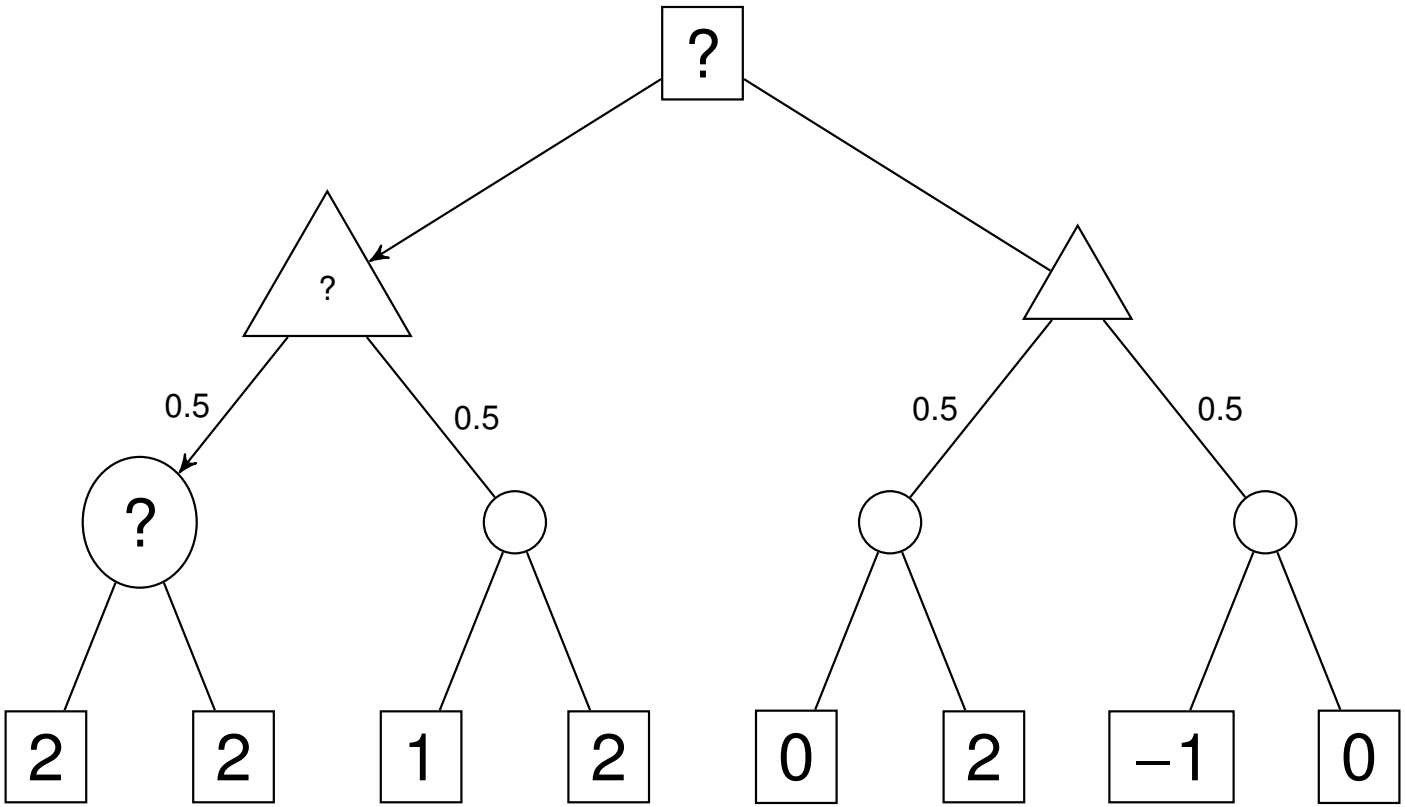
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



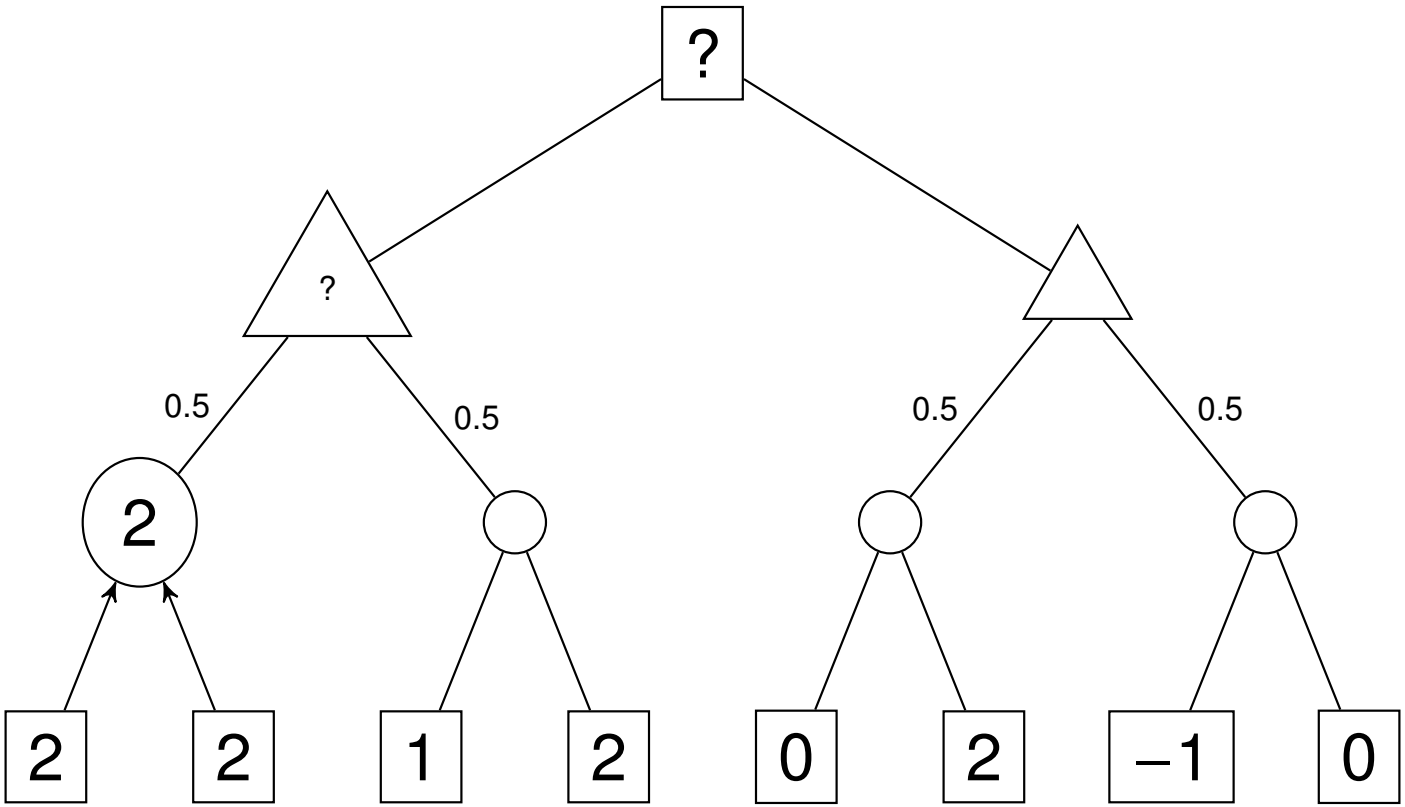
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



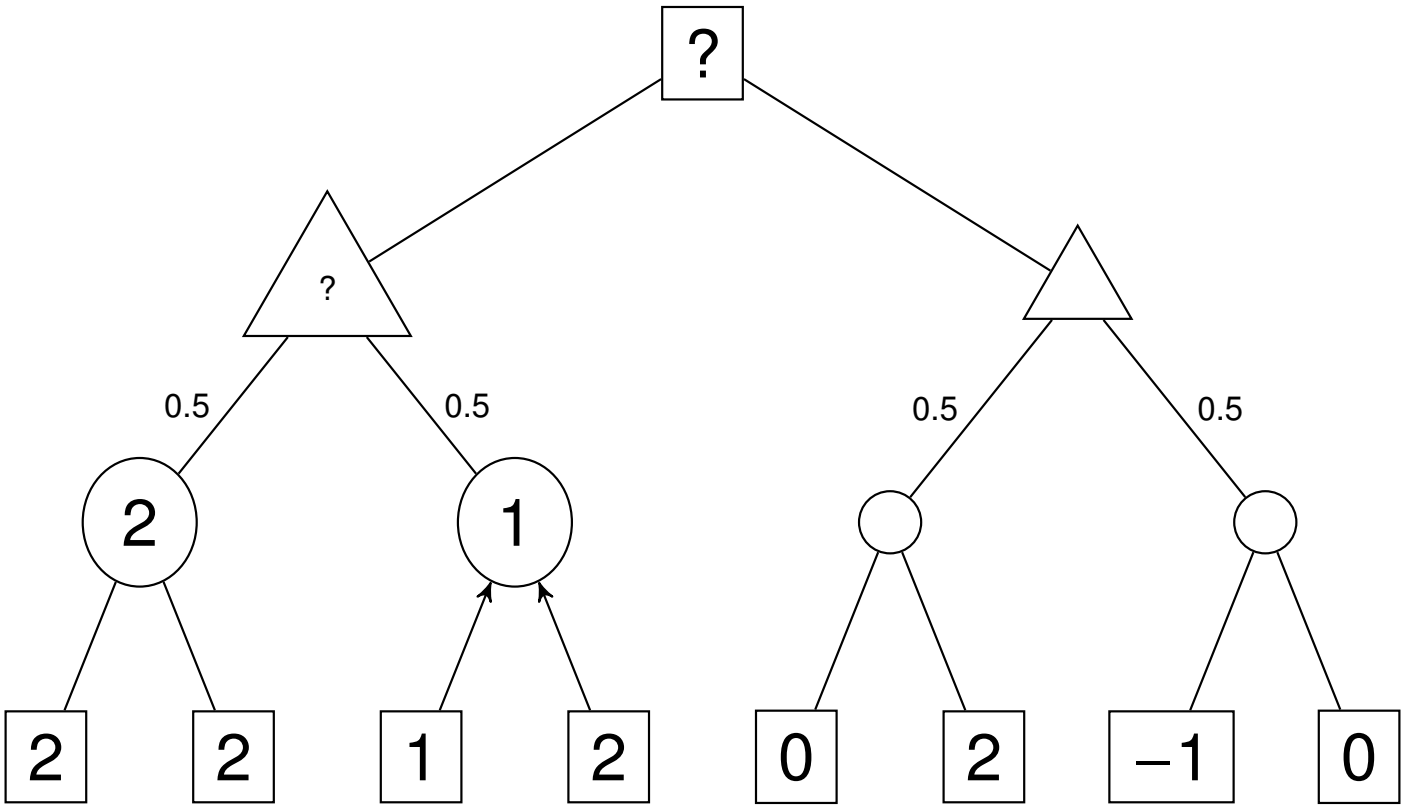
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



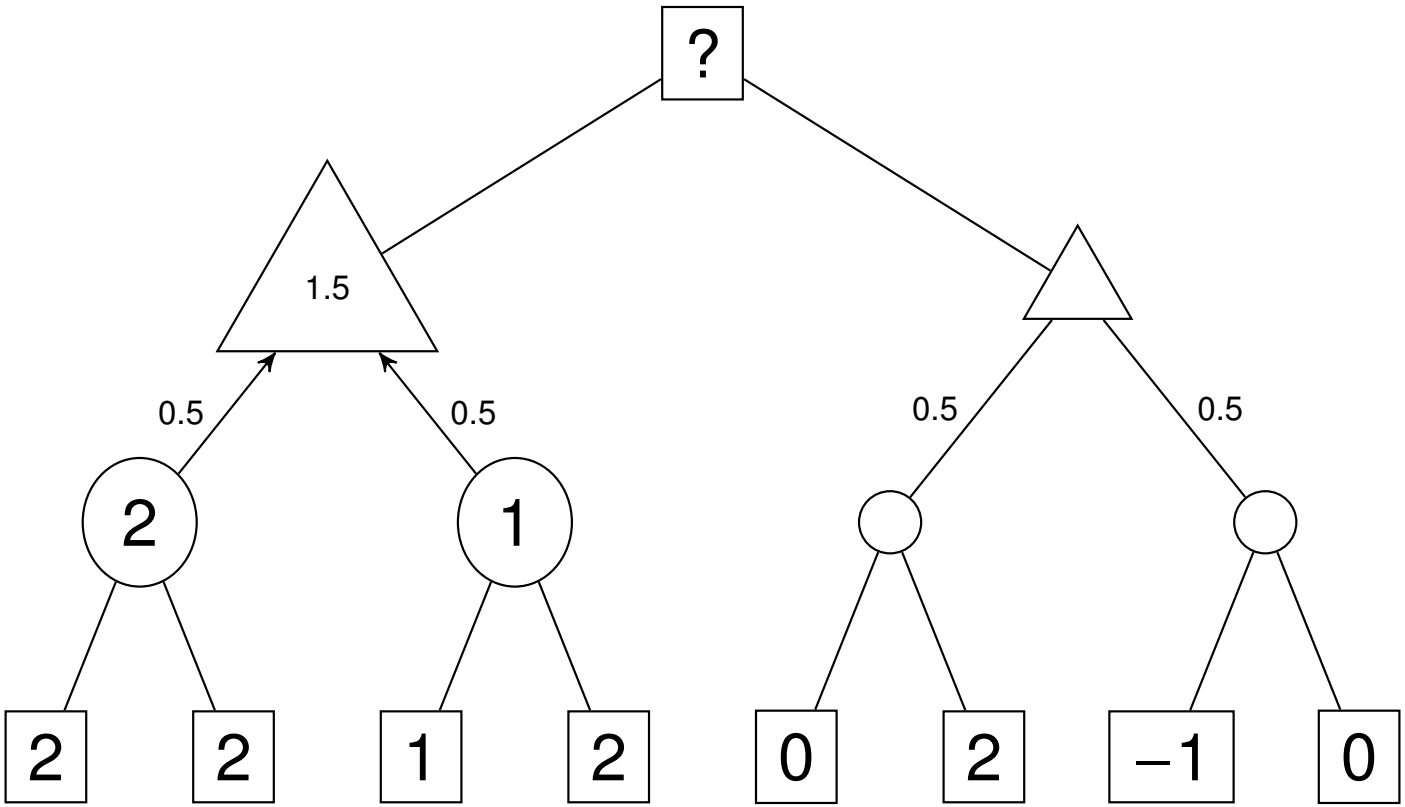
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



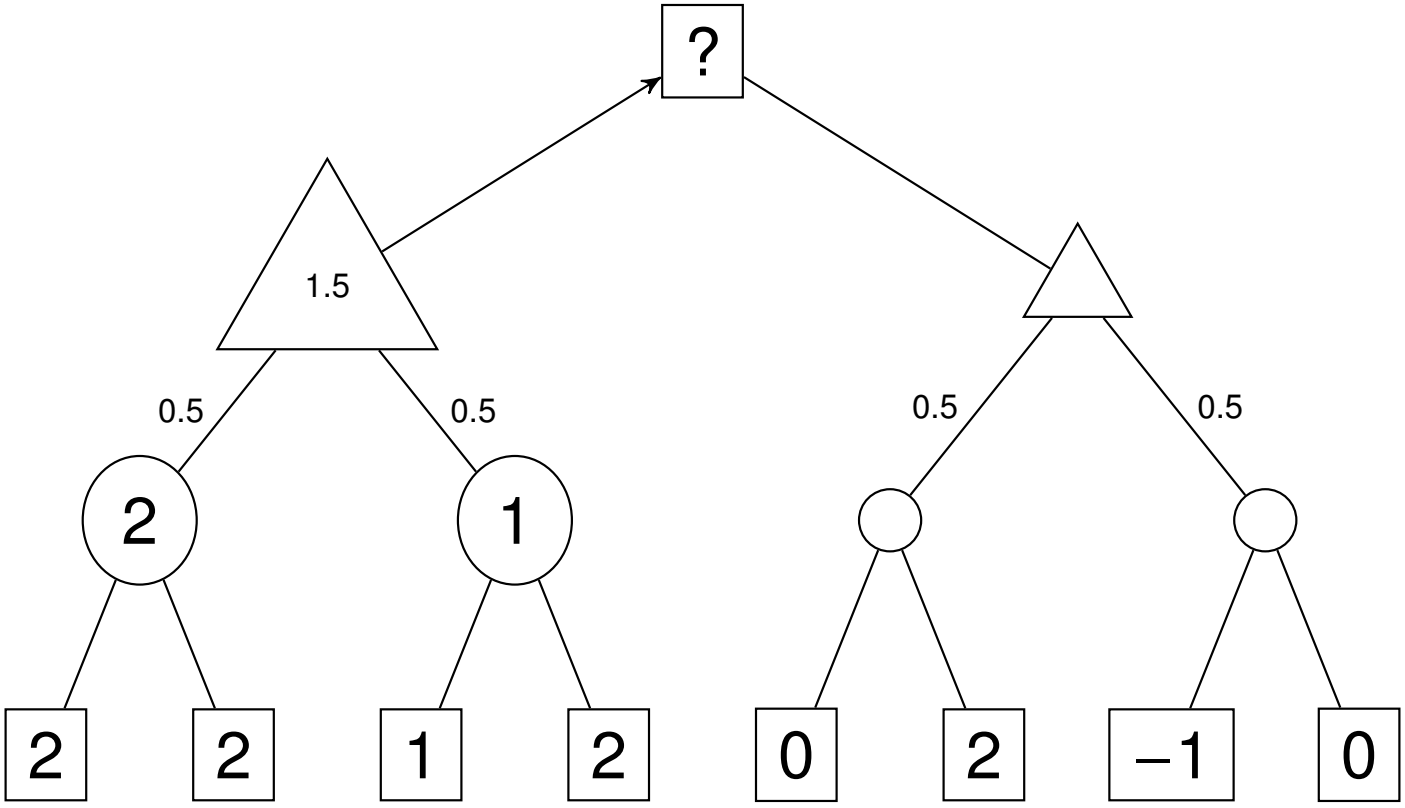
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



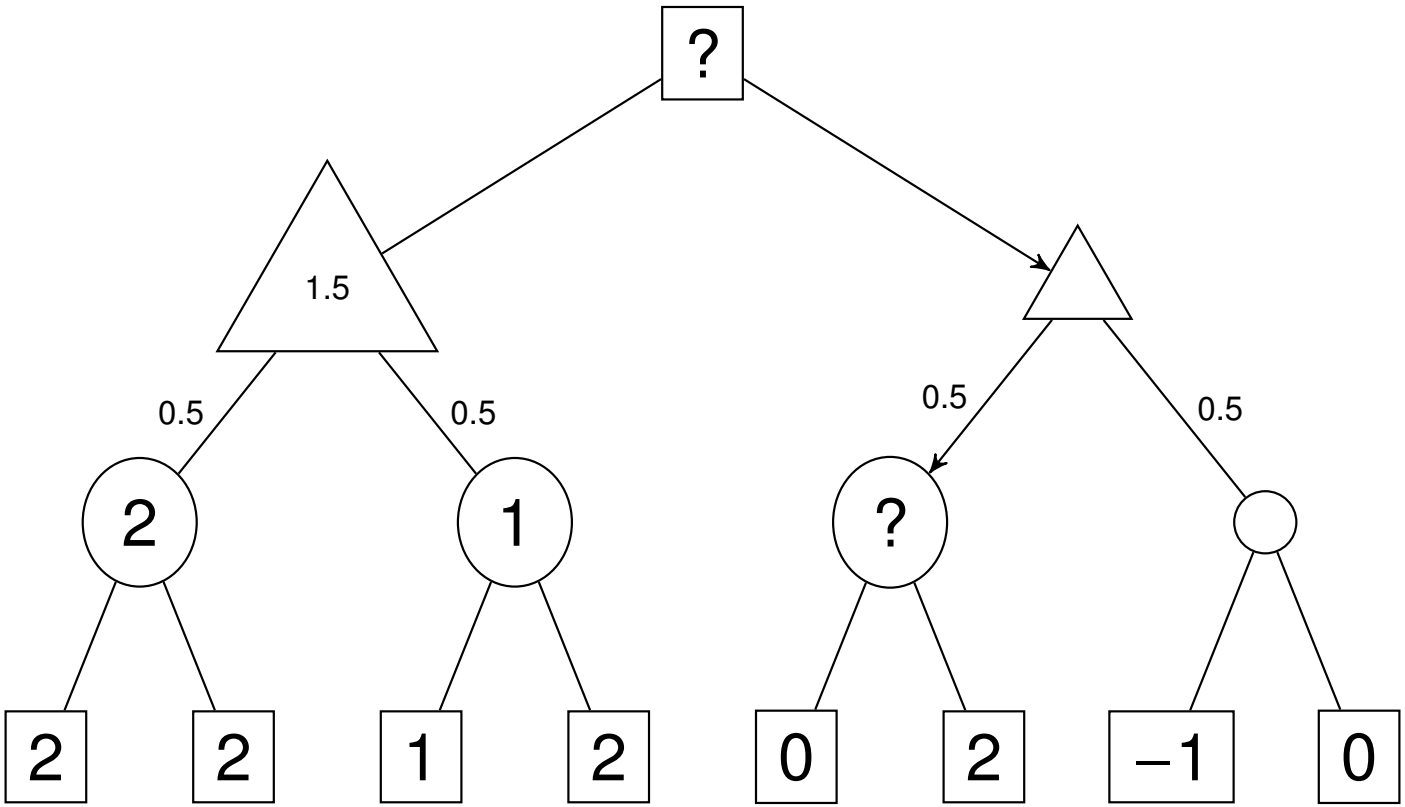
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



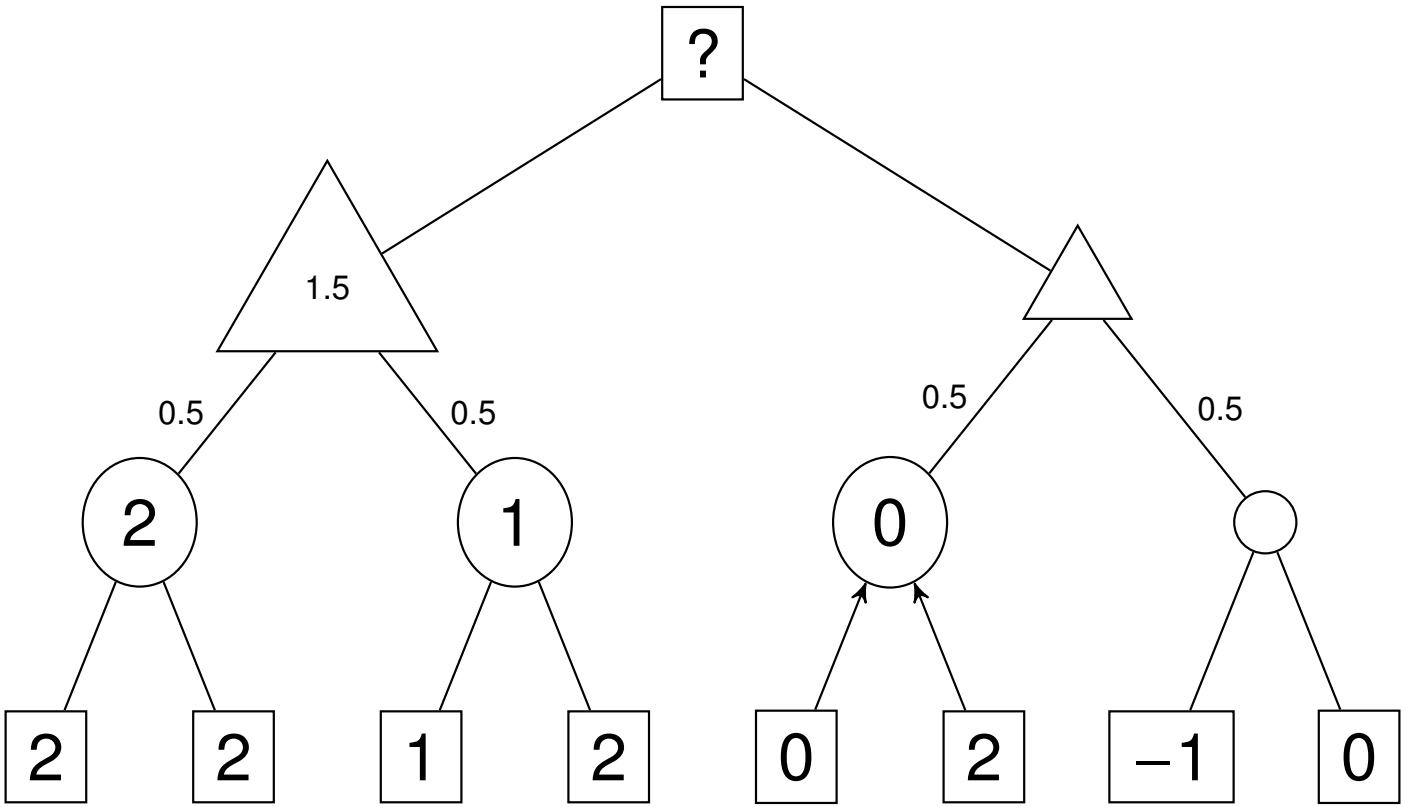
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



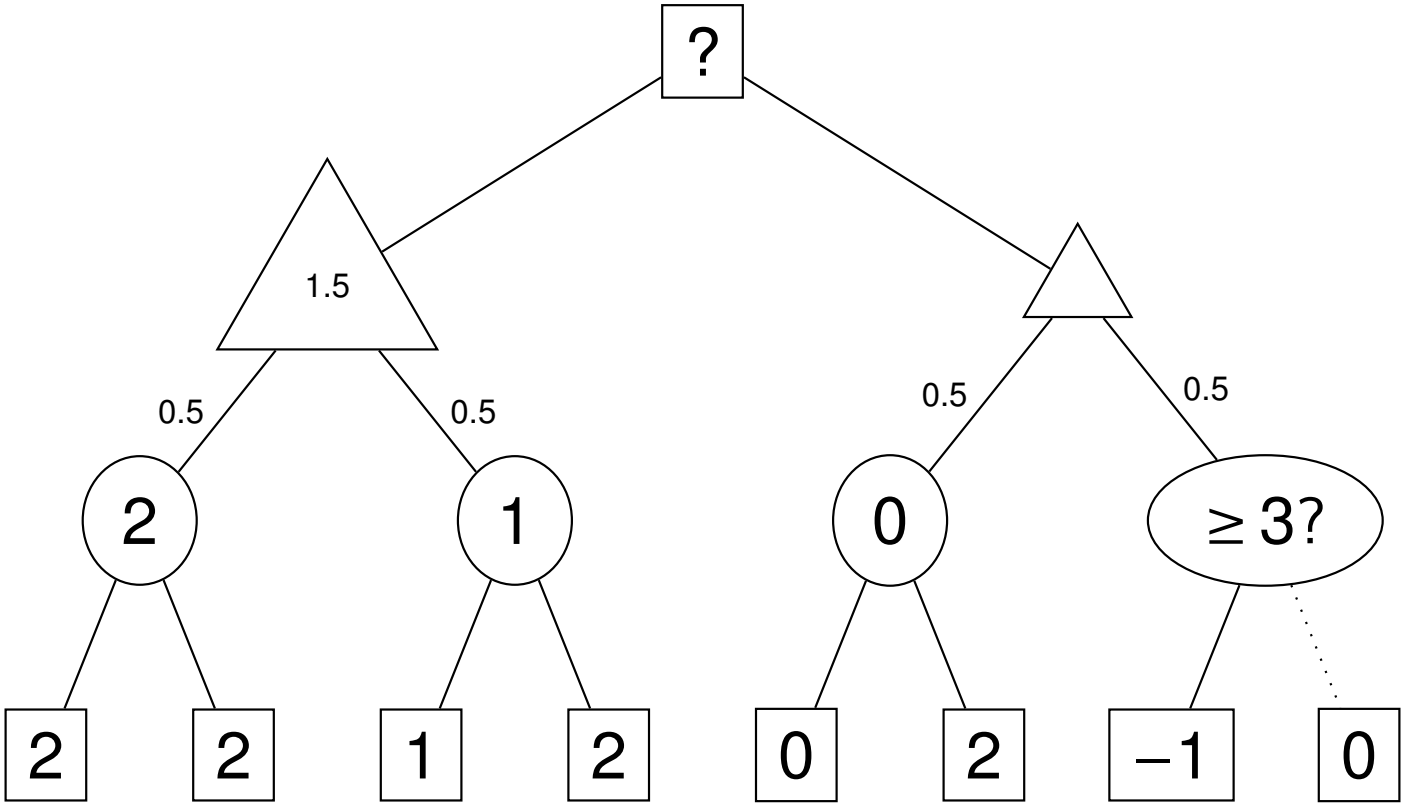
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



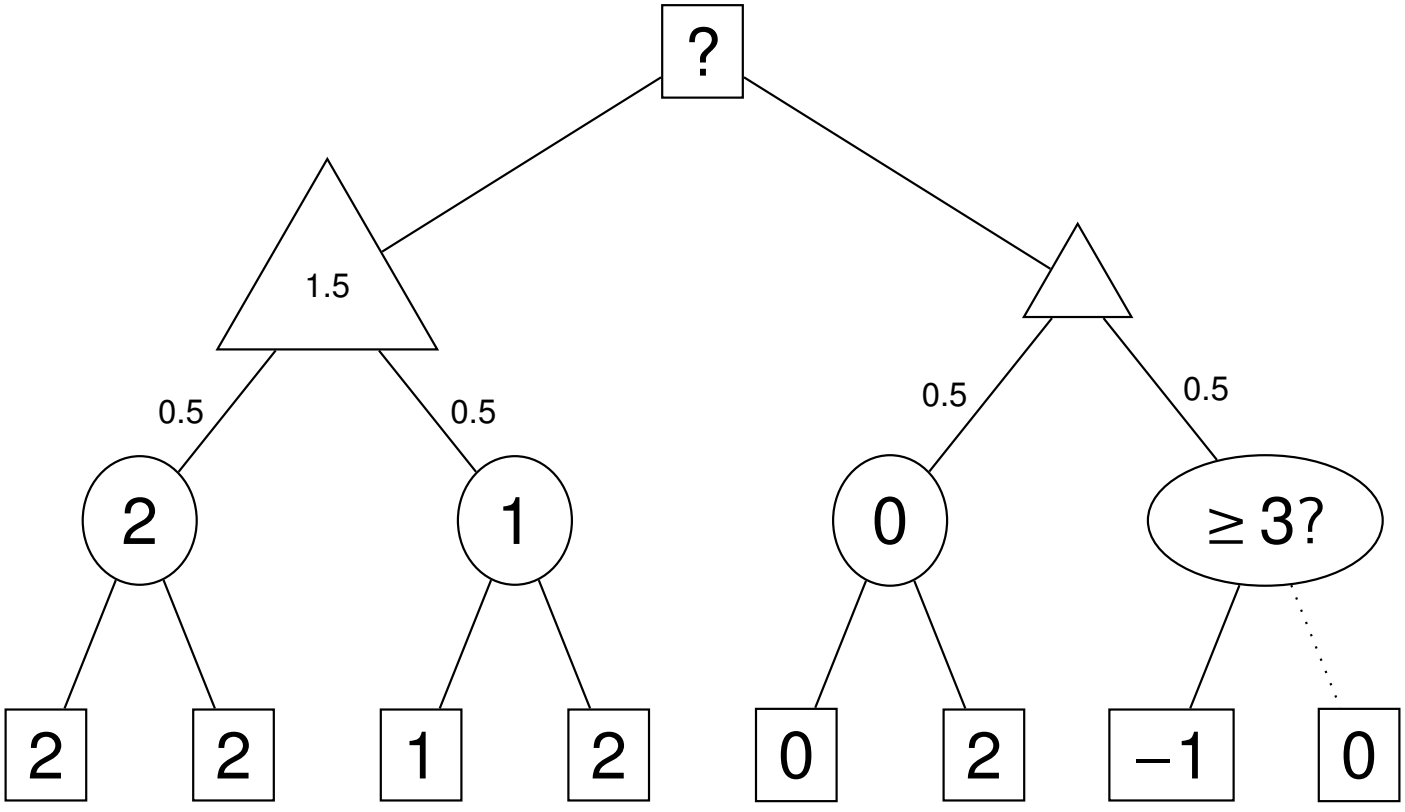
Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



Q2 Trace Expectimax

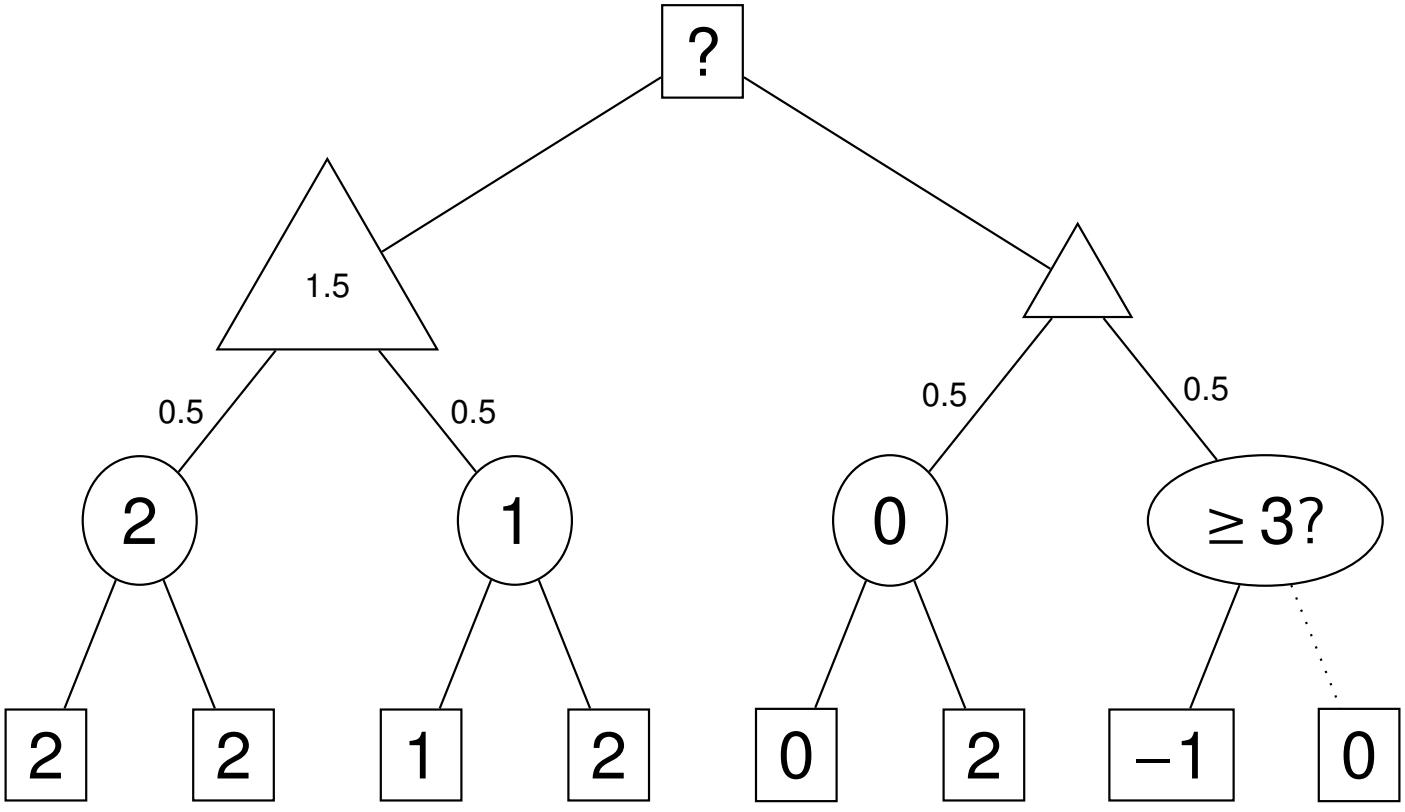
- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



- Cannot prune both the 7th and 8th node

Q2 Trace Expectimax

- Notation: box for *max*, circle for *min*, triangle for chance, (*exact expected value*)



- Cannot prune both the 7th and 8th node
- Can prune the 8th node

Q5 Heuristic function

- Aim: the search space is too large, cannot search to the leaf
- In Minimax: how “good” the position is for the *max* player
- Game Tic-Tac-Toe:

x	o	
	x	
		o

  - 3 in a row/column/diagonal for *x* (*o*) +1000 (-1000)
  - 2 in a row/column/diagonal for *x* (*o*) +100 (-100)
  - 1 in a row/column/diagonal for *x* (*o*) +10 (-10)
  - *x* occupy the center (*o*) +5 (-5)

Q5 Heuristic function

- Aim: the search space is too large, cannot search to the leaf
- In Minimax: how “good” the position is for the *max* player
- Game Tic-Tac-Toe:

x	o	
	x	
		o

  - 3 in a row/column/diagonal for *x* (*o*) +1000 (-1000)
  - 2 in a row/column/diagonal for *x* (*o*) +100 (-100)
  - 1 in a row/column/diagonal for *x* (*o*) +10 (-10)
  - *x* occupy the center (*o*) +5 (-5)
  - 10 + 10 + 10 - 10 - 10 + 5 = 15

Q6 Endgame Database

- A large database, the winner at each position

Q6 Endgame Database

- A large database, the winner at each position
- Normally generated by increasing the number of pieces on the board

Q6 Endgame Database

- A large database, the winner at each position
- Normally generated by increasing the number of pieces on the board
- Generalized from $N-1$ pieces to N pieces

Q6 Endgame Database

- A large database, the winner at each position
- Normally generated by increasing the number of pieces on the board
- Generalized from $N-1$ pieces to N pieces
- In chess, 7 pieces were solved

Q6 Endgame Database

- A large database, the winner at each position
- Normally generated by increasing the number of pieces on the board
- Generalized from $N-1$ pieces to N pieces
- In chess, 7 pieces were solved
- Benefit: early stopping for forward search