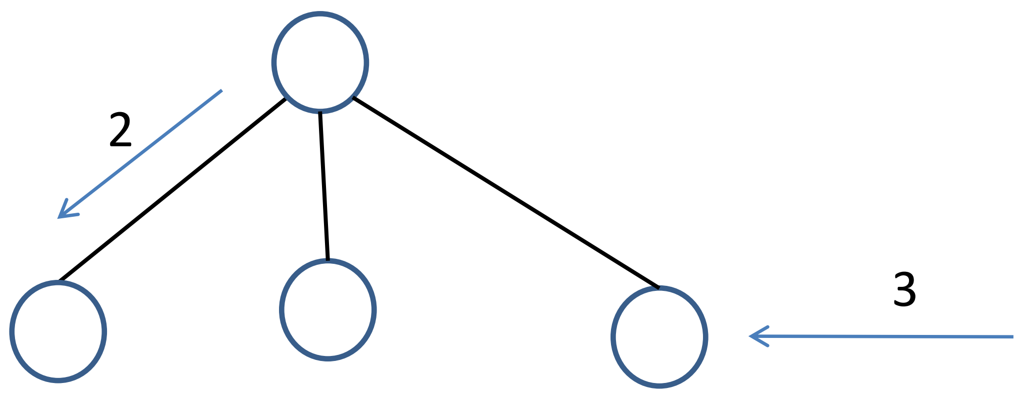
Artificial Intelligence Fundamentals

Games, Minimax and Alpha-Beta

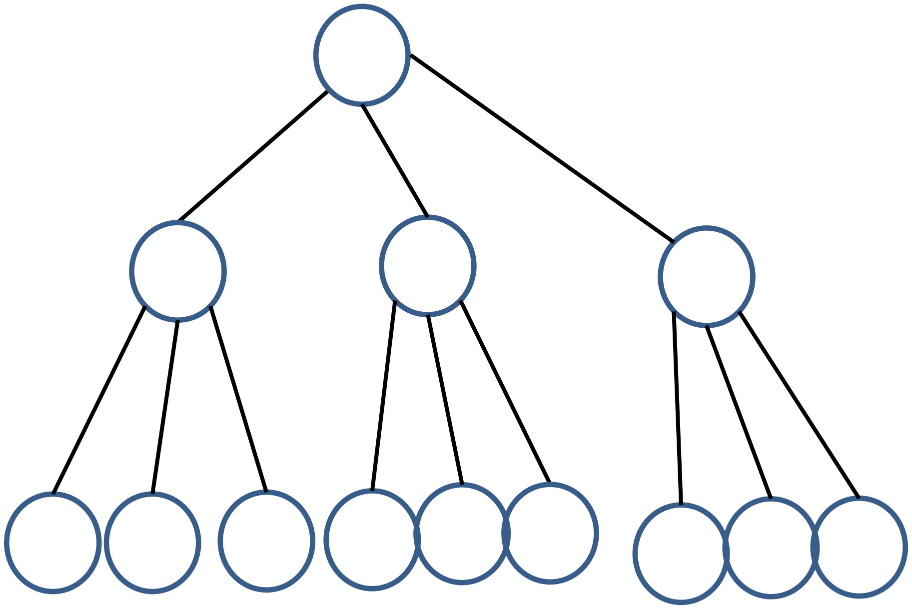
How we design a computer program to play a game ?

1. Analysis of the board + Strategy + Tactics -> Mixed up and somehow results a move
2. IF THEN Rules – If you can make a move then do it
3. Look ahead and evaluate – static function – linear scoring polynomial

$$S = g\left(f_1, f_2, \ldots, f_n\right) \text{ , where } f_i \text{ are features}$$
$$= c_1 f_1 + c_2 f_2 + \ldots + c_n f_n$$


2
3

Game tree



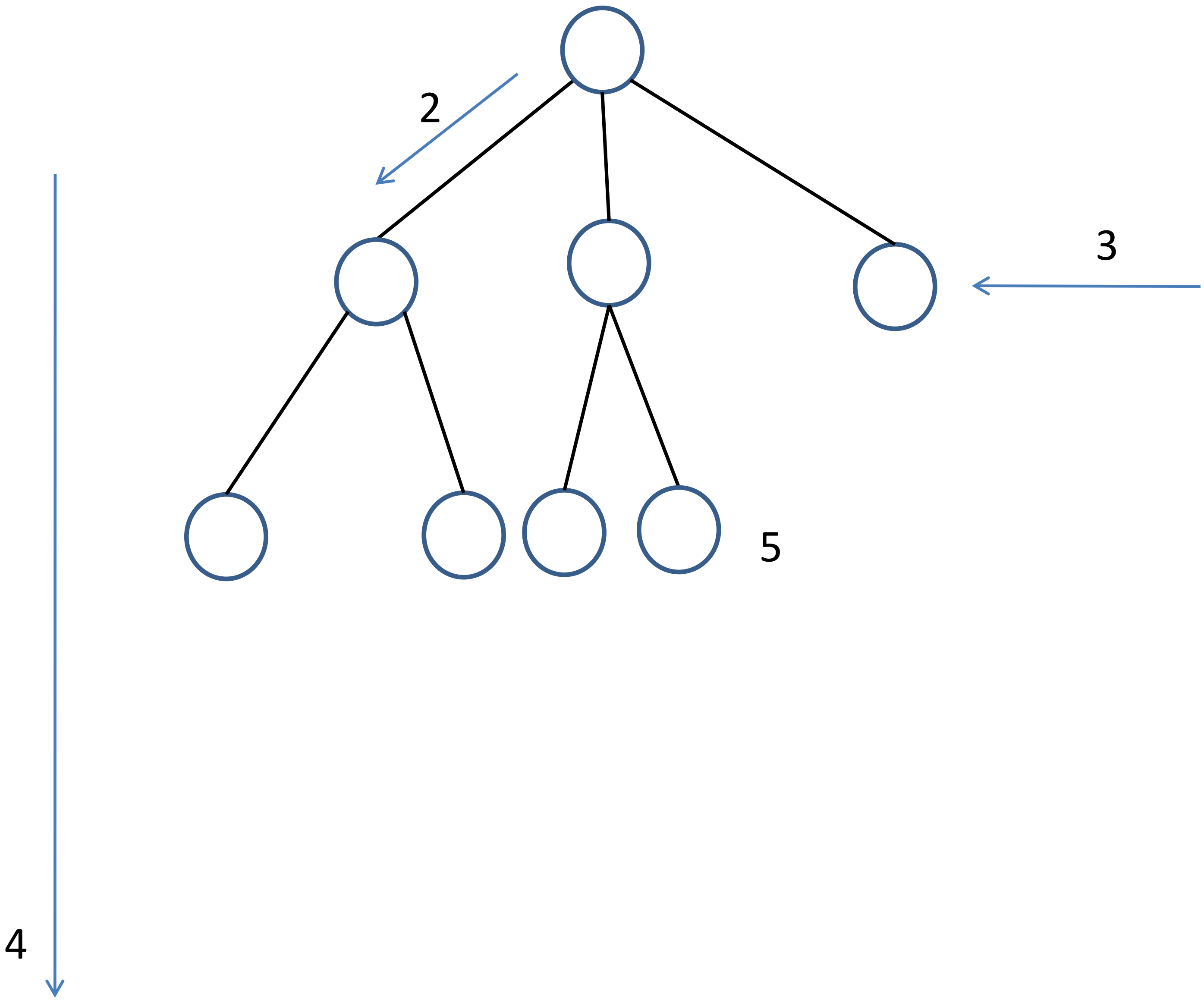
- Nodes – board configurations
- Branches – moves , transform one position into another
- *d* – depth of the tree (2)
- *b* – branching factor (3)
- b^d – terminal leaves (9)

4. Exhaustive search – Chess example

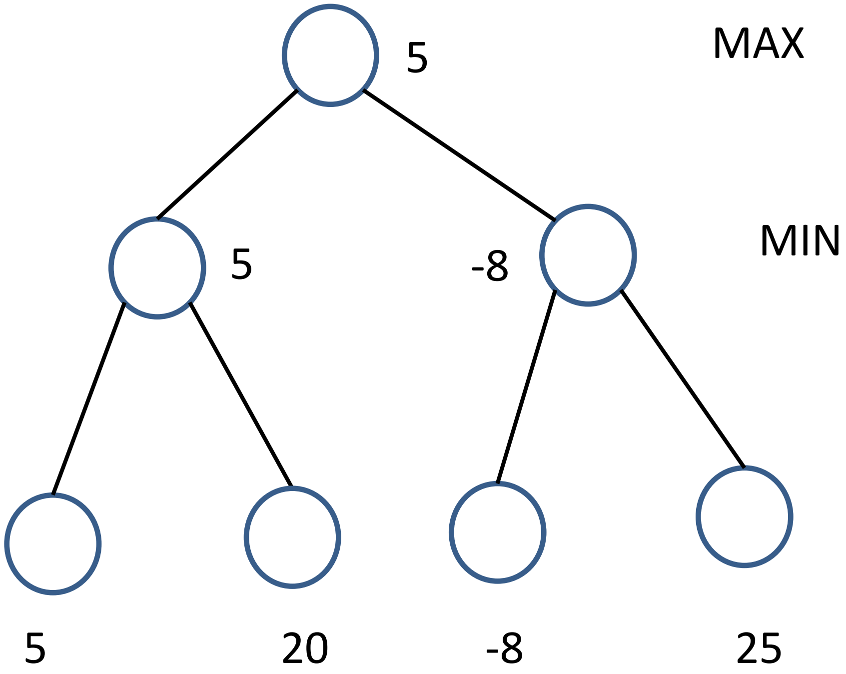
- Branching factor – 16
- Depth – 100
- Chess possibilities - 10^{120}
  - Universe contains - 10^{80} atoms
  - 1 year – 3* 10^7 seconds
  - 1 sec - 10^9 nano sec
  - Time from the beginning of the universe - 10^{10} years
  - Total – 10^{106}
- Analysis would be just getting started

5. Look ahead as far as possible

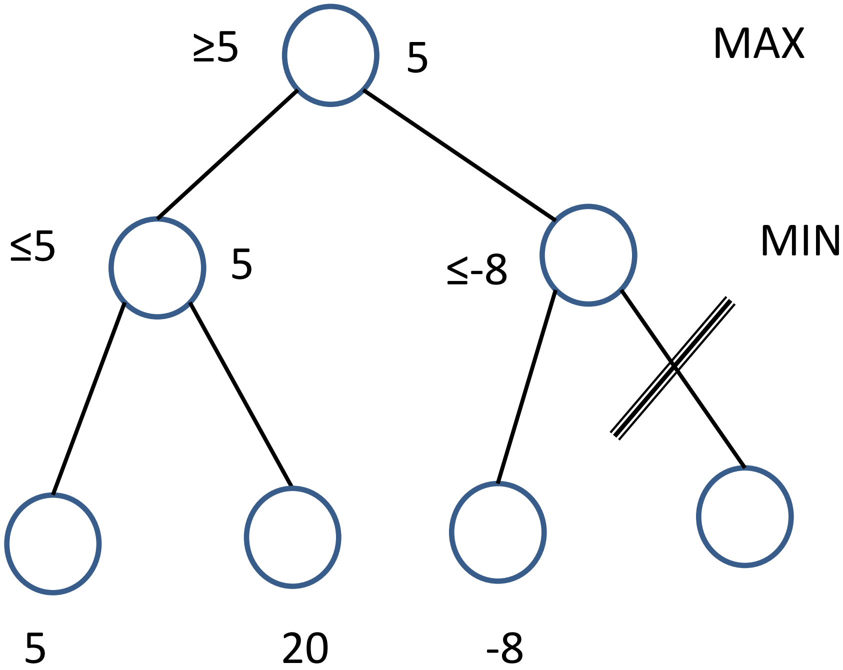
- Shannon & Turing
- Static evaluation – compute a number that reflects the board quality
  - Positive values favors one player (MAX player)
  - Negative values favors the other (MIN player)

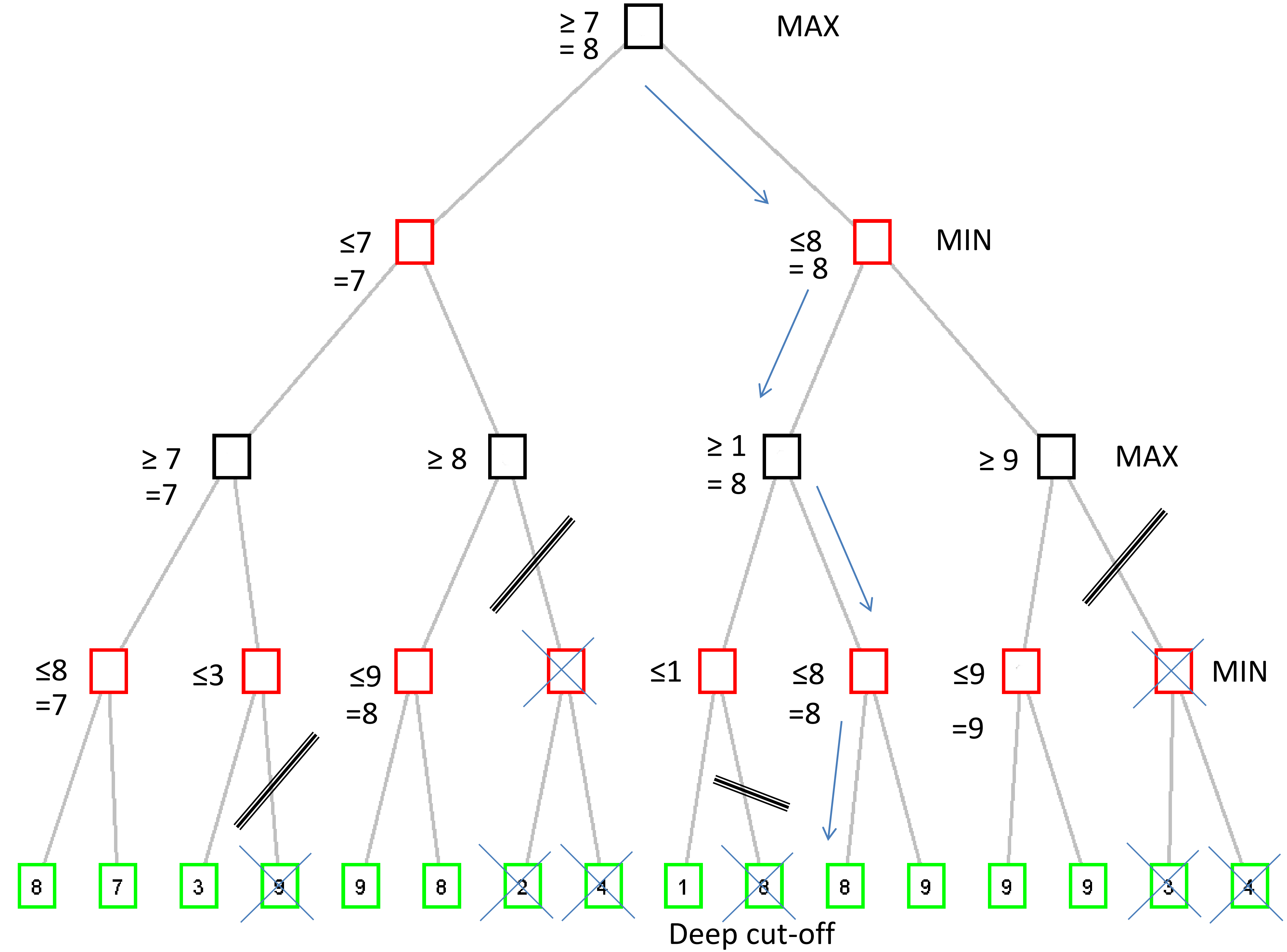

2
3
5
4

Minimax



Alpha Beta




≥ 7
= 8
MAX
≤7
=7
≤8
= 8
MIN
≥ 7
=7
≥ 8
≥ 1
= 8
≥ 9
MAX
≤8
=7
≤3
≤9
=8
≤1
≤8
=8
≤9
=9
MIN
8
7
3
9
9
8
2
4
1
8
8
9
9
9
3
4
Deep cut-off

ALPHA-BETA algorithm

- If the level is the top, let alpha be -∞ and let beta be ∞
- If the limit of search has been reached, compute the static value of the current position relative to the appropriate player. Report the result.
- If the level is a minimizing level
  - Until all children are examined with ALPHA-BETA or until alpha is equal to or grater than beta
    - Use ALPHA-BETA procedure, with the current alpha and beta values, on a child; note the value reported
    - Compare the value reported with the beta value; if the reported value is smaller, reset beta to the new value
  - Report beta
- Otherwise, the level is a maximizing level:
  - Until all children are examined with ALPHA-BETA or until alpha is equal to or grater than beta
    - Use ALPHA-BETA procedure, with the current alpha and beta values, on a child; note the value reported
    - Compare the value reported with the alpha value; if the reported value is larger, reset alpha to the new value
  - Report alpha

Best-case and worst-case

- With ALPHA-BETA and for optimal arrangement, the number of static evaluations has the following formulae:

$$s = 2 * b^{\frac{d}{2}} - 1$$

- Worst –case -> no cuttings

$$s = b^d$$

Progressive Deepening

- The branching factor is not always the same -> How deep can I go giving a certain amount of time?
- The number of nodes requiring static evaluation at the bottom of the tree is:

$$s = b^d$$

- The number of nodes in the rest of the tree is:

$$1 + b + b^2 + b^3 + \cdots + b^{d-1} = \frac{b^d - 1}{b - 1}$$

- The ratio of the number of nodes in the bottom level to the number of nodes up to the bottom level is:

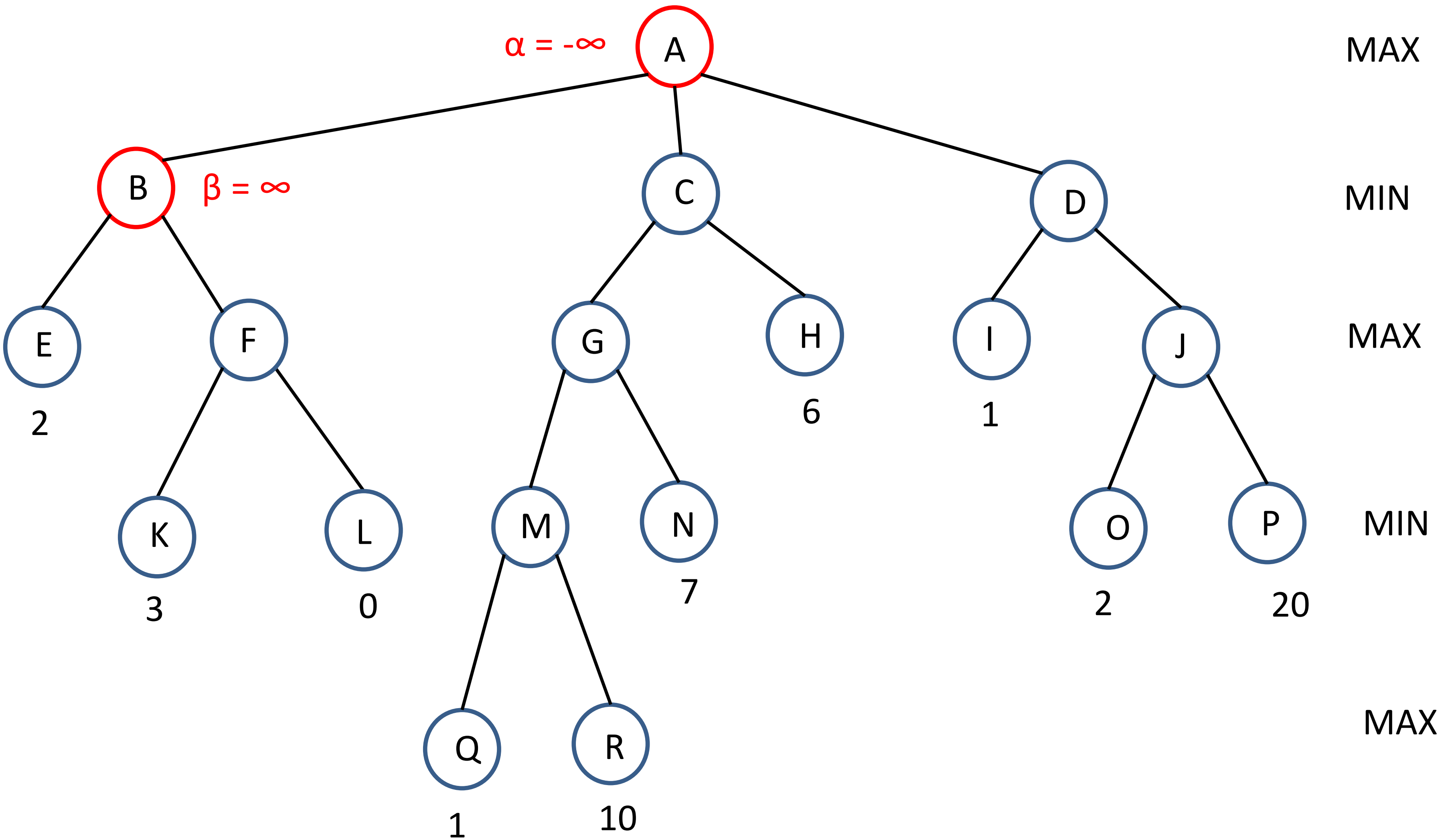
$$s = 1 + b + b^2 + b^3 + \cdots + b^{d-1} = \frac{b^d - 1}{b - 1} \qquad \text{ratio} = \frac{b^d}{\frac{b^d - 1}{b - 1}} \approx b - 1$$

- Improve the performance of ALPHA-BETA – reorder the nodes

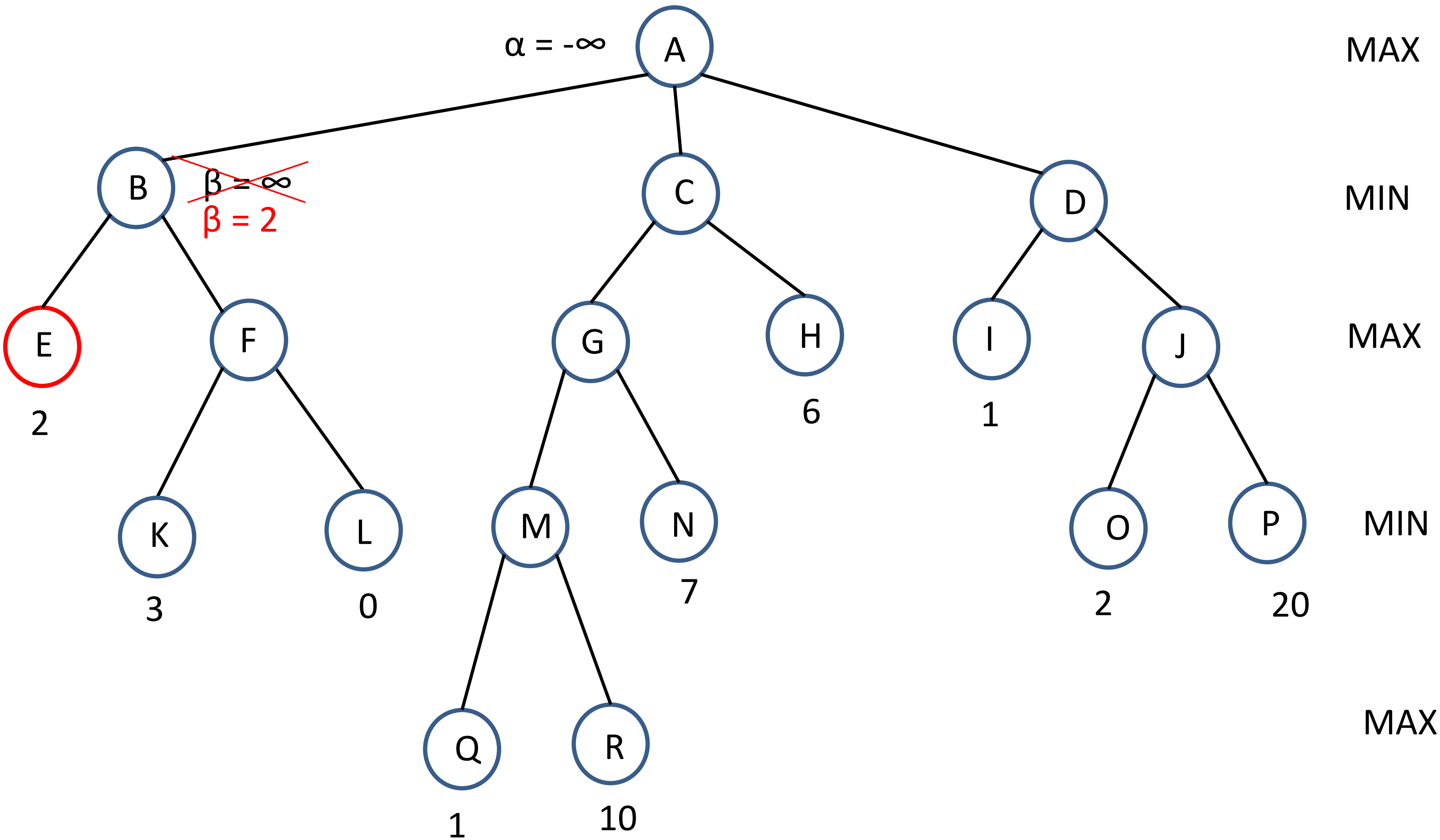
Deep-Blue

- Minimax – 14-15 levels
- Alpha Beta
- Progressive Deepening
- Parallel computing
- Opening book
- End game special situations
- Uneven tree development

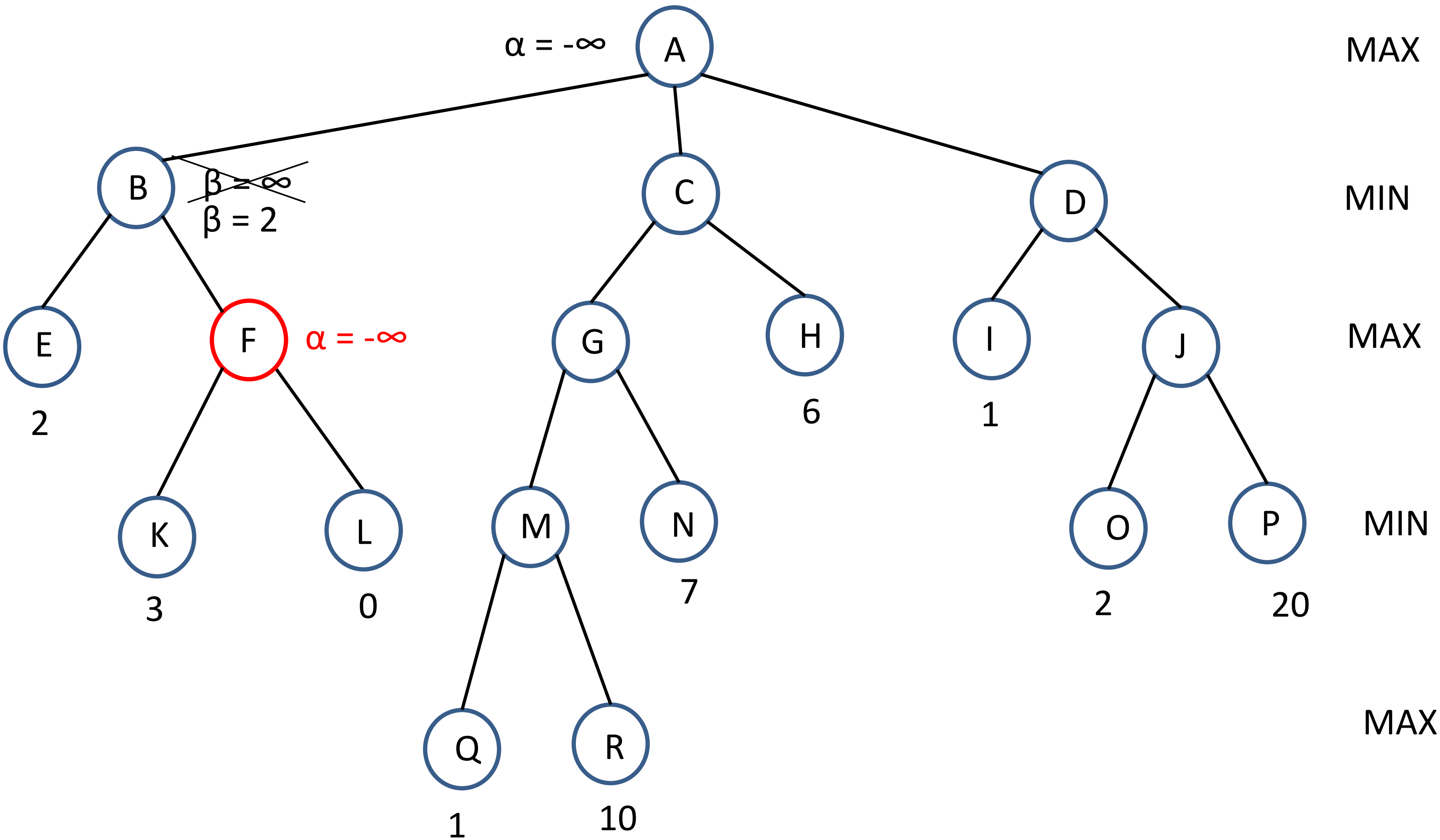
Alpha Beta example



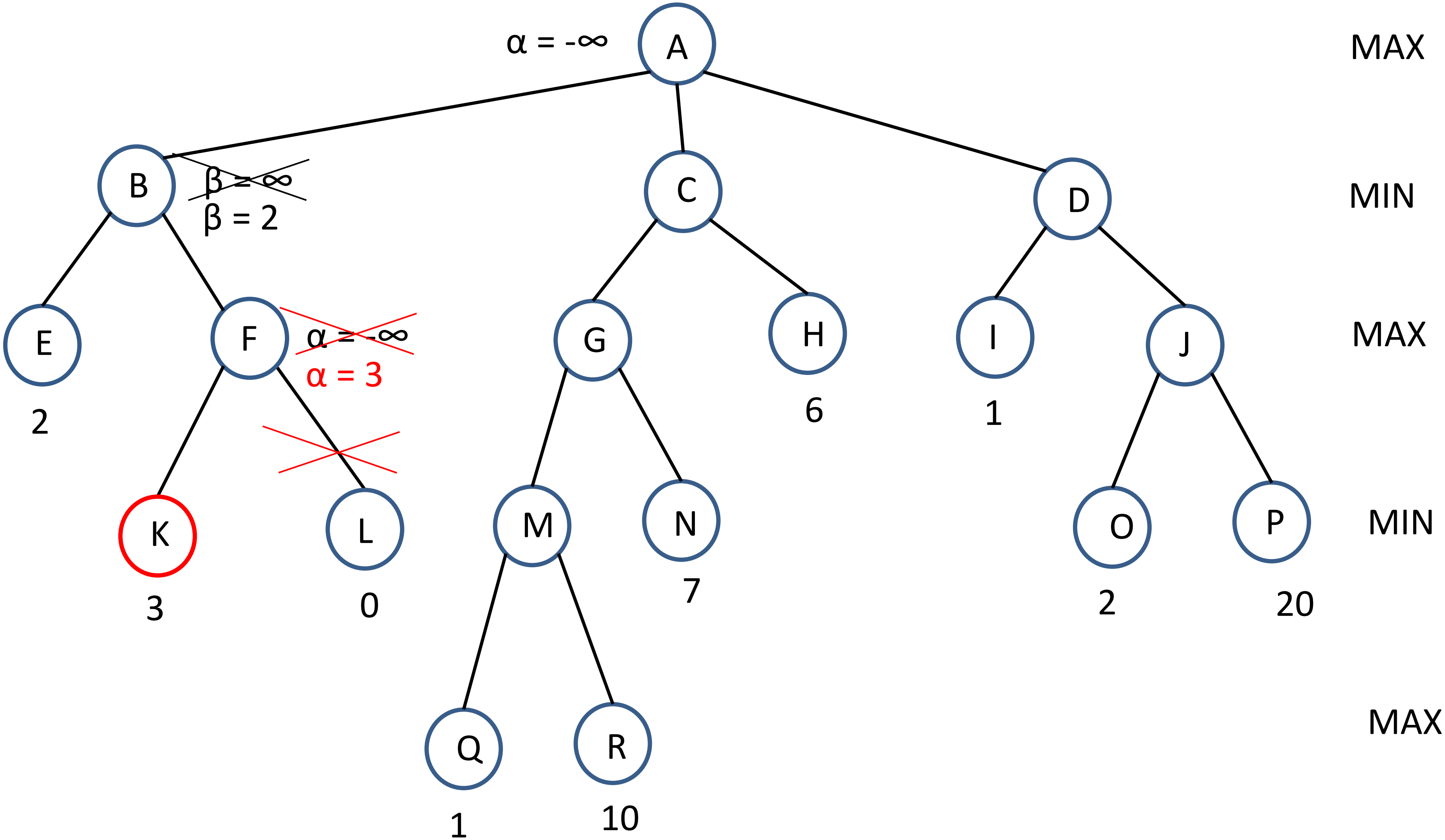
Alpha Beta example



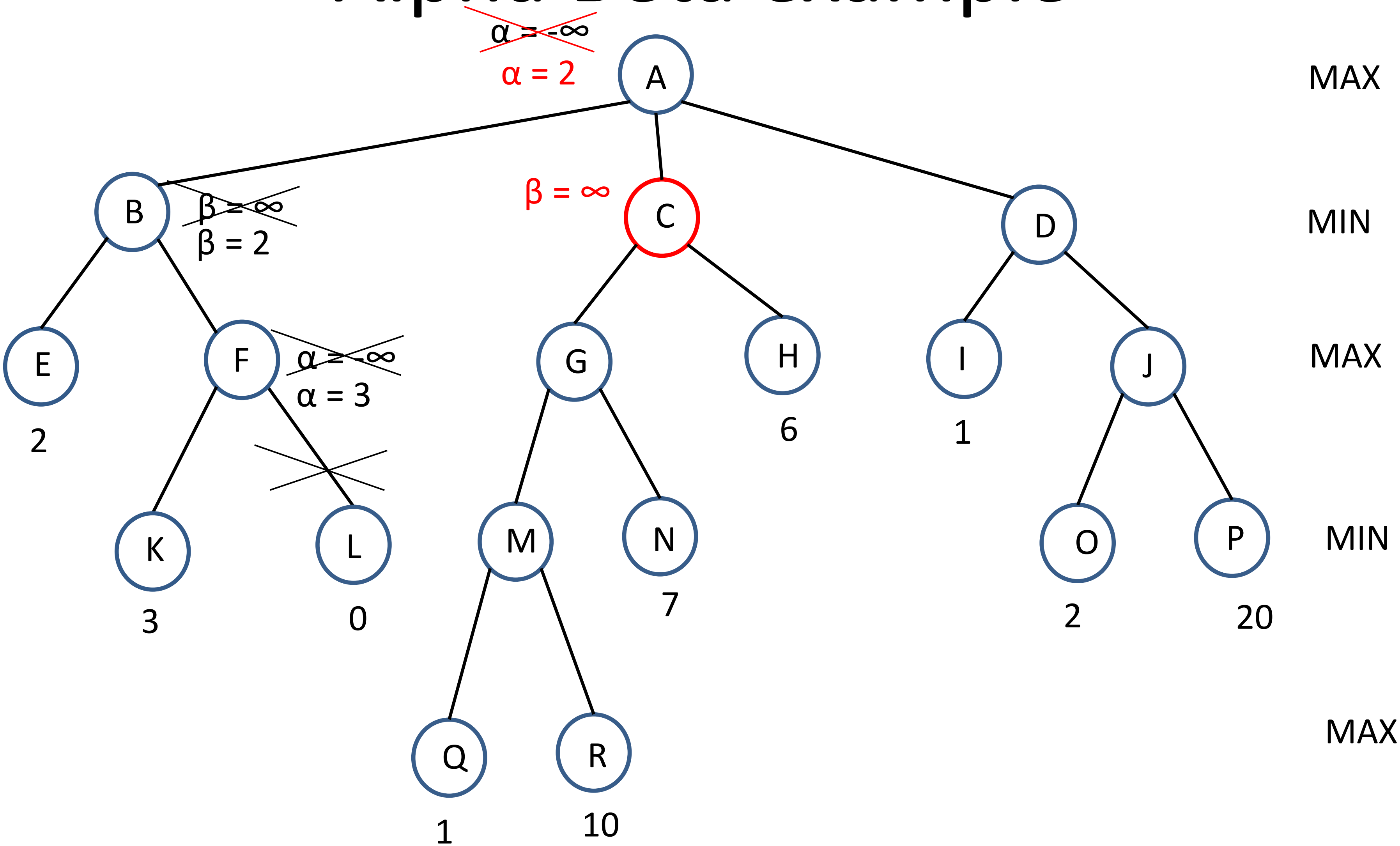
Alpha Beta example



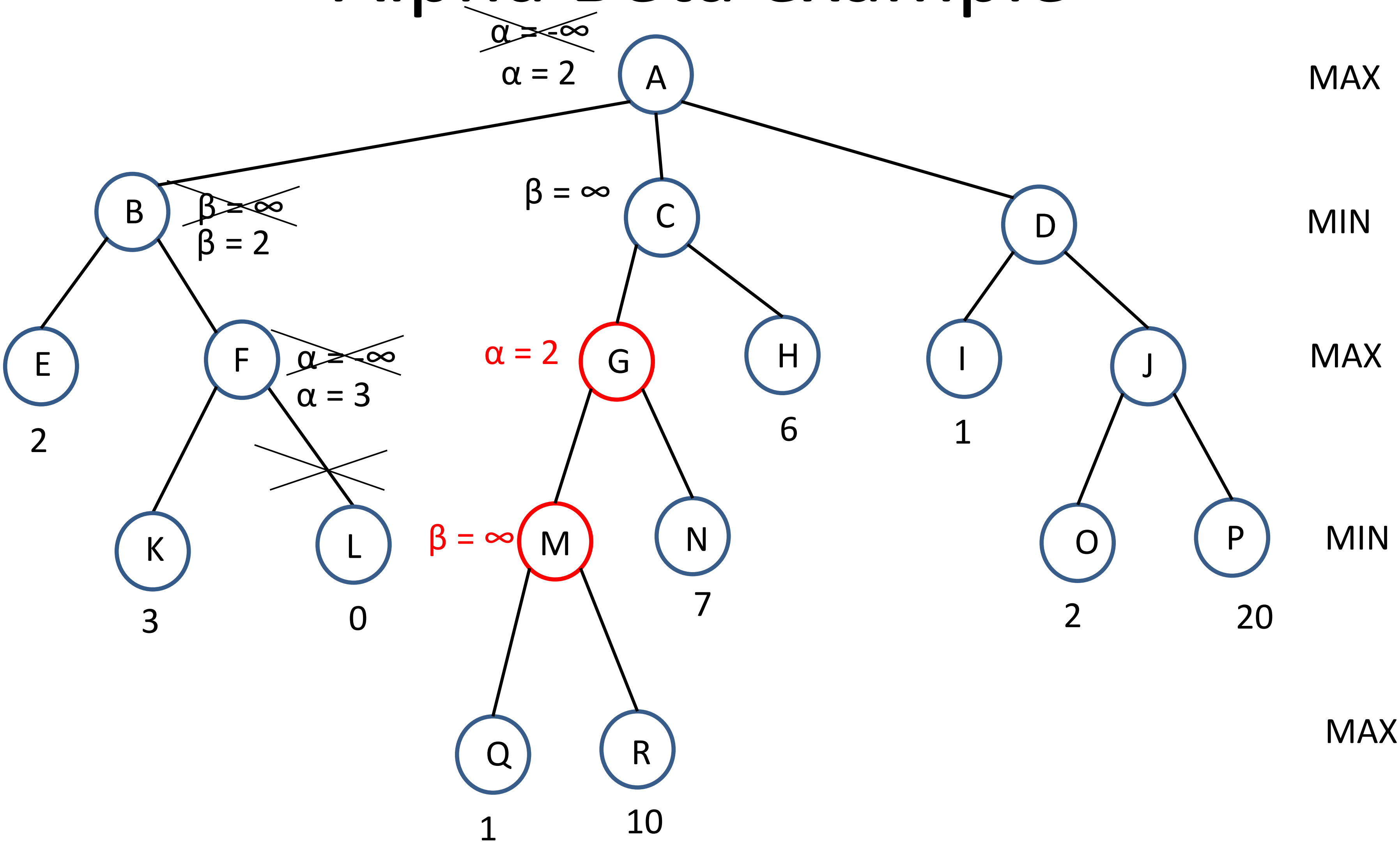
Alpha Beta example



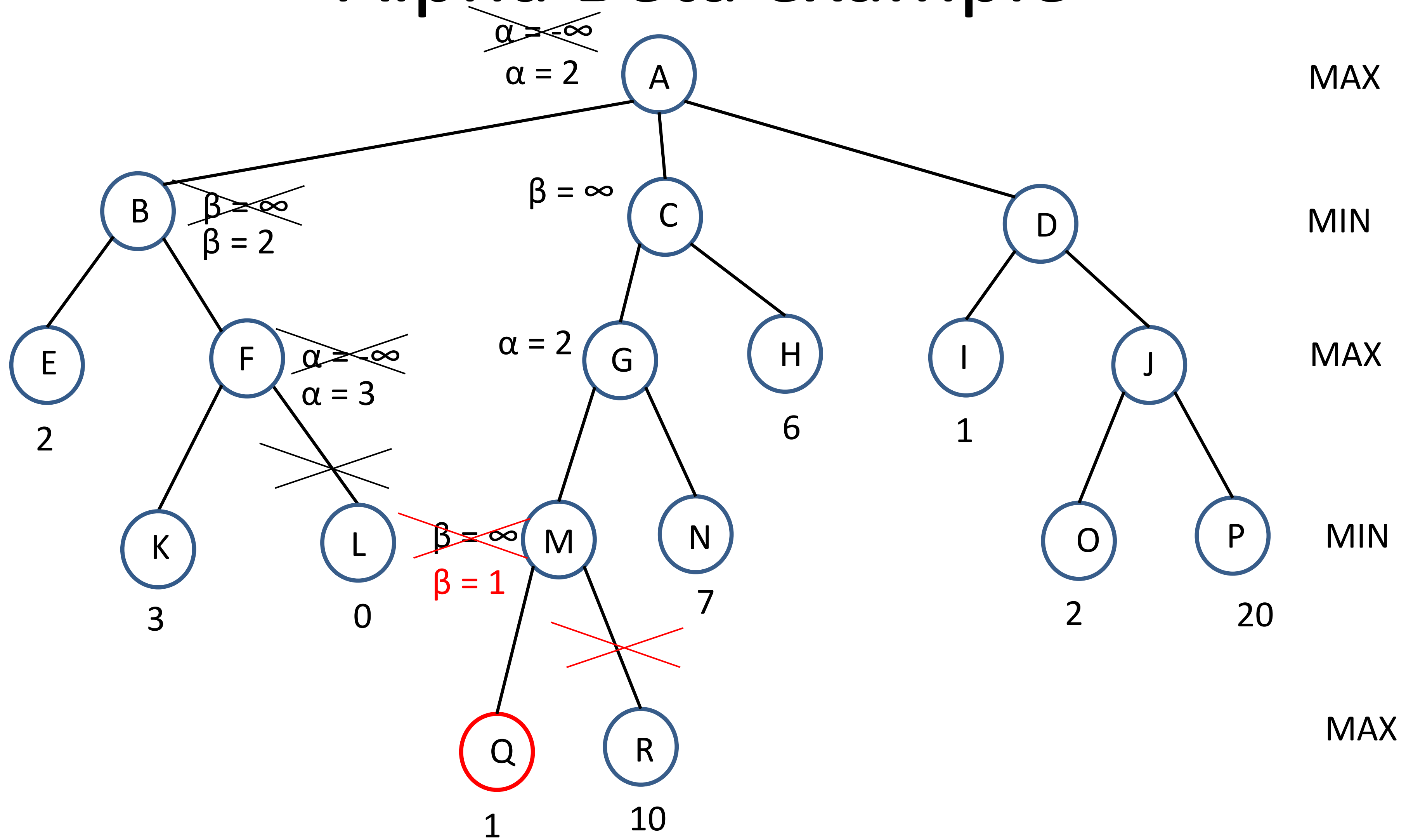
Alpha Beta example



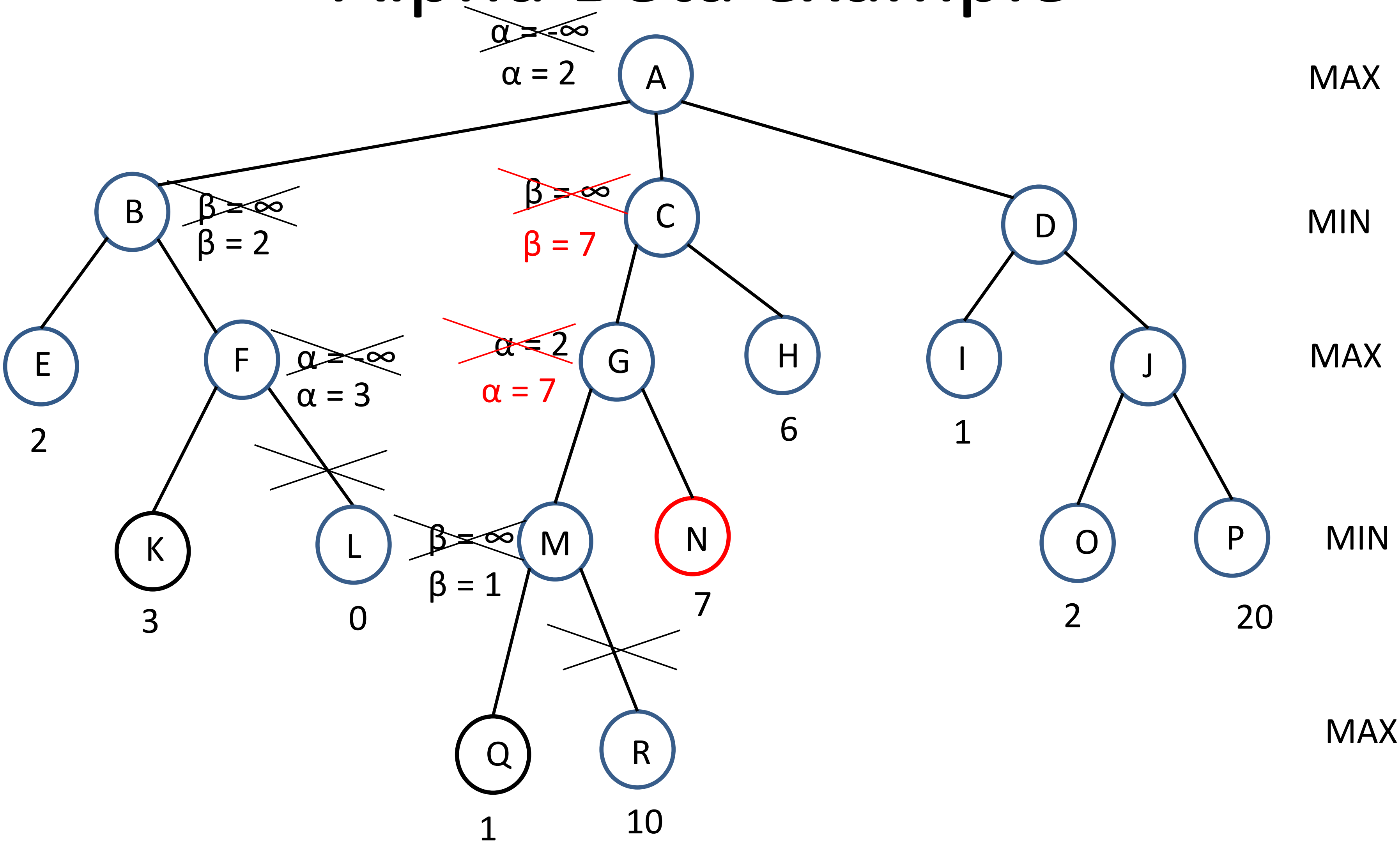
Alpha Beta example



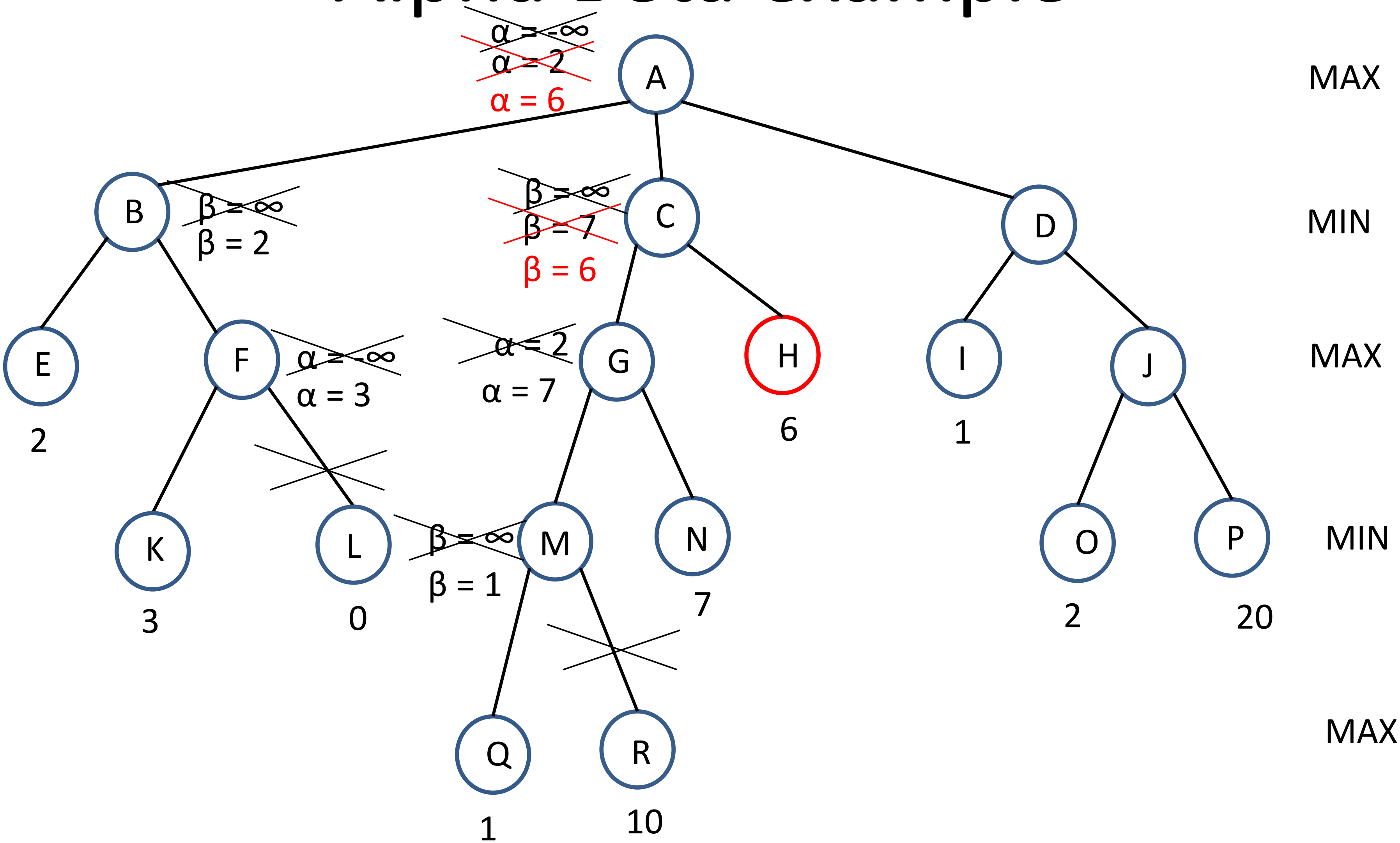
Alpha Beta example



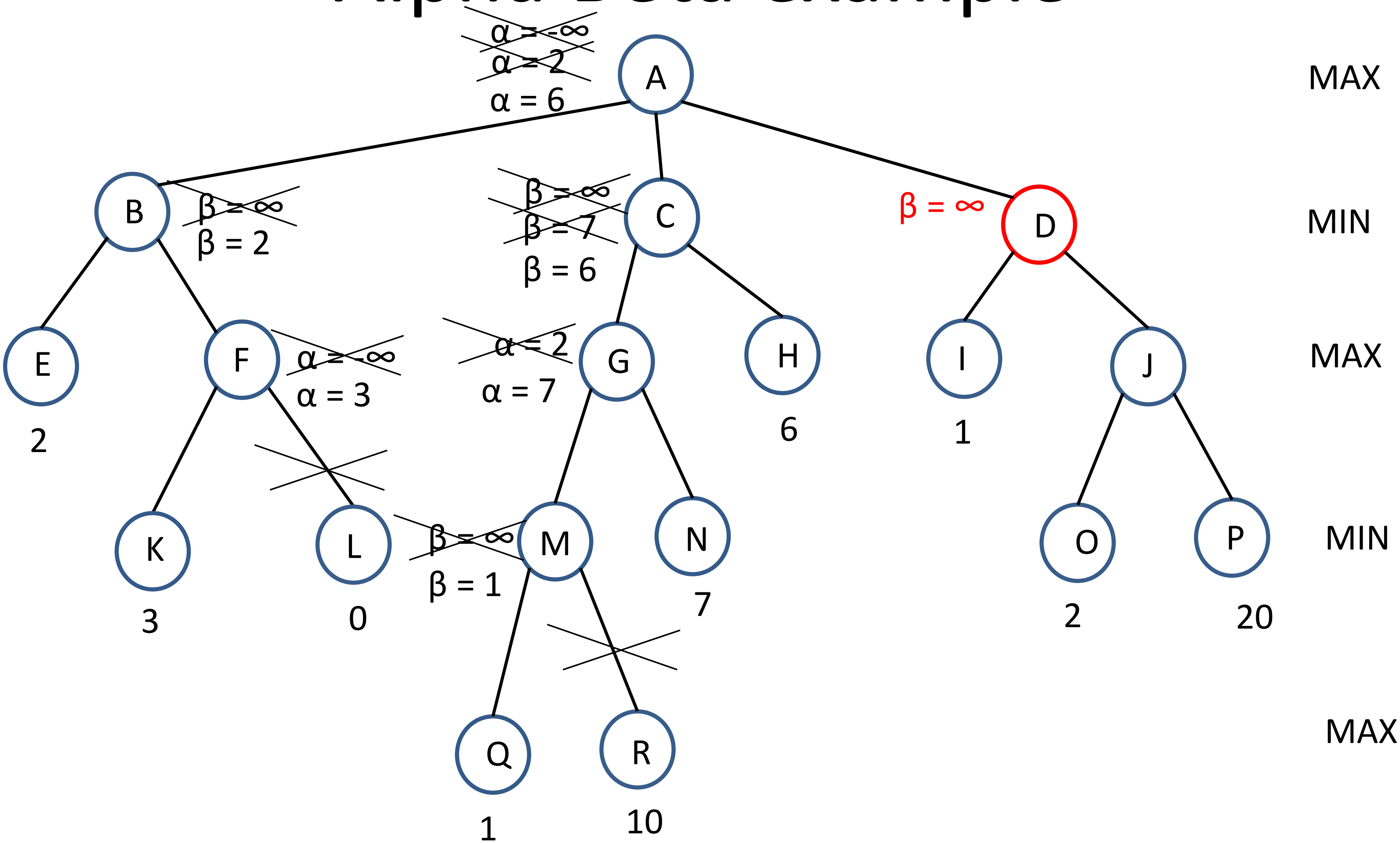
Alpha Beta example



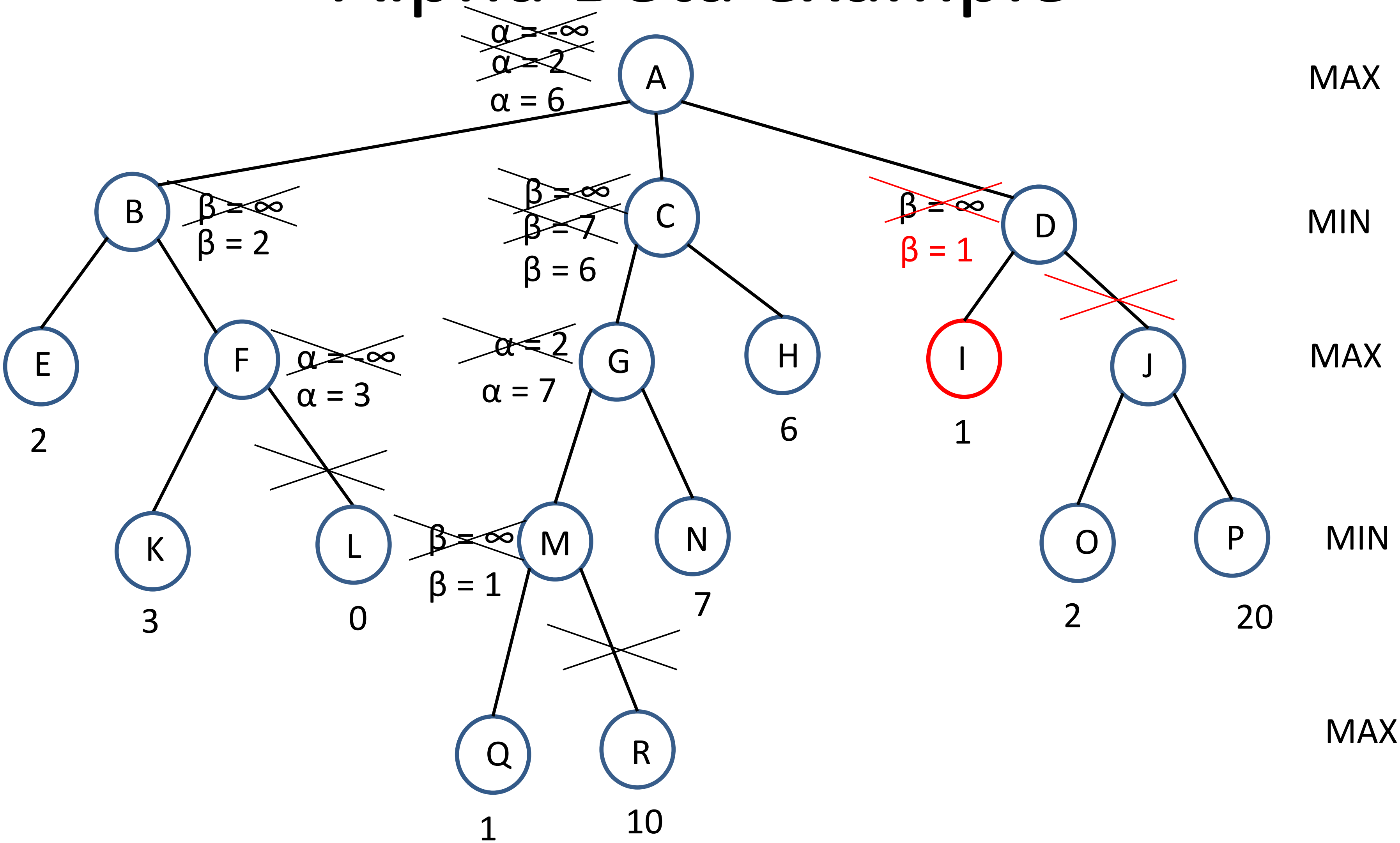
Alpha Beta example



Alpha Beta example



Alpha Beta example



List the leaf nodes in the order that they are statically evaluated: E, K, Q, N, H and I

Related resources

- http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/exams/MIT6_034F10_quiz2_2007.pdf

Readings

- Artificial Intelligence (3rd Edition), Patrick Winston, Chapter 6