CSC242: Intro to AI

Lecture 5

Adversarial Search

Help Sessions

- Grad TA Xiaowan Dong
  - 5:00-6:00 pm Tuesday & Wednesday, CSB 724
- Undergrad TAs Sean Esterkin and Dan Scarafoni
  - 2:30-3:30 pm Thursdays, CS Majors Lab CSB 633
  - 5:00-6:00 p.m. Mondays, CS Majors Lab CSB 633
- Instructor Henry Kautz
  - 12:00-1:00 pm Friday, CSB 709

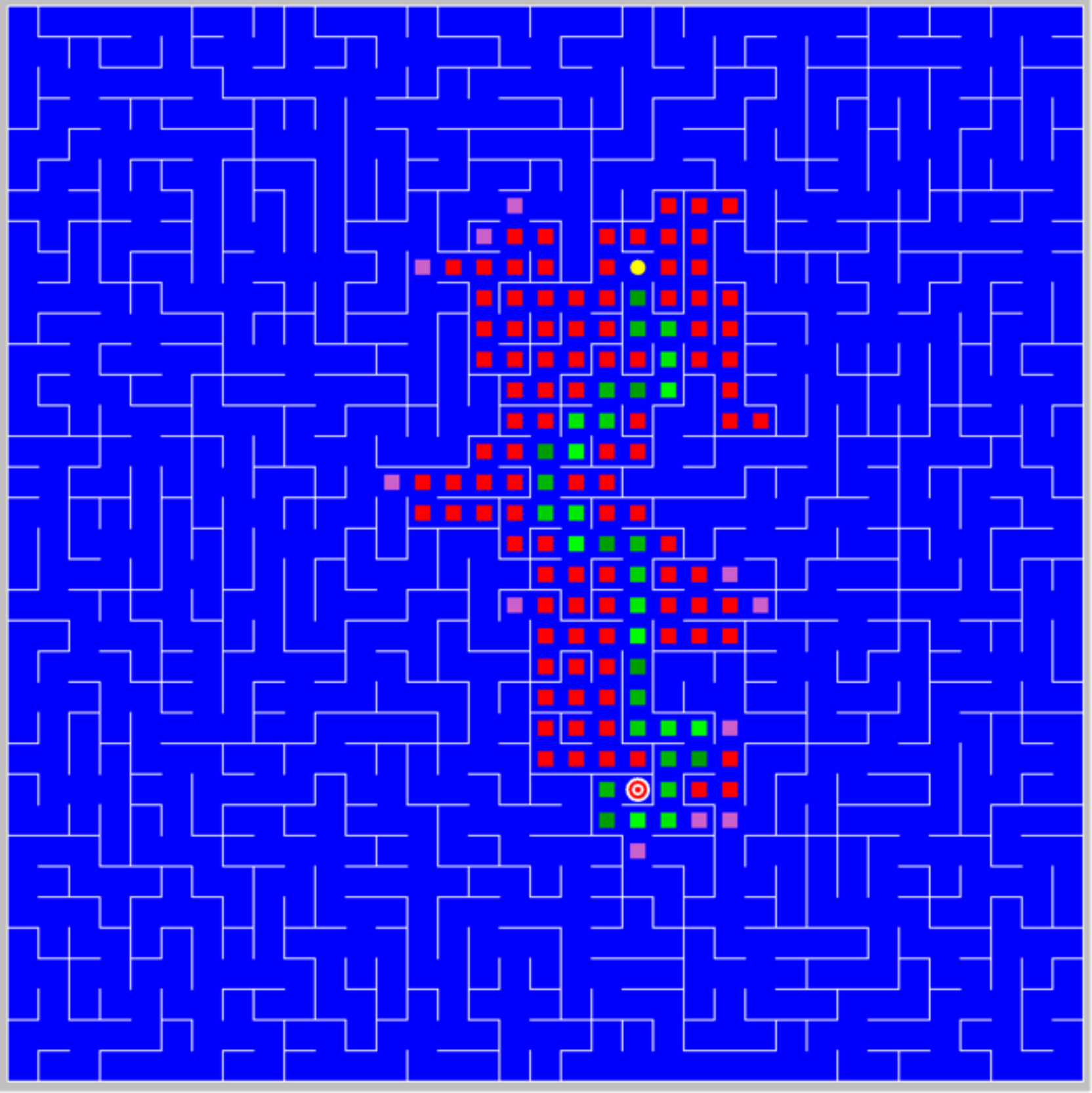

BACK
TO
THE PAST

www.cs.rochester.edu/u/kautz/MazesOriginal/search/applet.html

www.cs.rochester.edu/u/kautz/MazesOriginal/search/applet.html

Reader

Gmail Docs Search Demo Henry Kautz UR Calendar Cascade Server UR Logins Reviewing Finanacial



Create

Width: 35

Height: 35

Cycles: 0

Clear

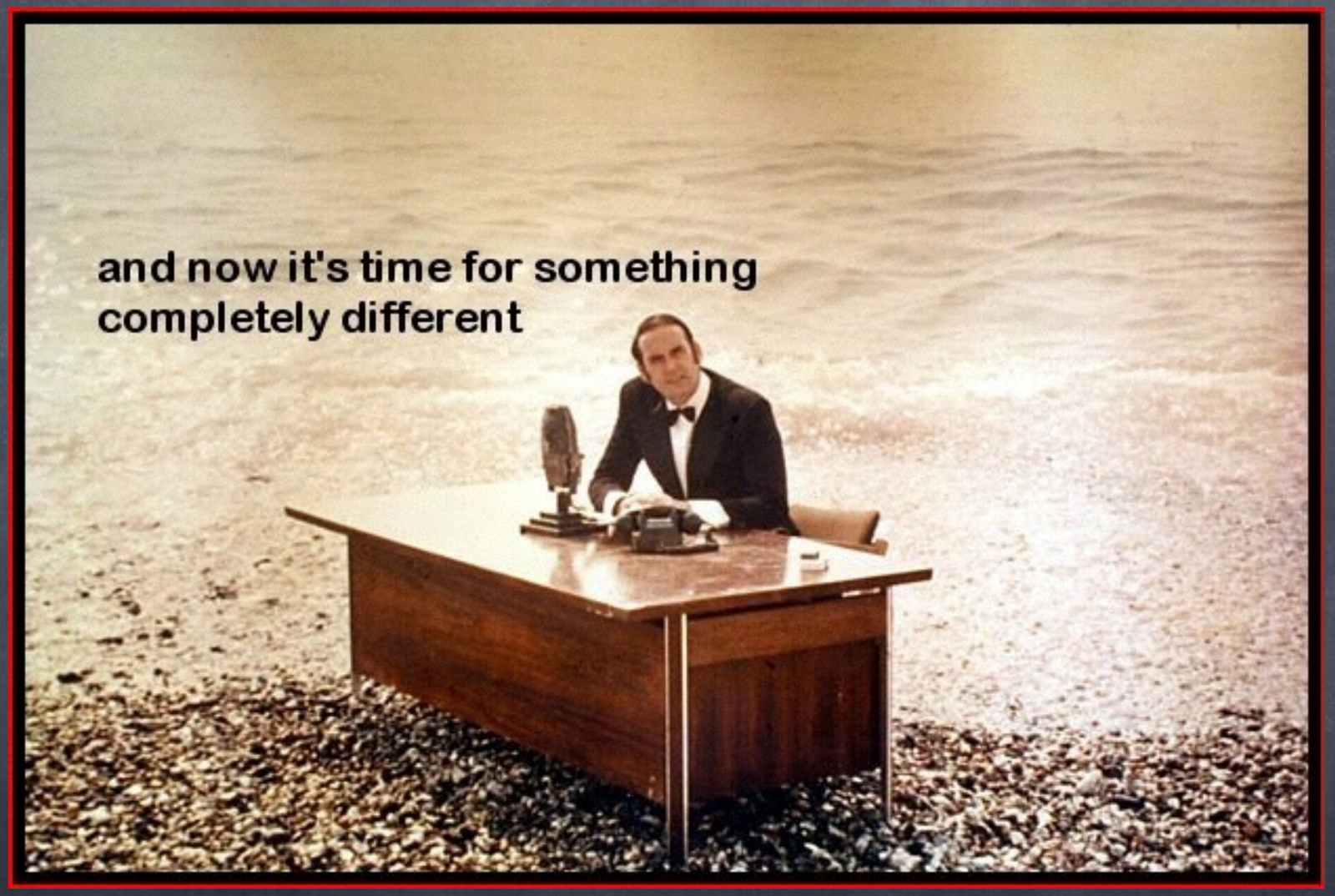
Search:

- DFS
- BFS
- IDS
- Best
- A*

Solution

Length: 33

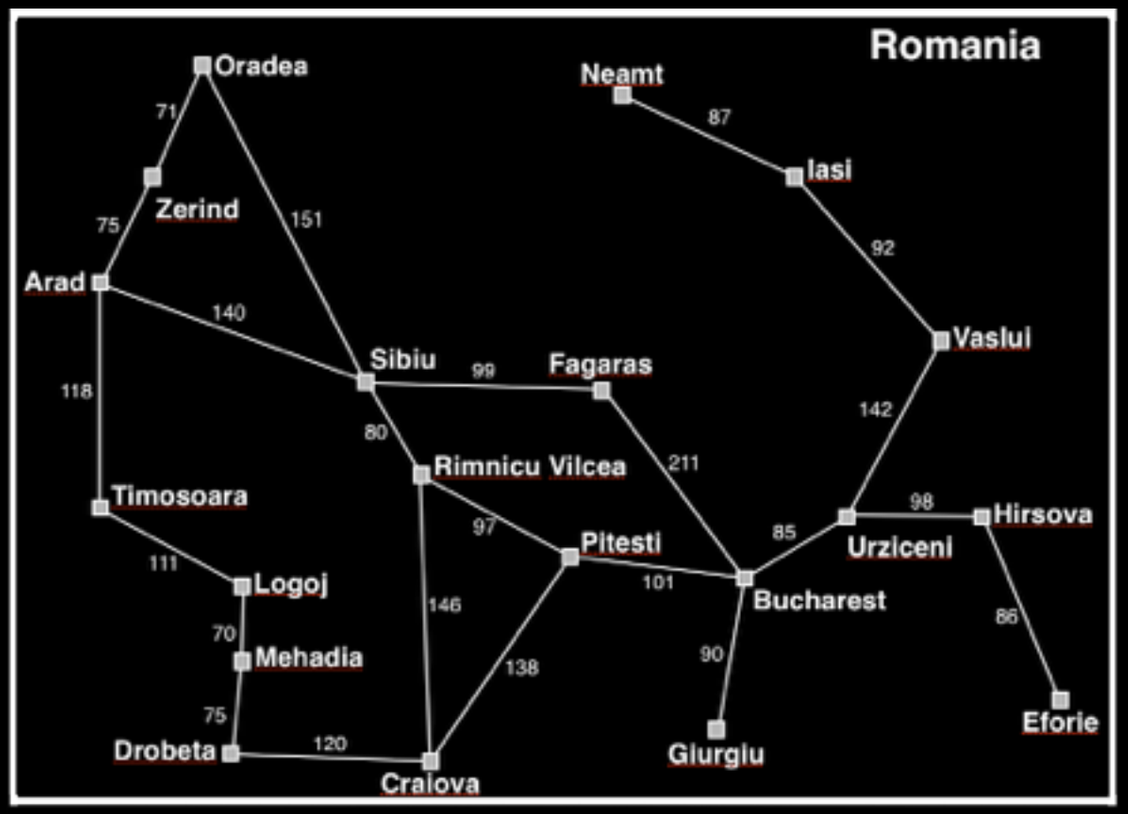
Nodes: 133


and now it's time for something
completely different

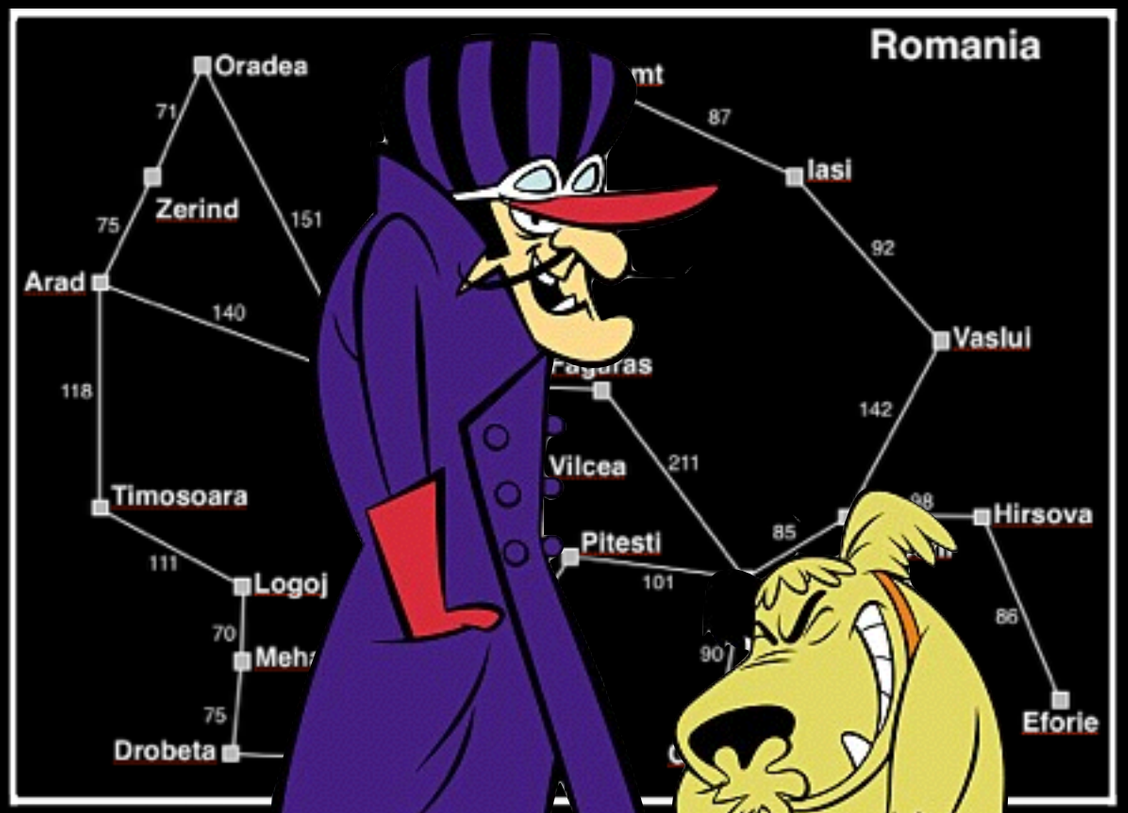
Adversarial Search

How are the games the same?
How are they different?

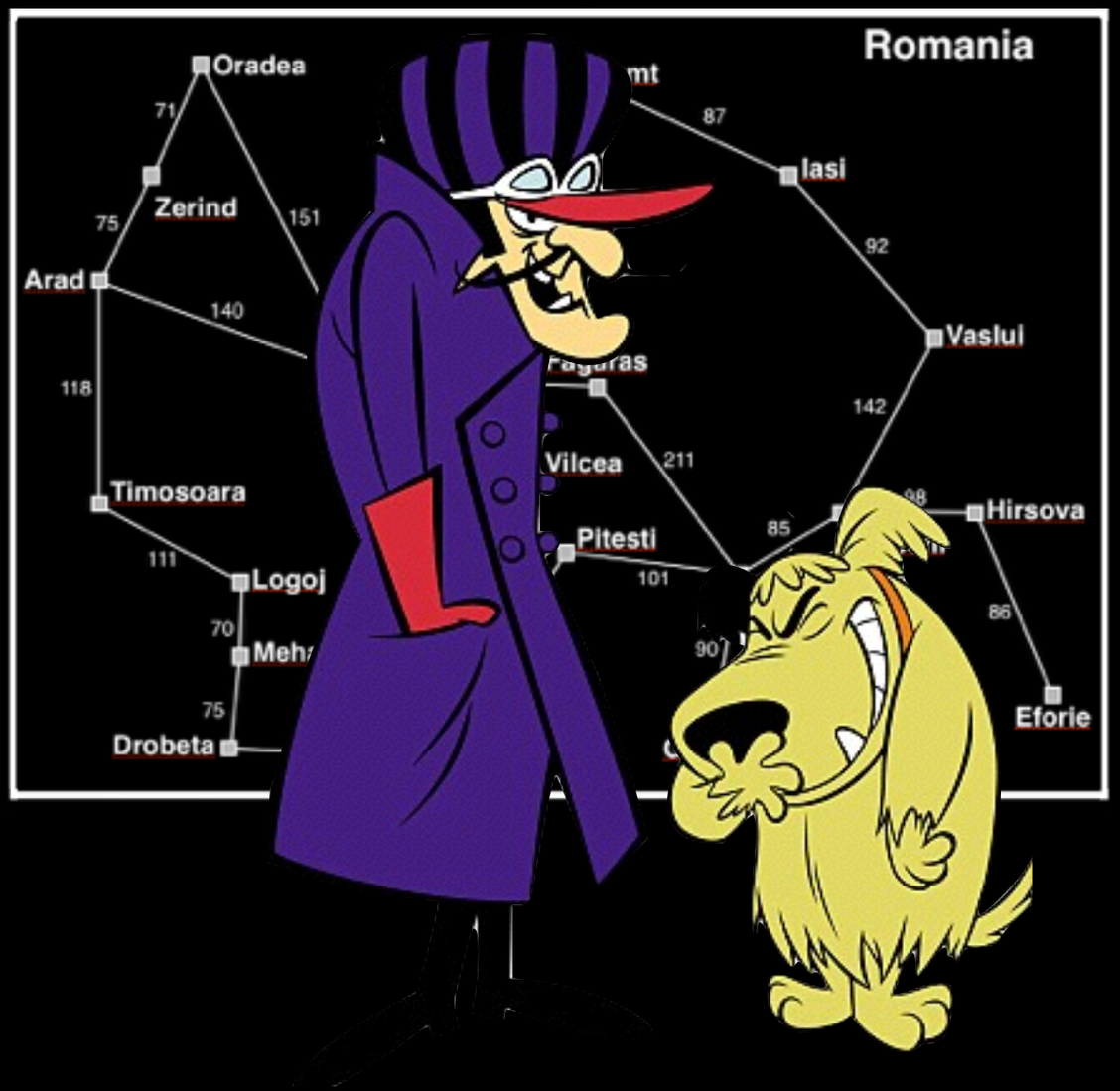
Adversarial Search


Romania
Oradea
Neamt
71
87
Iasi
Zerind
151
75
92
Arad
Vaslui
140
Sibiu
99
Fagaras
118
142
80
Rimnicu Vilcea
211
Timosoara
98
Hirsova
97
Pitesti
85
Urziceni
111
Logoj
101
Bucharest
146
86
70
Mehadia
138
90
75
Eforie
Drobeta
120
Giurgiu
Craiova

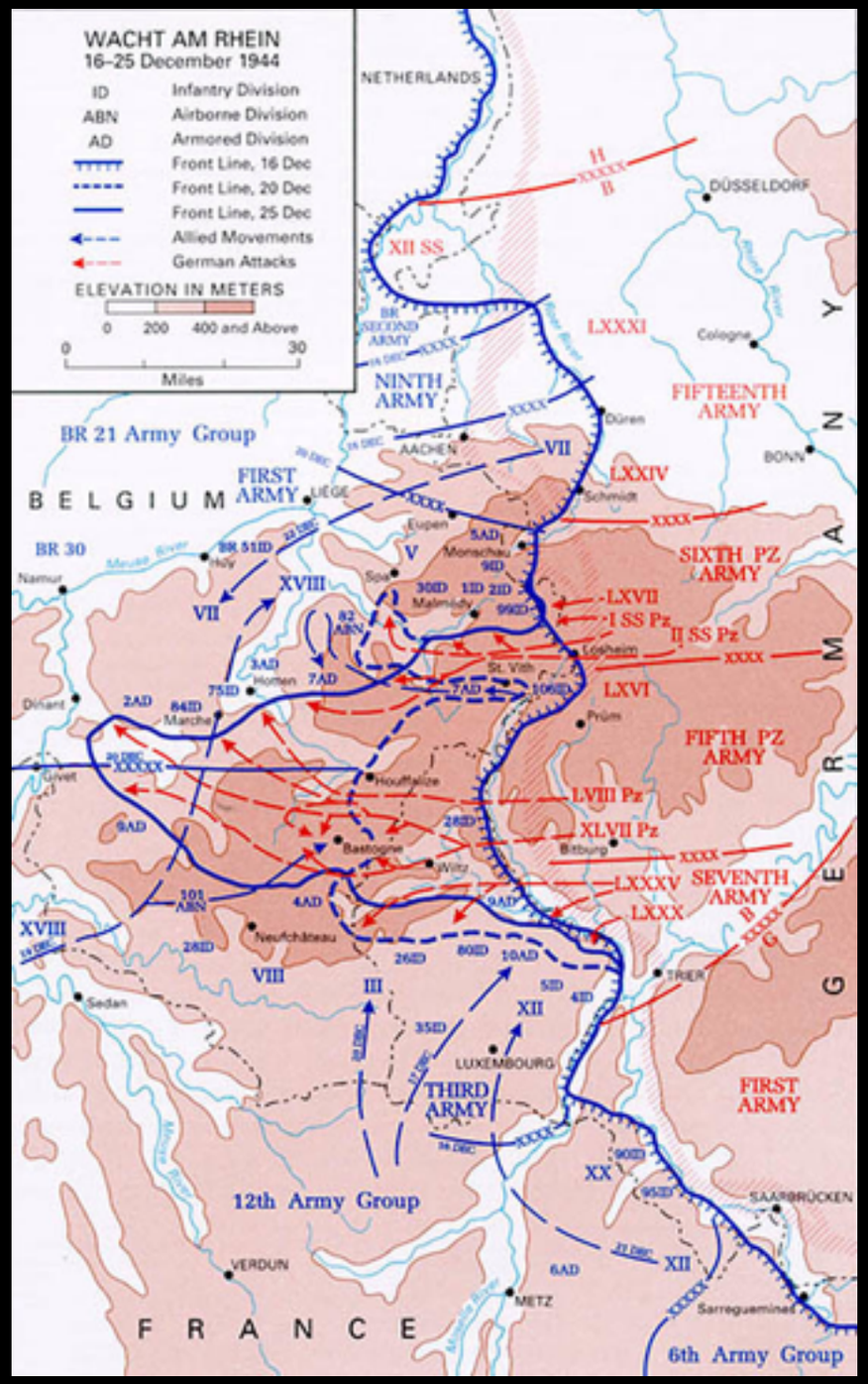

Thinkfun
Fifteen Puzzle
Fifteen Puzzle


Romania
Oradea
71
Zerind
75
151
Arad
140
118
Timosoara
111
Logoj
70
Meh
75
Drobeta
mt
87
Iasi
92
Vaslui
142
98
Hirsova
86
Eforie
Vilcea
211
Pitesti
101
85
90

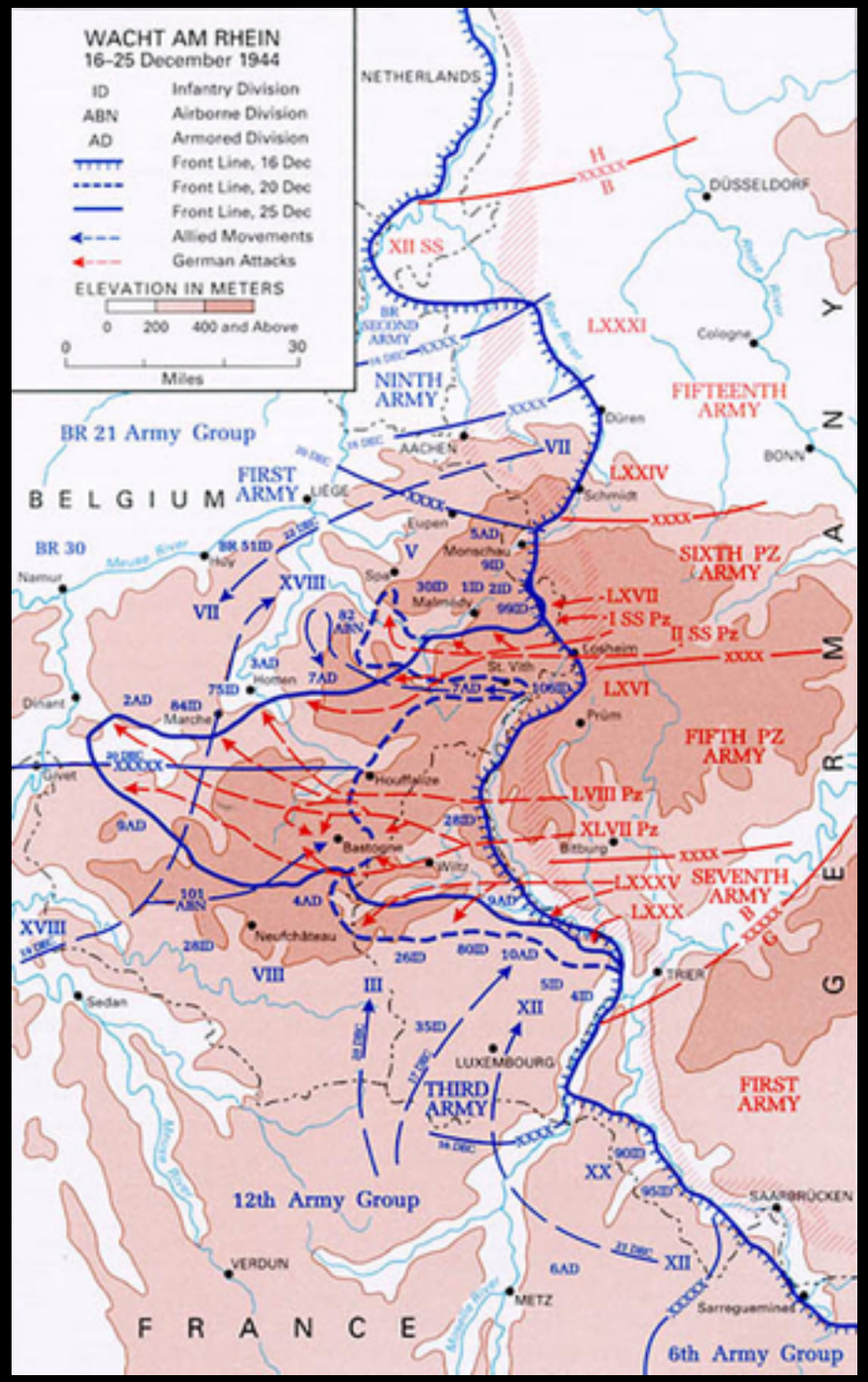

Thinkfun
Fifteen Puzzle
Fifteen Puzzle


Romania
Oradea
71
Zerind
75
151
Arad
140
118
Timosoara
111
Logoj
70
Meh
75
Drobeta
mt
87
Iasi
92
Vaslui
Fagaras
142
Vilcea
211
98
Hirsova
Pitesti
85
101
86
90
Eforie

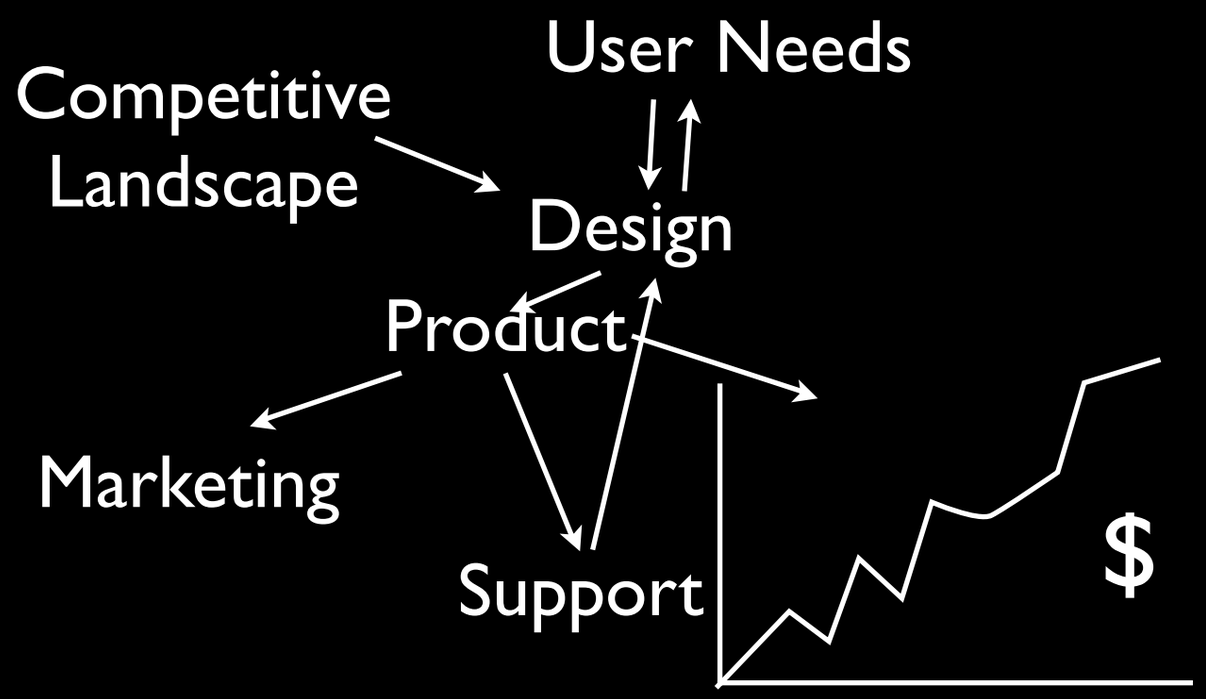

Thinkfun
Fifteen Puzzle
Fifteen Puzzle

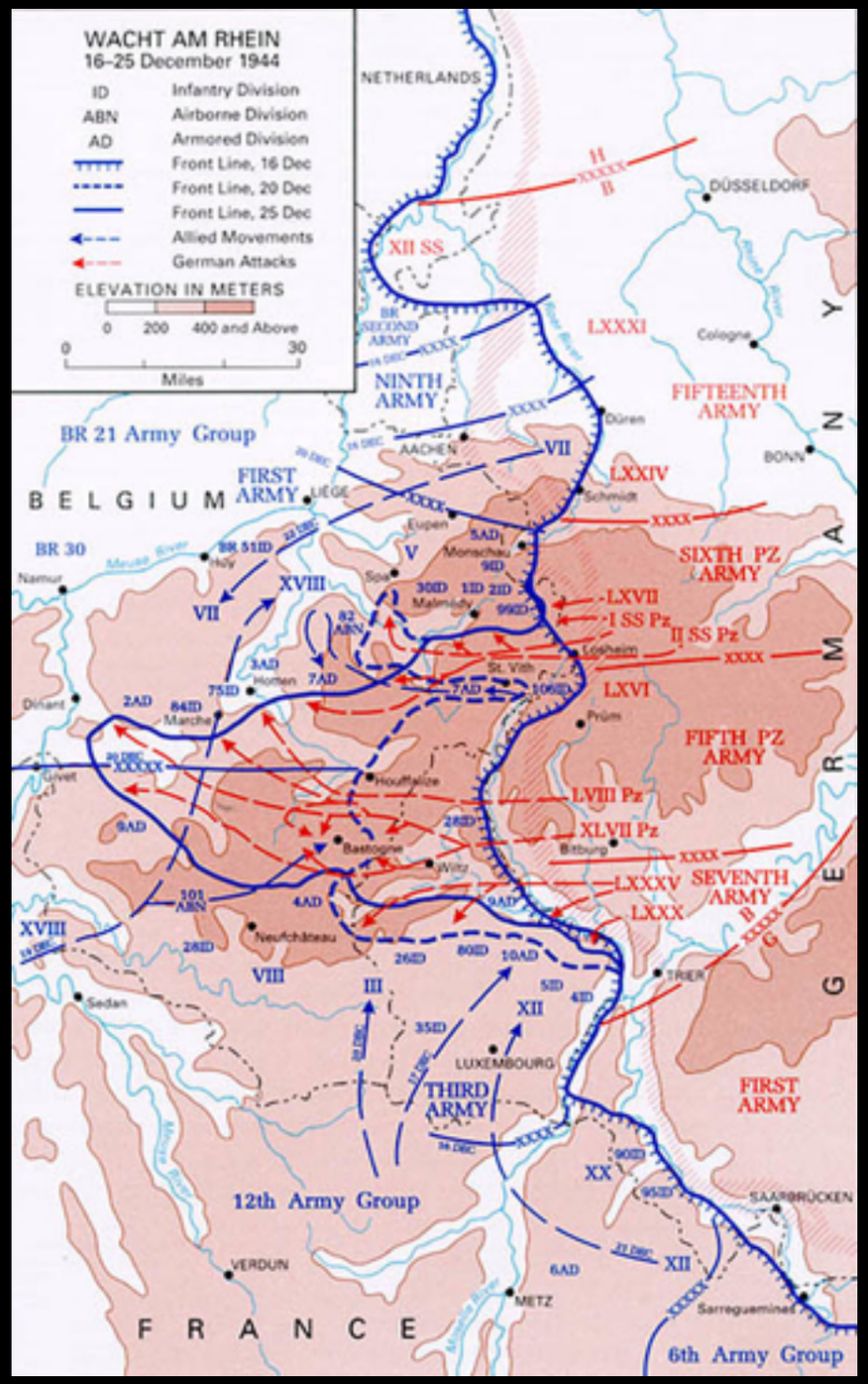


Battle of The Bulge
16-25 Dec 1944

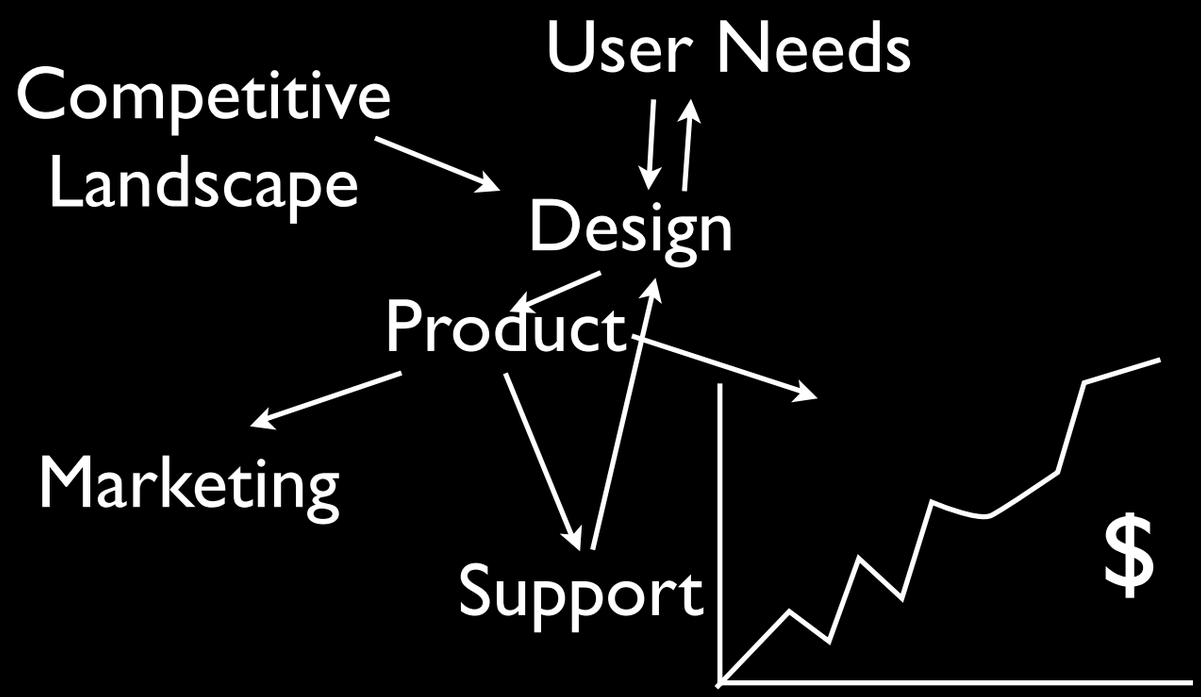


Battle of The Bulge
16-25 Dec 1944





Battle of The Bulge
16-25 Dec 1944



Multi-Agent Environments

- Unpredictability of other agents → contingencies (strategies)
- Agents goals in conflict → competition

Games

Games

Games

Games

- "Require the ability to make *some* decision even when calculating the *optimal* decision is infeasible"
- "Penalize inefficiency severely"

Games ≠ Toy Problems

$9! = 362880$

$35^{100} = 10^{154}$

2×10^{170}

Abstract Games

Abstract Games

- Deterministic (no chance)
- Nondeterministic (dice, cards, etc.)

Abstract Games

- Perfect information (fully observable)
- Imperfect information (partially observable)

Abstract Games

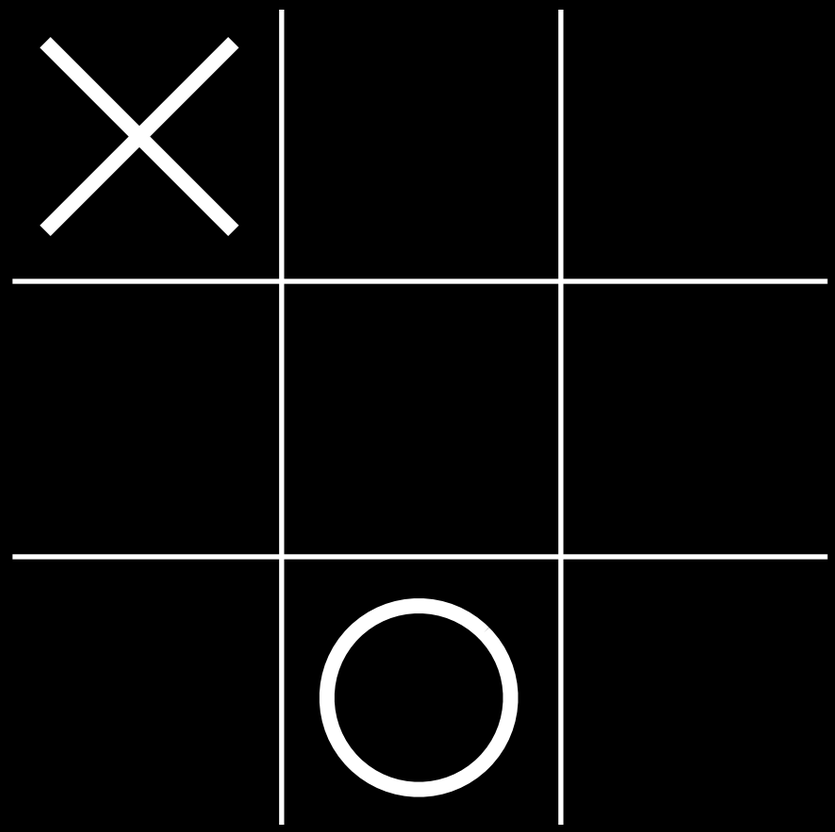
- Zero-sum (total payoff the same in any game)
  - What's good for me is bad for you and vice-versa
- Arbitrary utility function

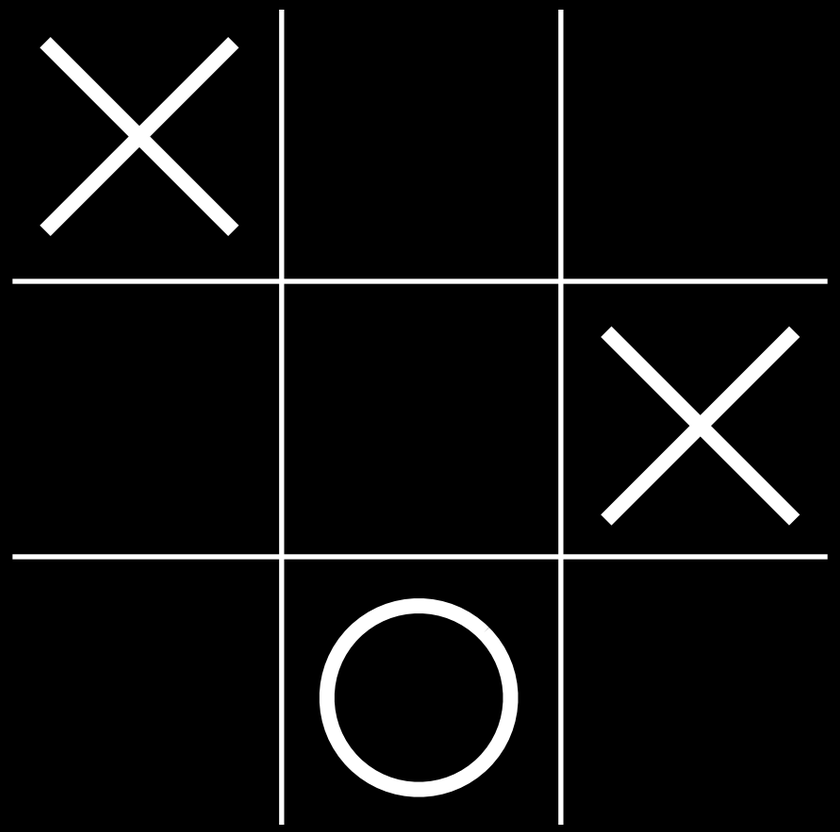
Types of Games

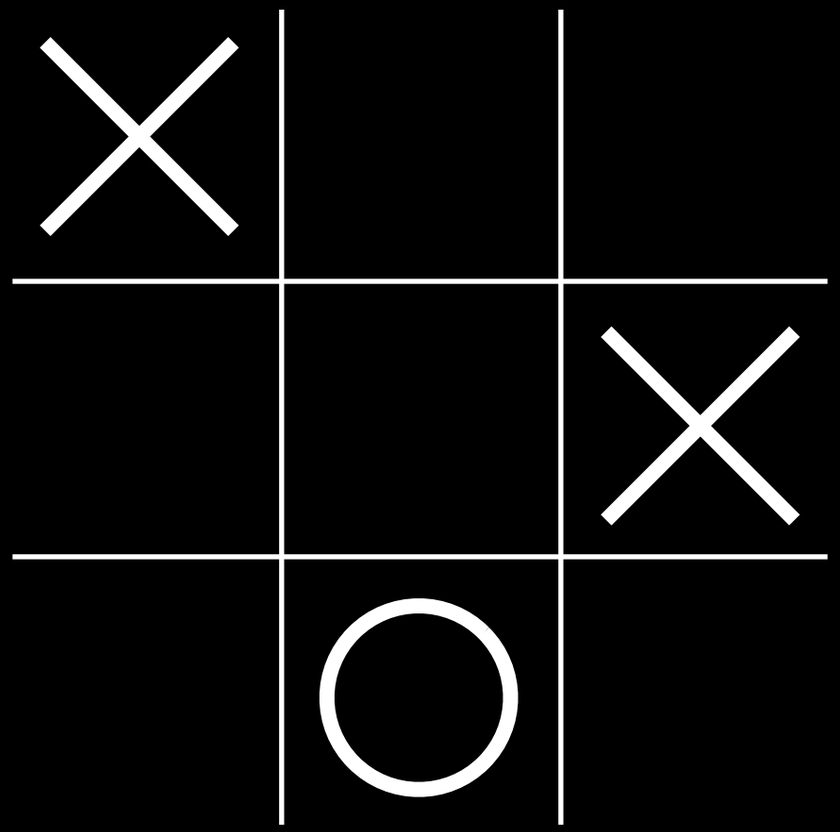
Deterministic (no chance)	Nondeterministic (dice, cards, etc.)
Perfect information (fully observable)	Imperfect information (partially observable)
Zero-sum (total payoff the same in any game)	Arbitrary utility functions

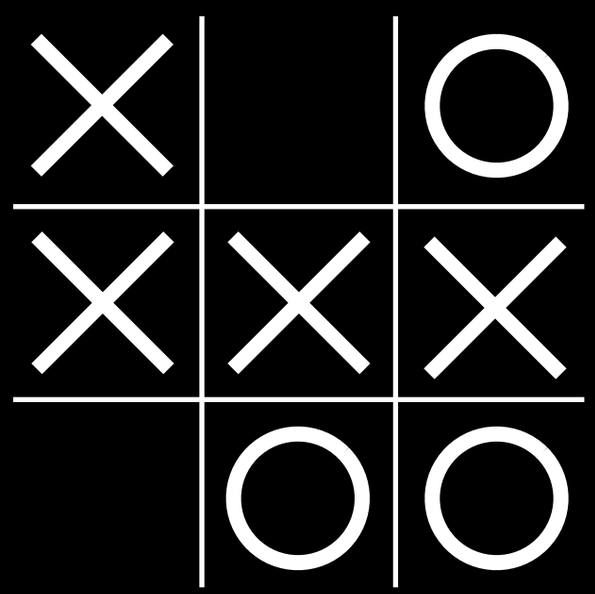
Outline

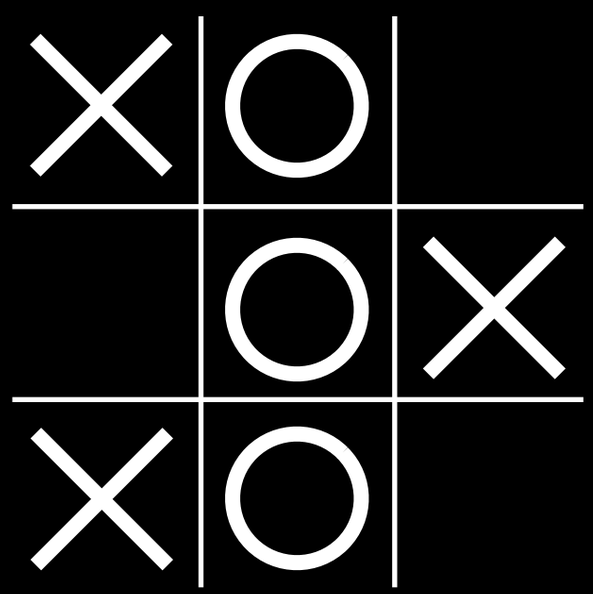
- In deterministic, perfect information, zero-sum games:
  - How to find optimal moves in deterministic games
  - How to find good moves when time is limited

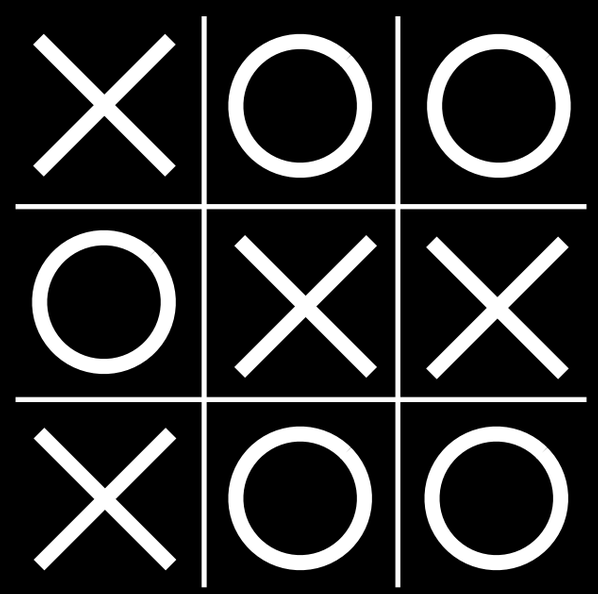


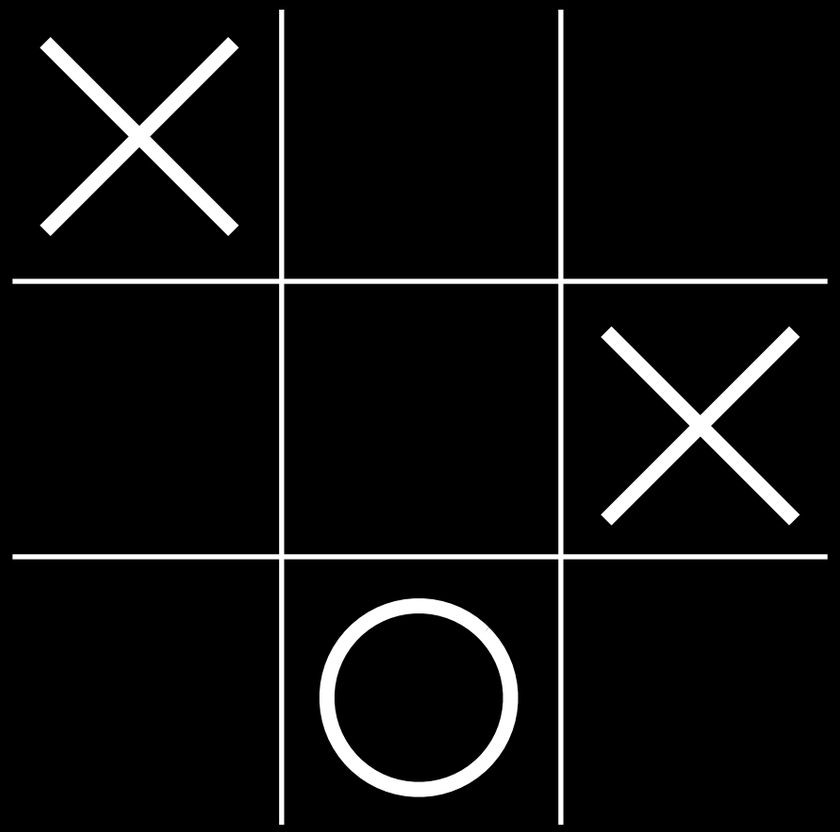


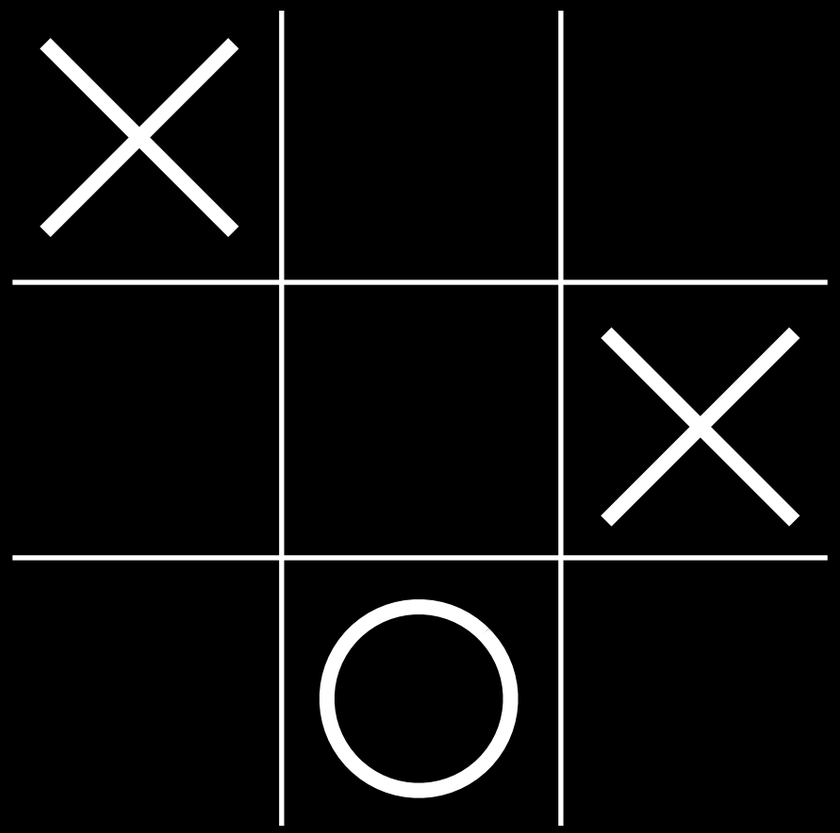




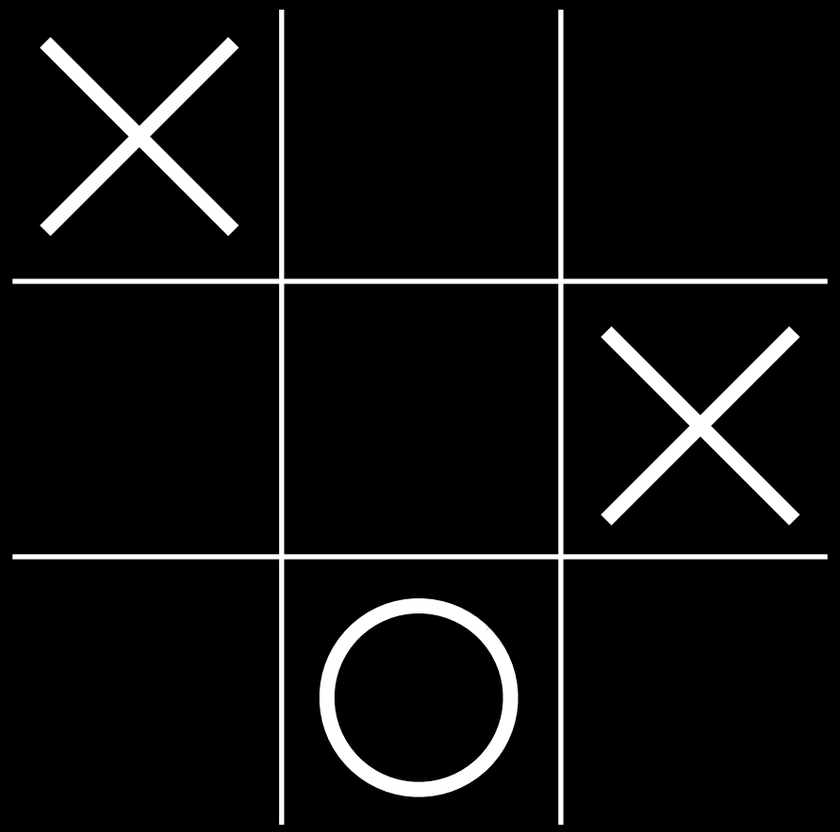




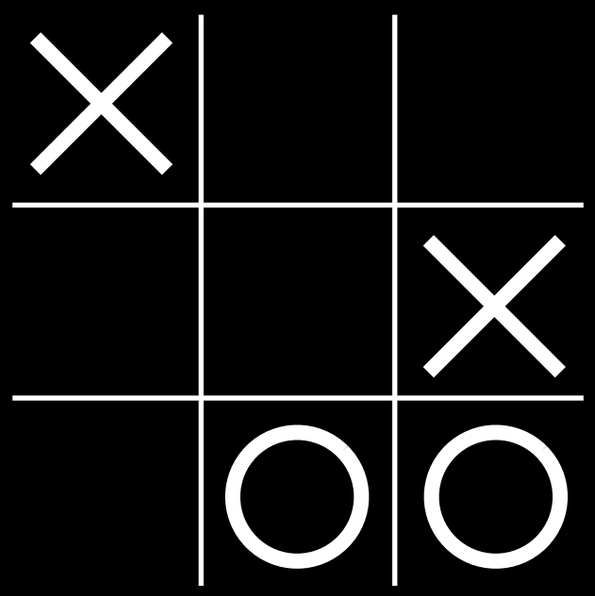


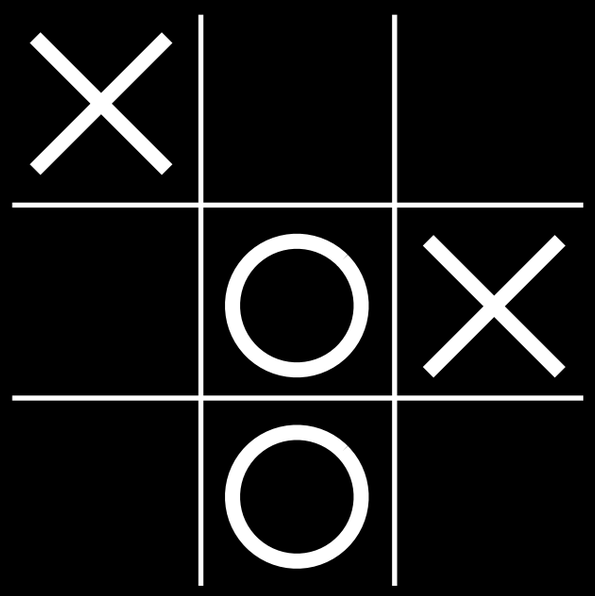


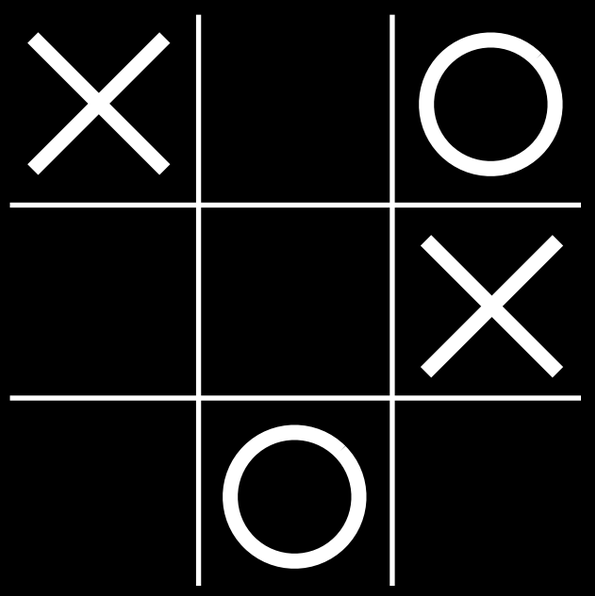
Next Player: O

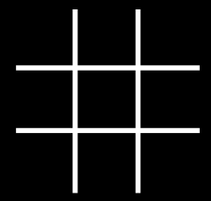


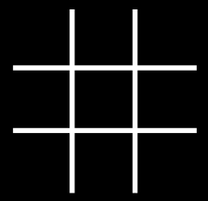
Next Player: O

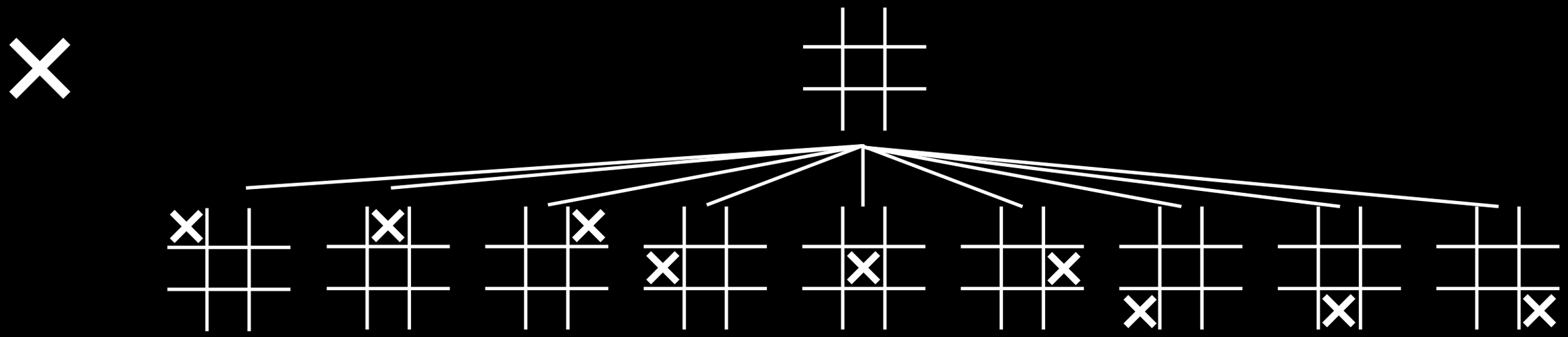


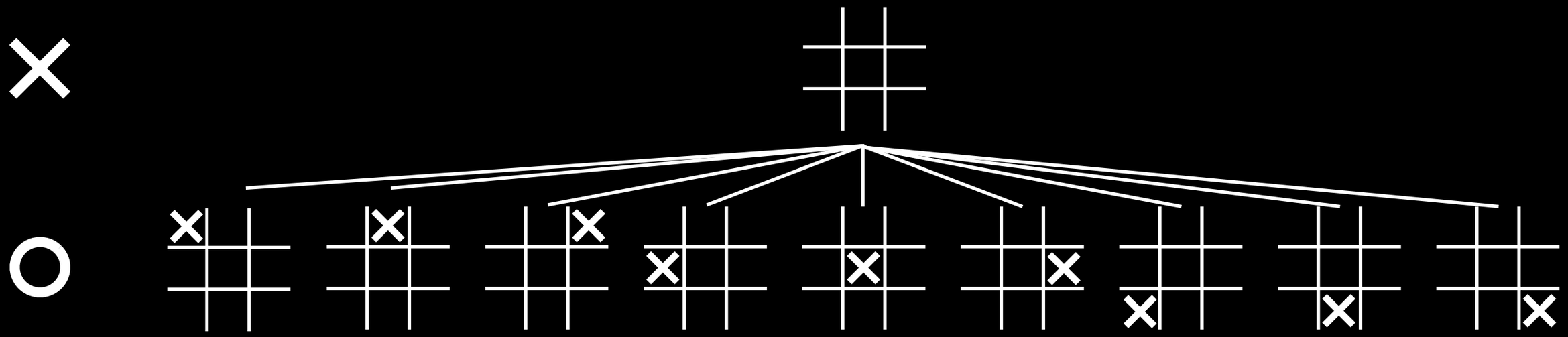


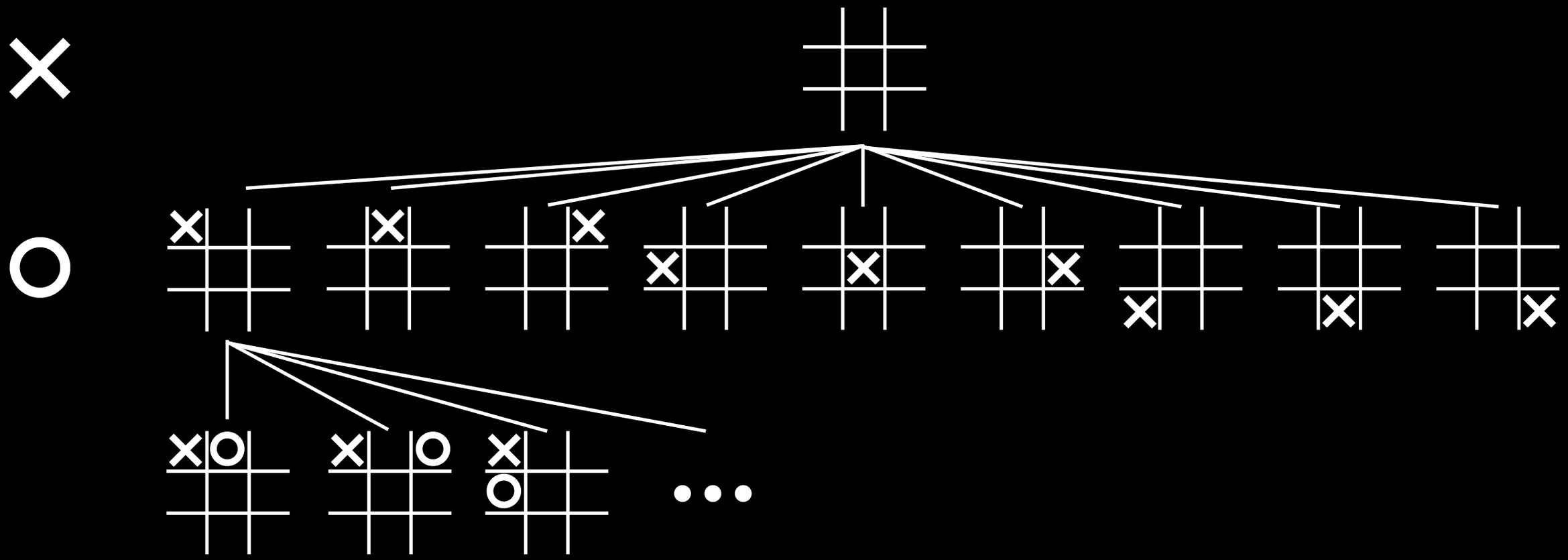


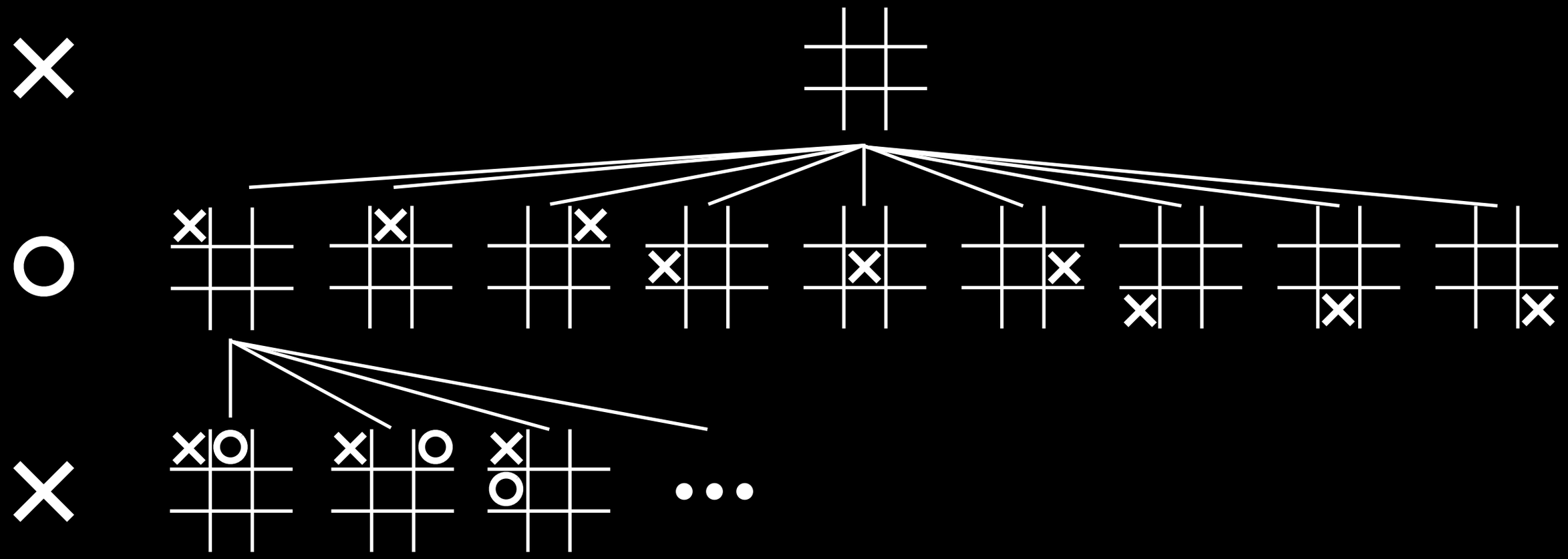


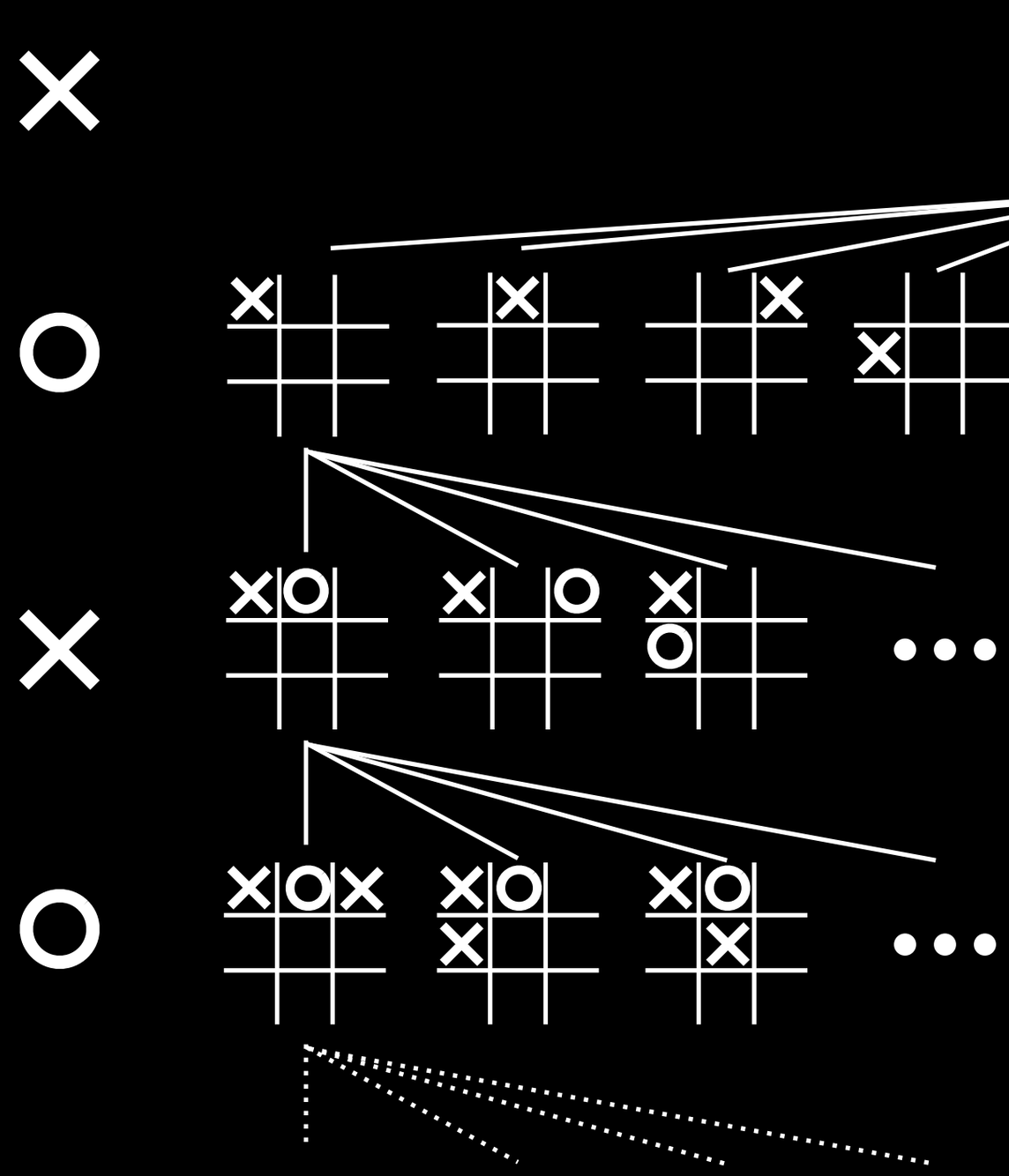




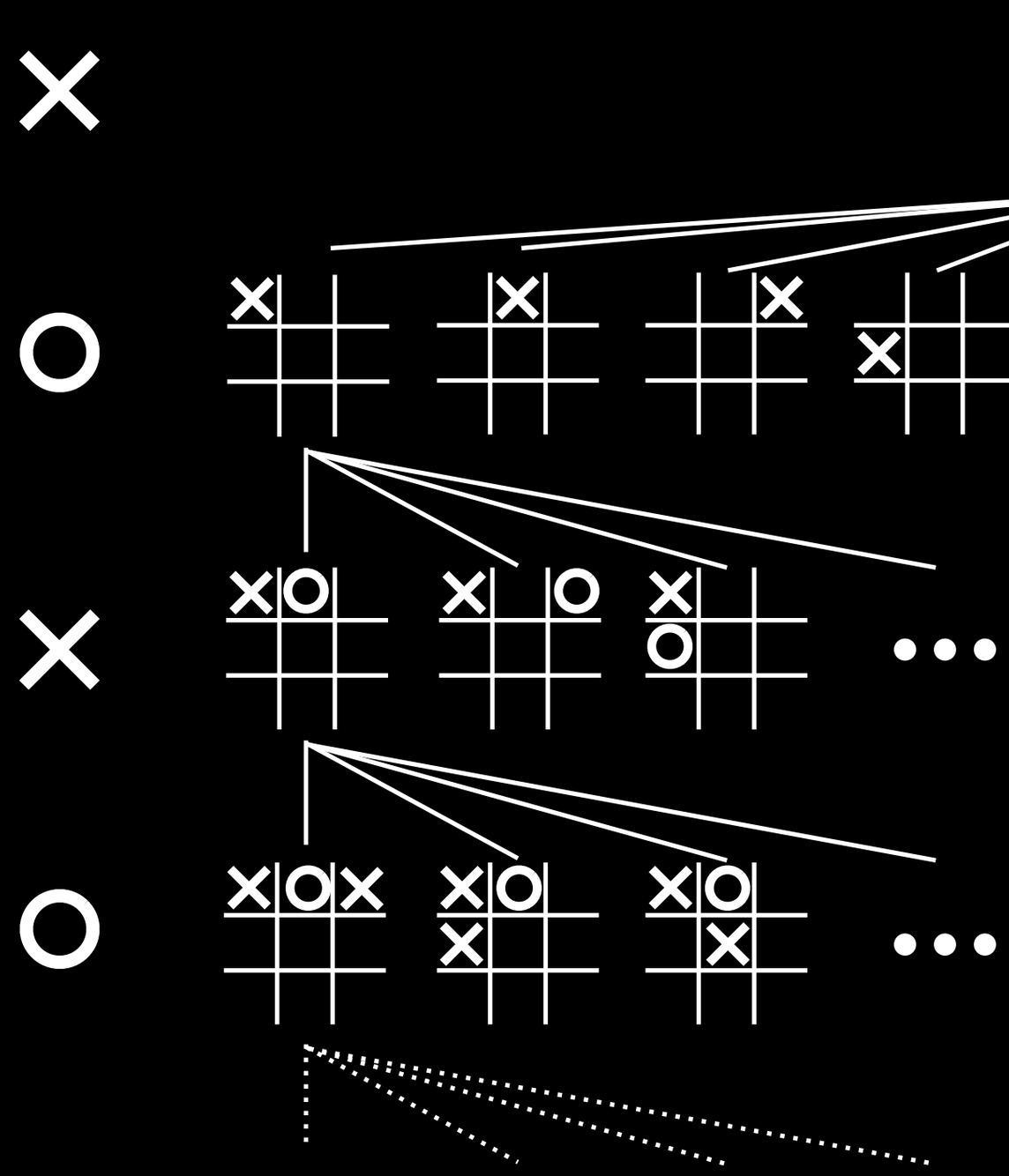




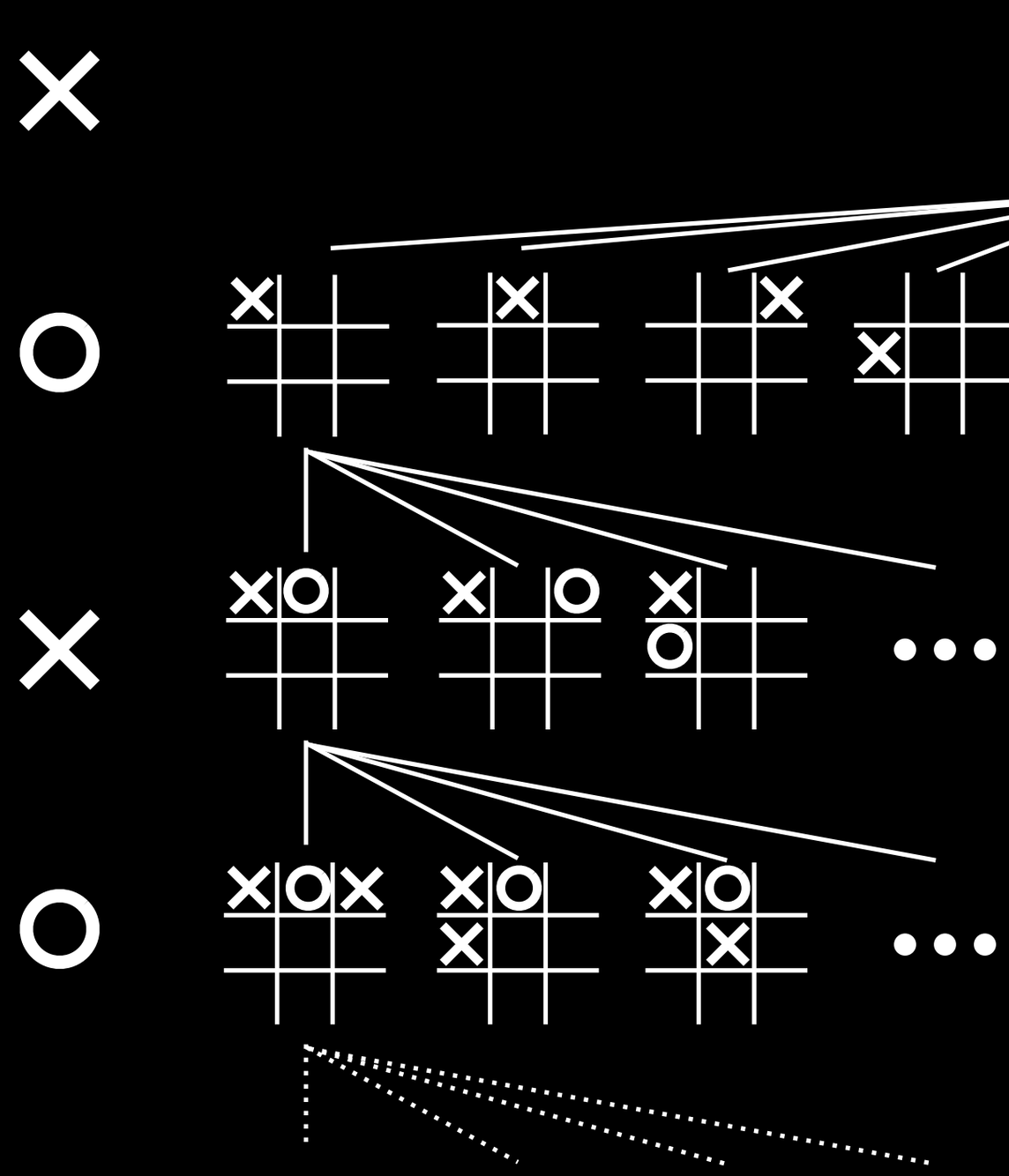

X
O
X
...



Term.



Term.

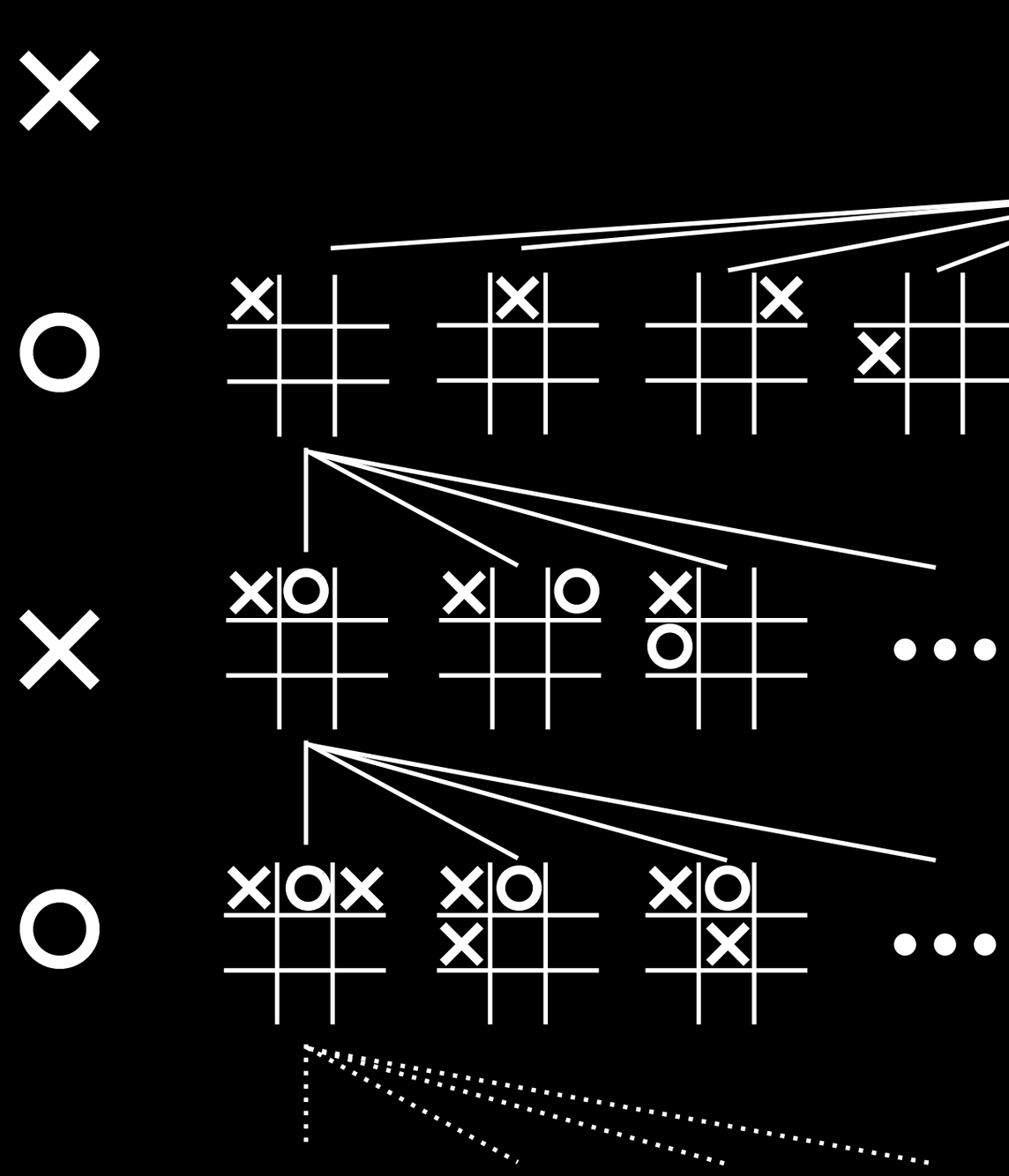


Term.

Deciding what to do in a game requires thinking about what the opponent will do, and having a strategy that takes that into account

Deciding what to do in a game requires thinking about what the opponent will do, and having a strategy that takes that into account

Not simply a sequence of actions, but a contingency plan that specifies what to do depending on what state one finds oneself in (AIMA 4.3)



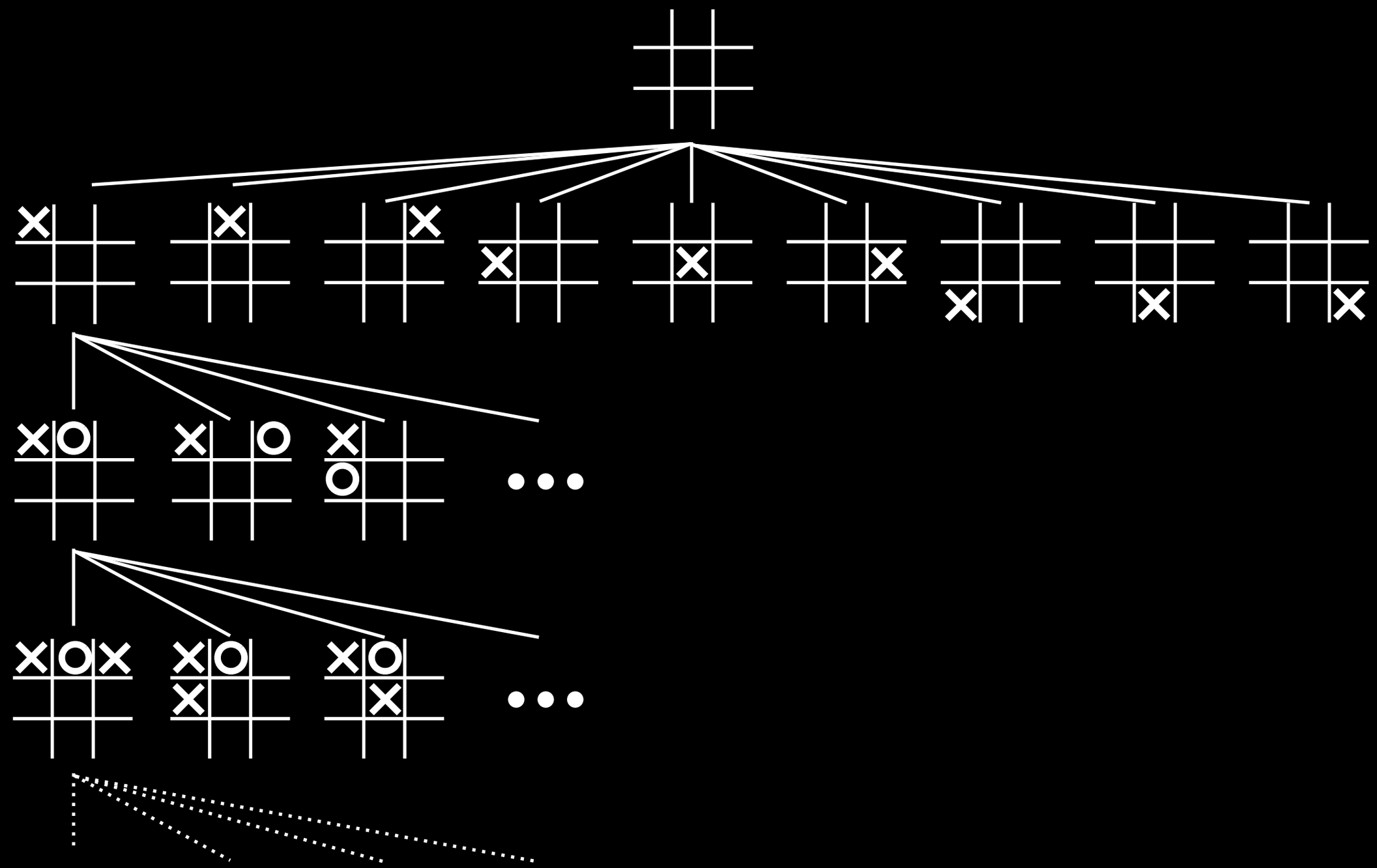
Term.

X

O

X

O



Term.

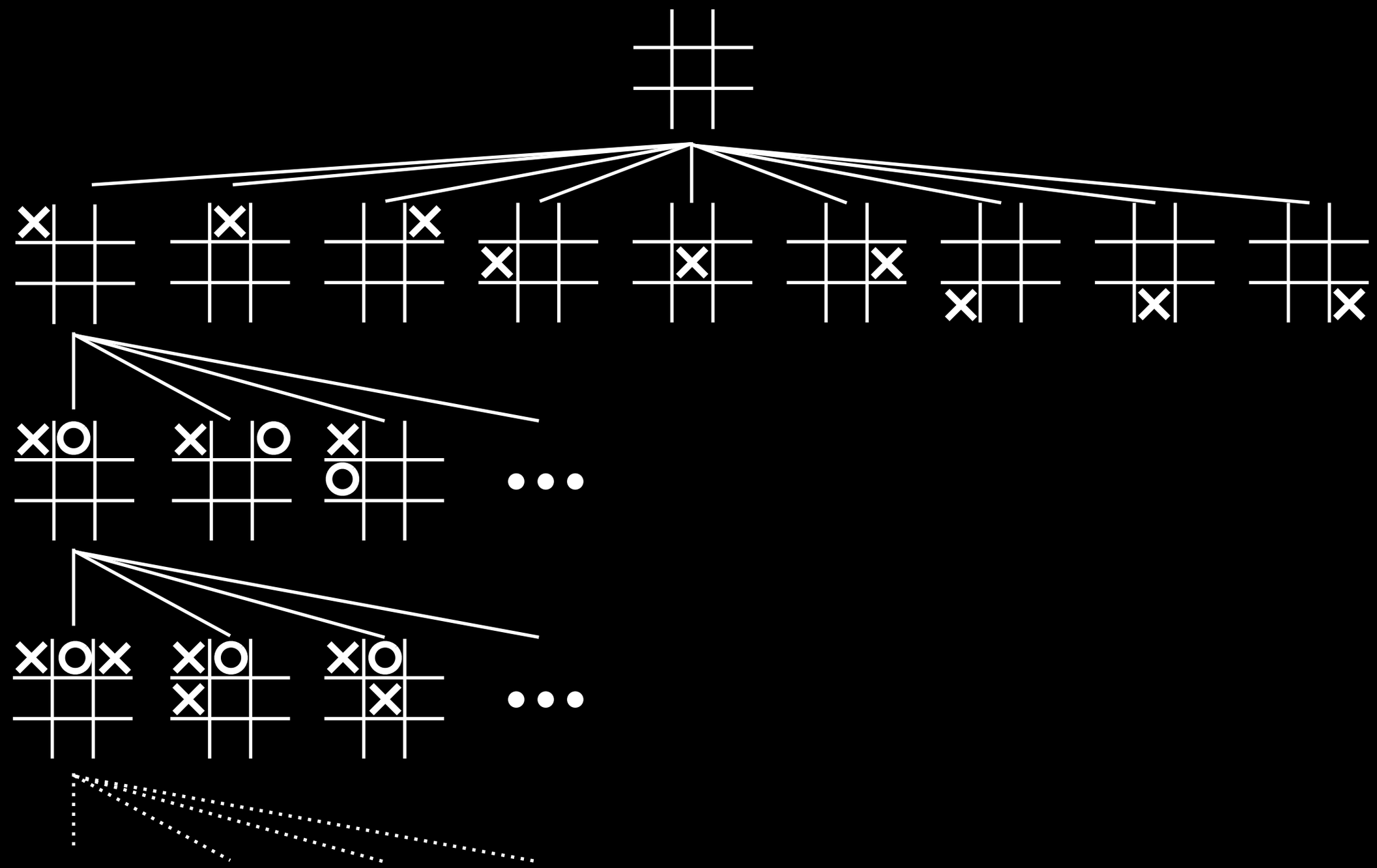
Utility -1 0 +1

X

O

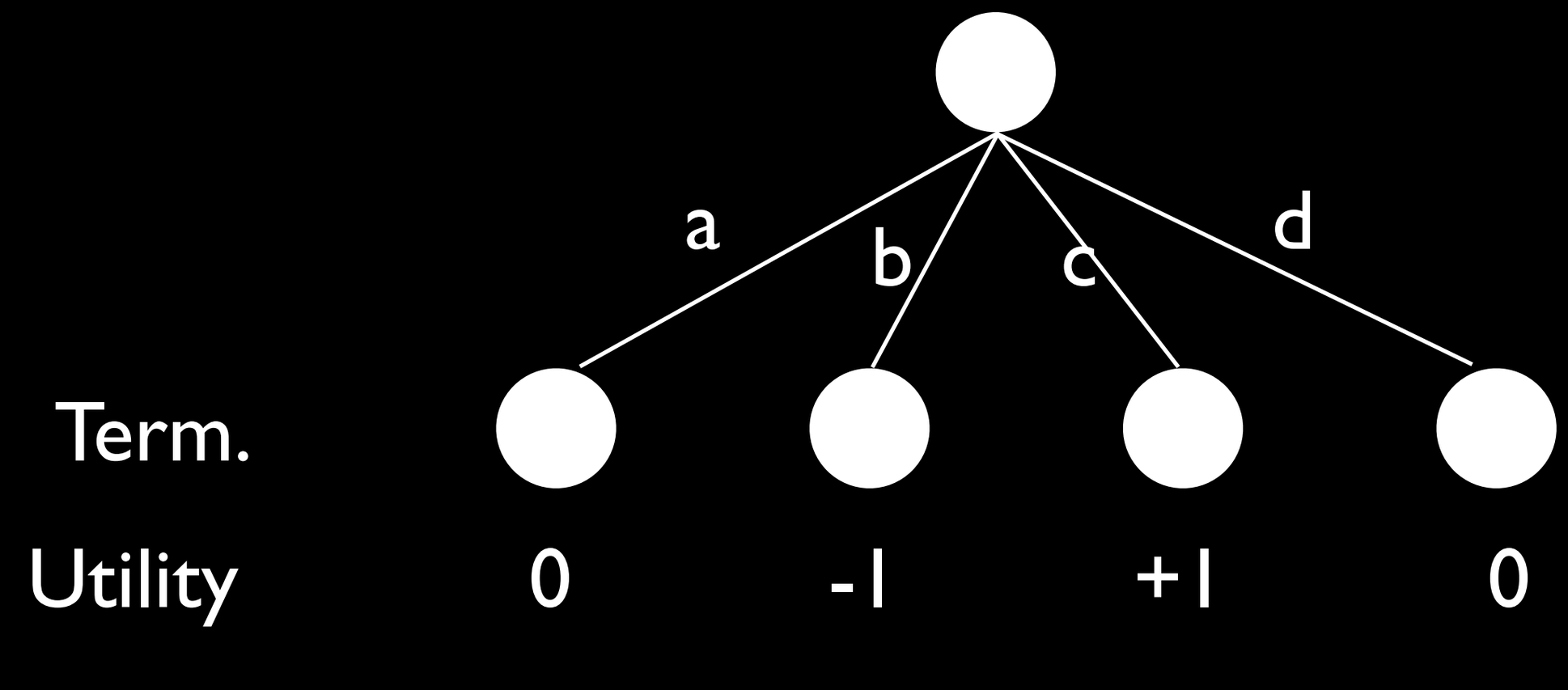
X

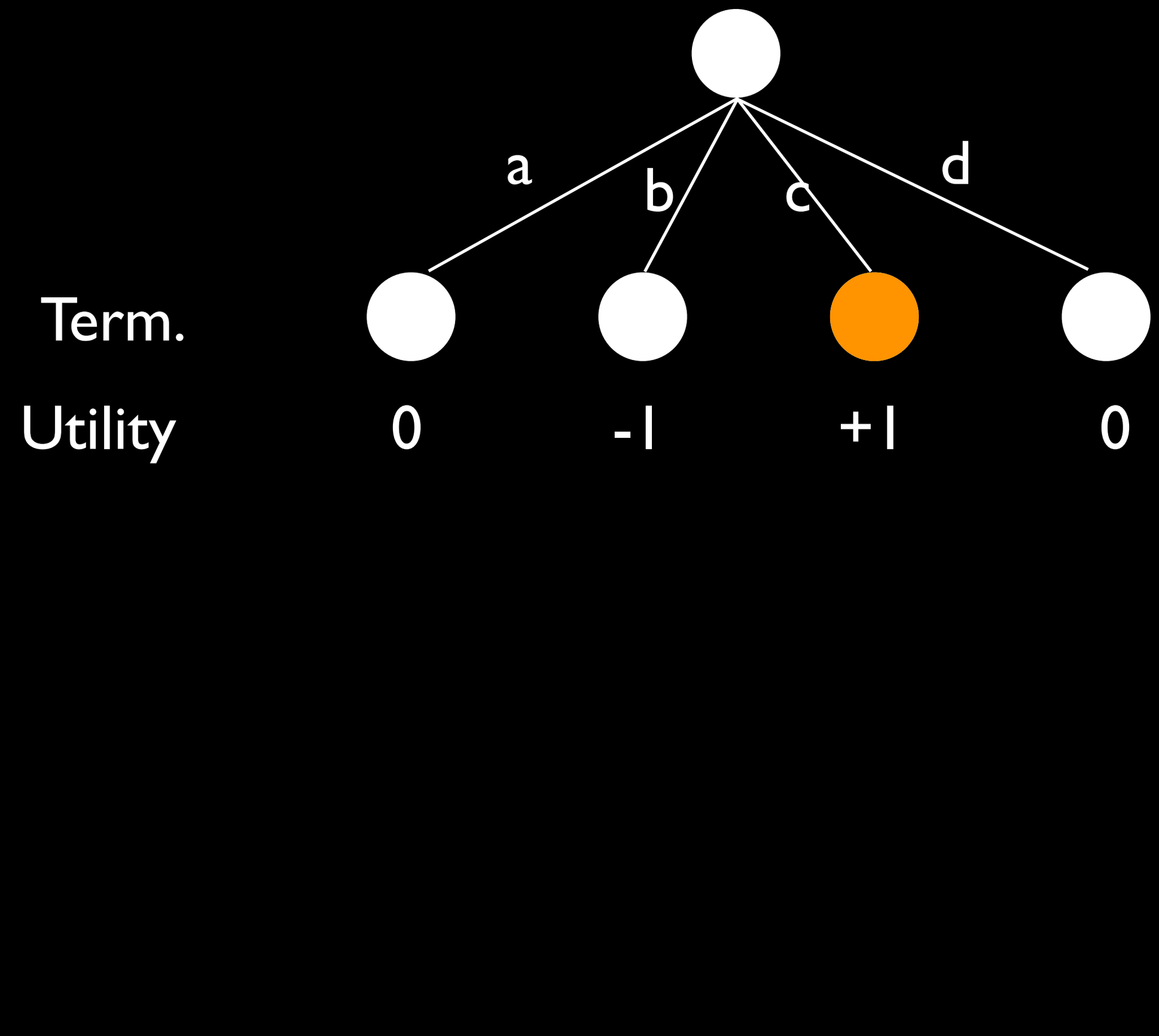
O



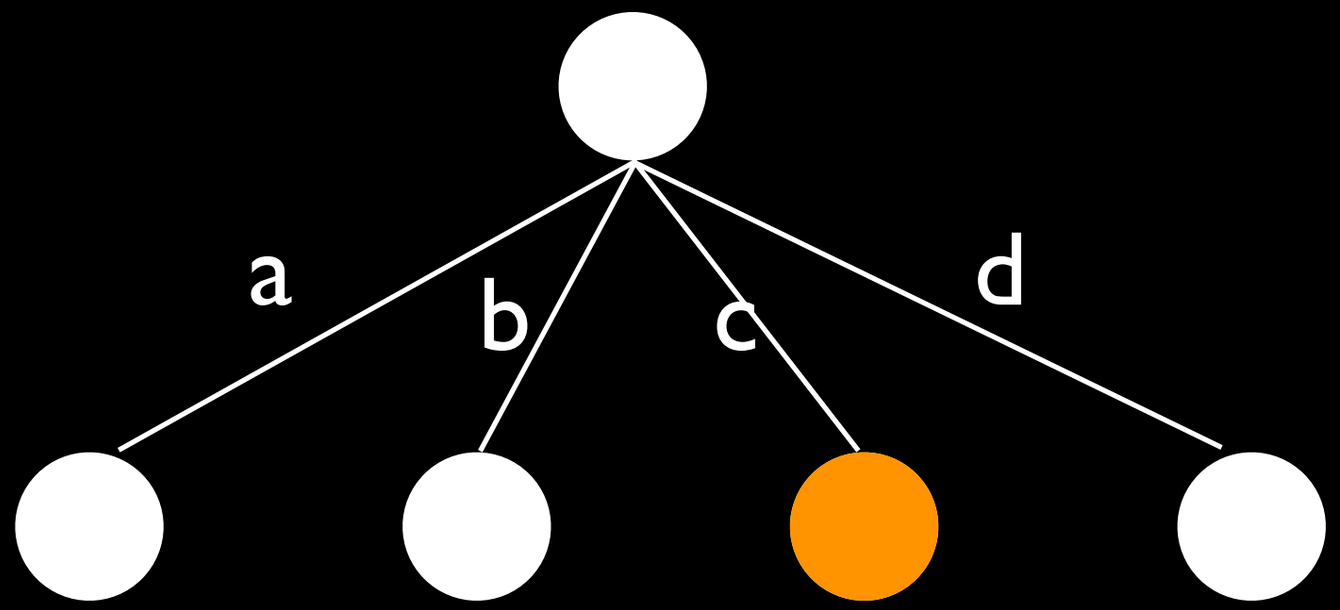
Term.

Utility -1 0 +1


a
b
c
d
Term.
Utility
0
-1
+1
0

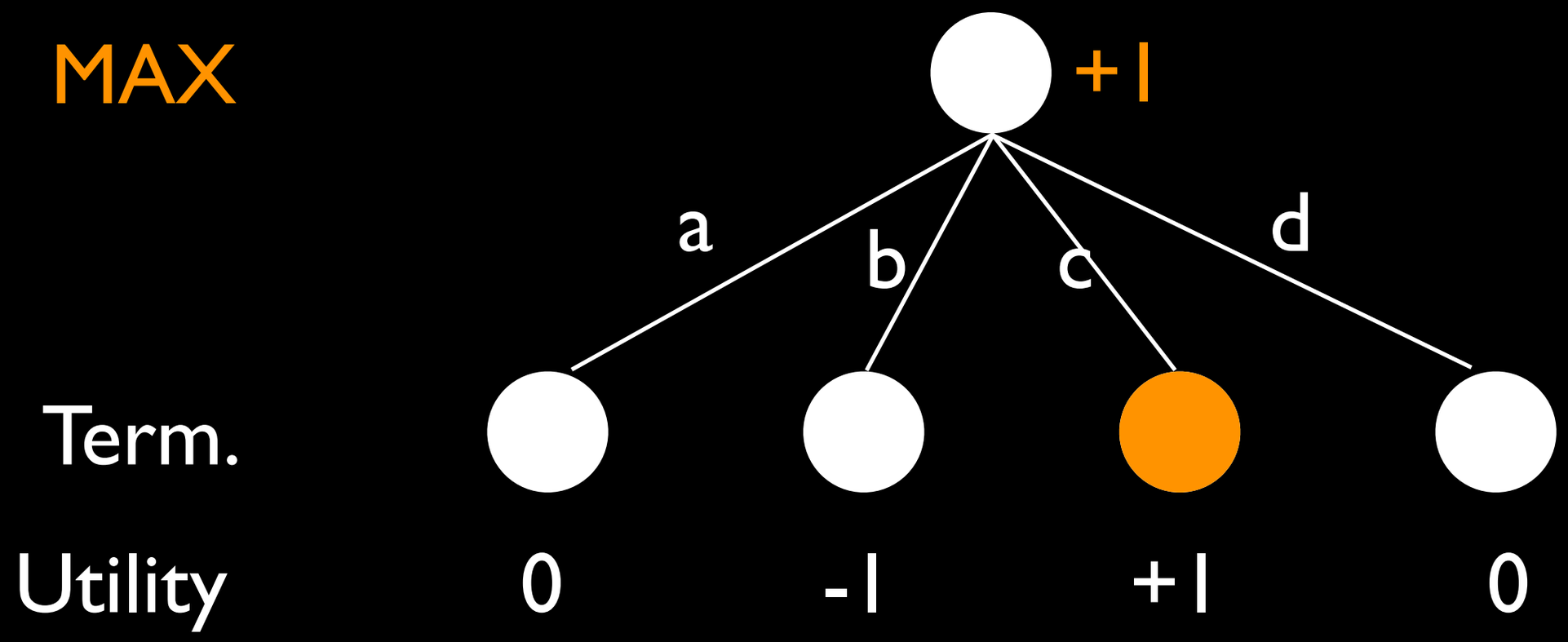

a
b
c
d
Term.
Utility
0
-1
+1
0

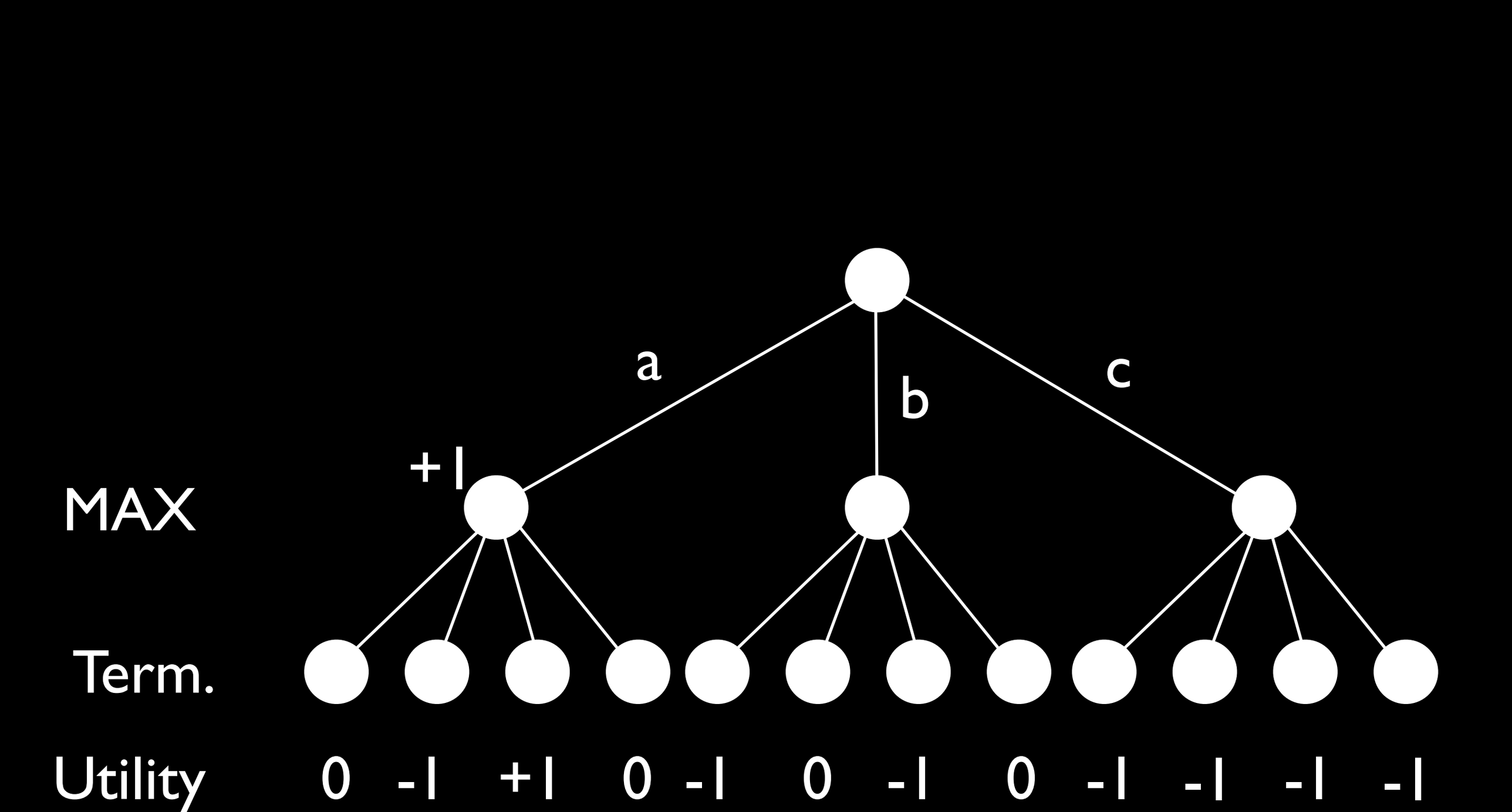
MAX

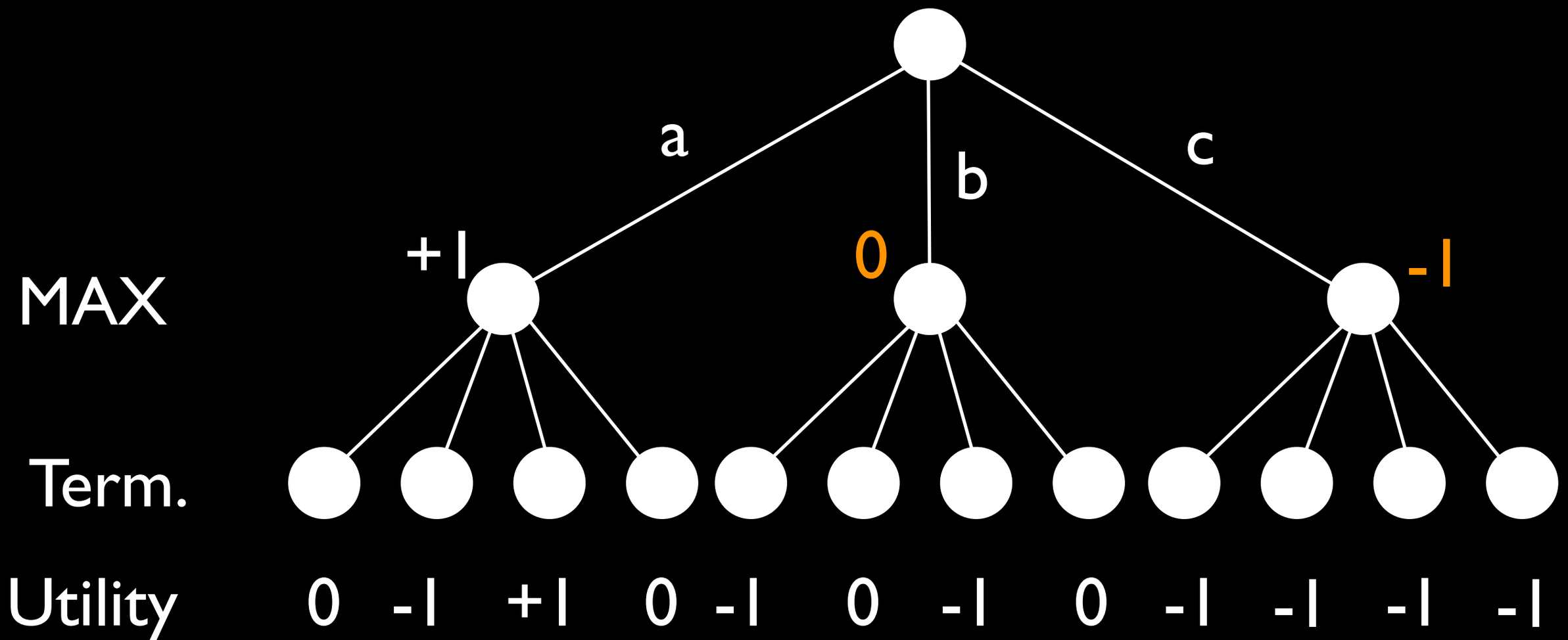


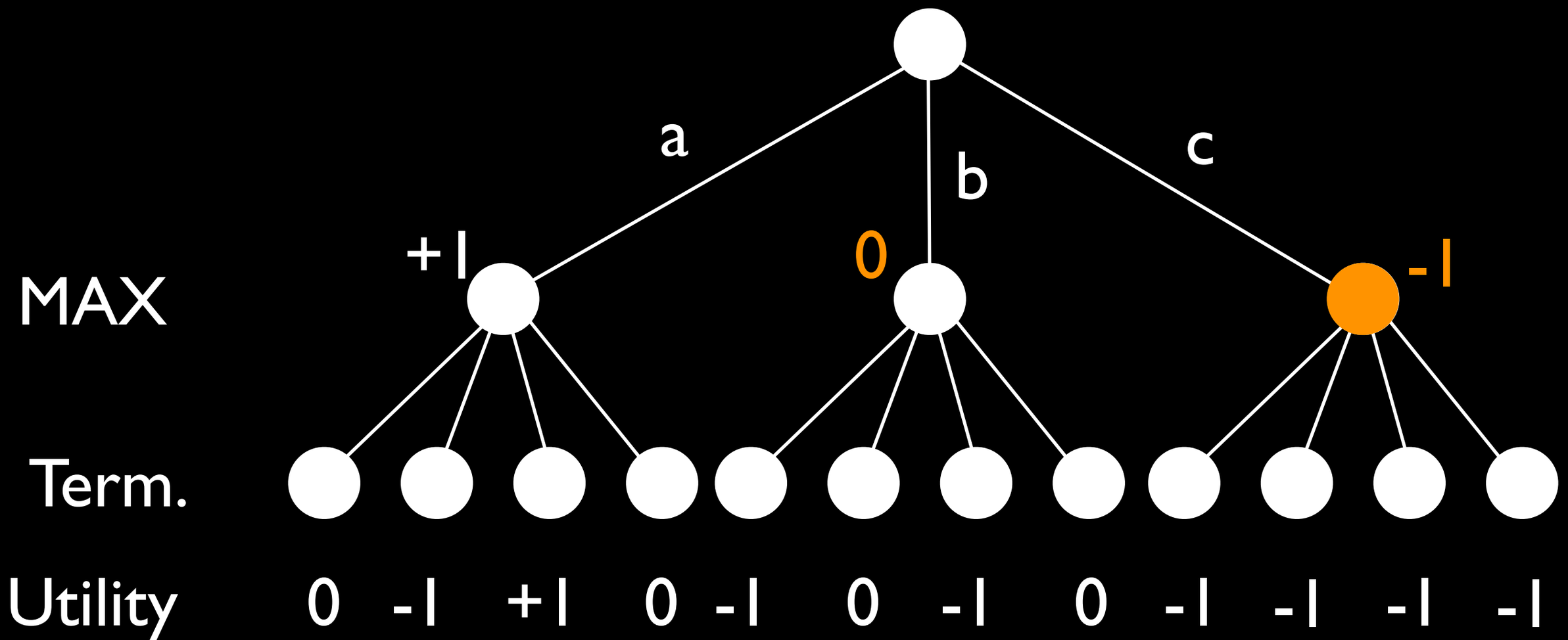
Term.

Utility 0 -1 +1 0


MAX
+1
a
b
c
d
Term.
Utility
0
-1
+1
0

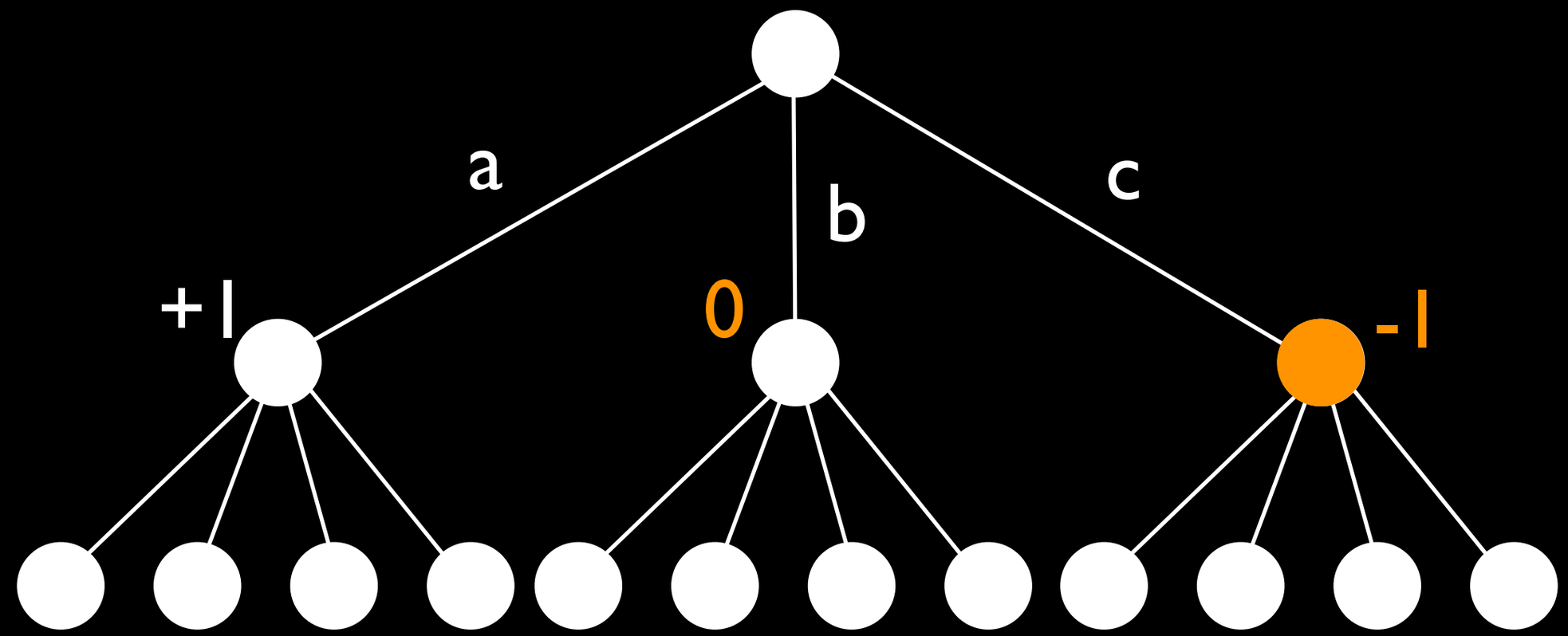

a
b
c
+1
MAX
Term.
Utility
0 -1 +1 0 -1 0 -1 0 -1 -1 -1 -1


a
b
c
MAX
+1
0
-1
Term.
Utility
0 -1 +1 0 -1 0 -1 0 -1 -1 -1 -1


a
b
c
MAX
+1
0
-1
Term.
Utility
0 -1 +1 0 -1 0 -1 0 -1 -1 -1 -1

MIN

MAX



Term.

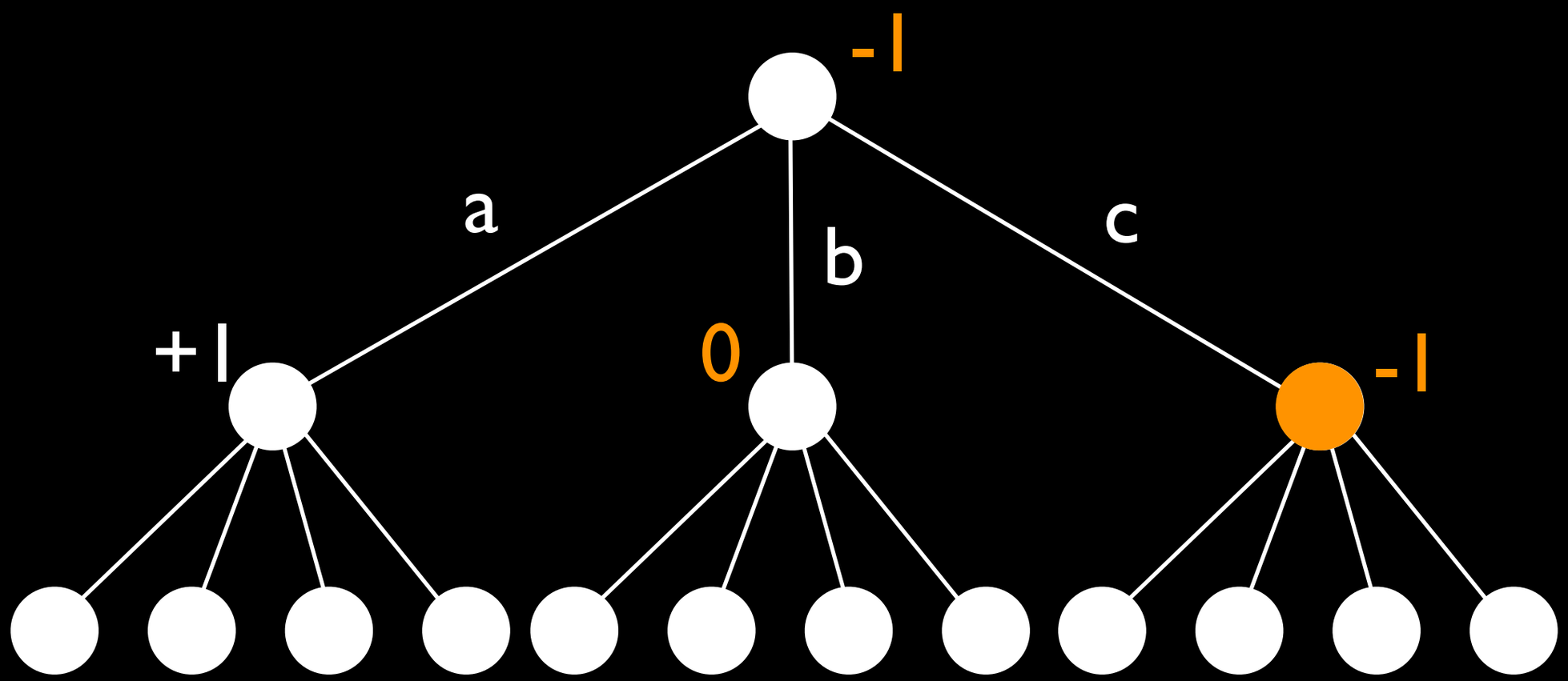
Utility 0 -1 +1 0 -1 0 -1 0 -1 -1 -1 -1

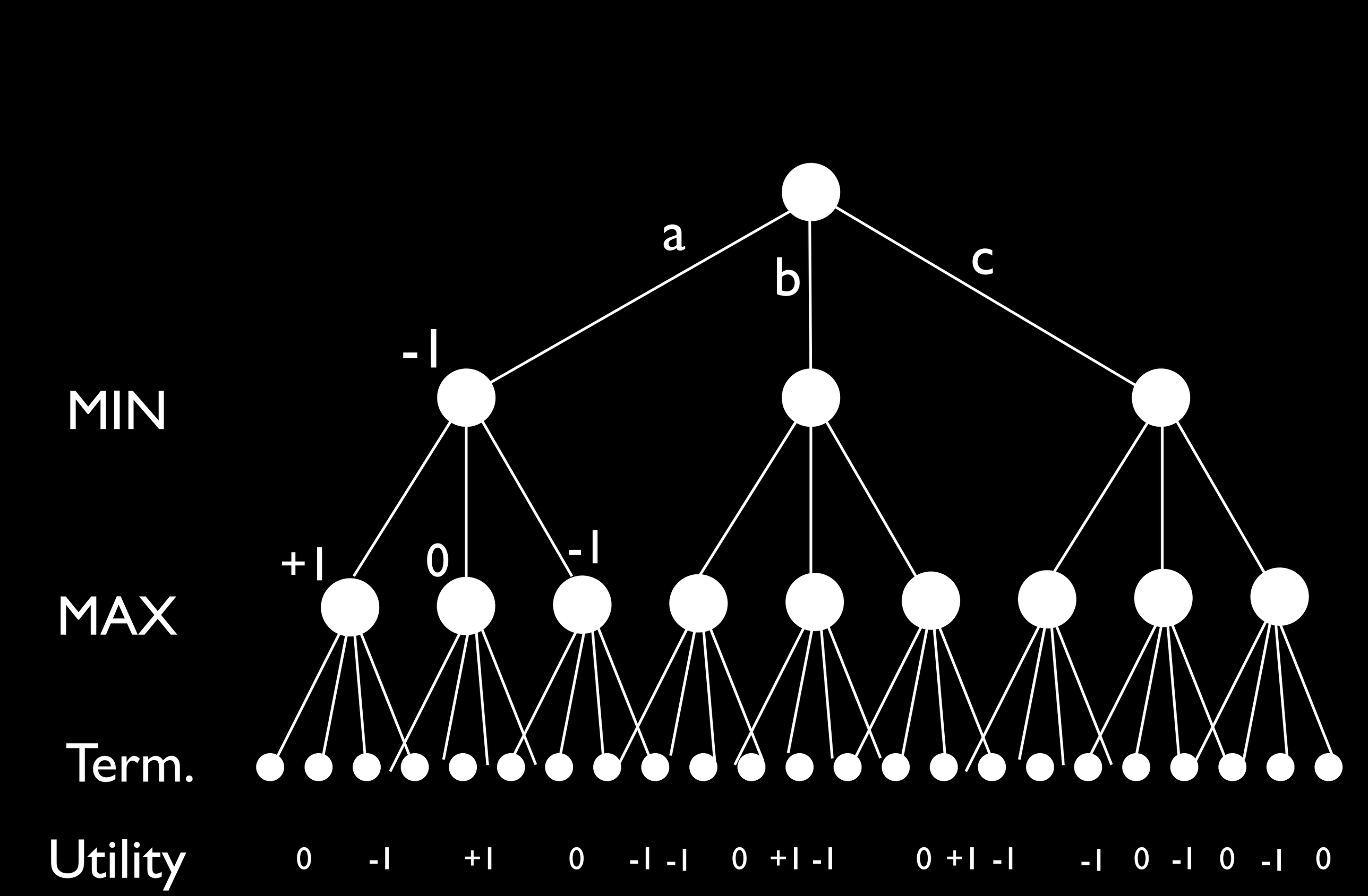
MIN

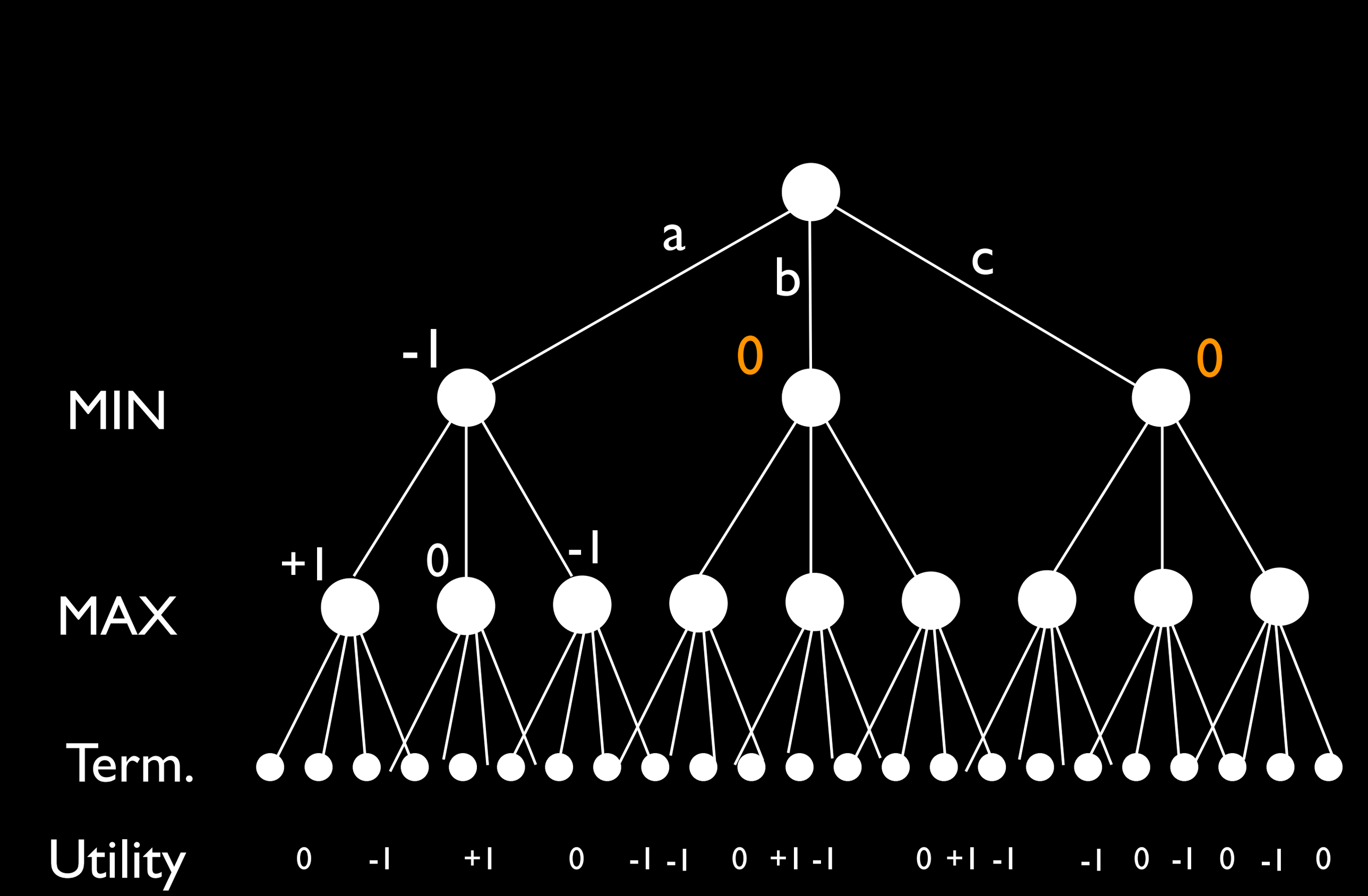
MAX

Term.

Utility




a
b
c
-1
MIN
+1
0
-1
MAX
Term.
Utility
0 -1 +1 0 -1 -1 0 +1 -1 0 +1 -1 -1 0 -1 0 -1 0


a
b
c
-1
0
0
MIN
+1
0
-1
MAX
Term.
Utility
0 -1 +1 0 -1 -1 0 +1 -1 0 +1 -1 -1 0 -1 0 -1 0

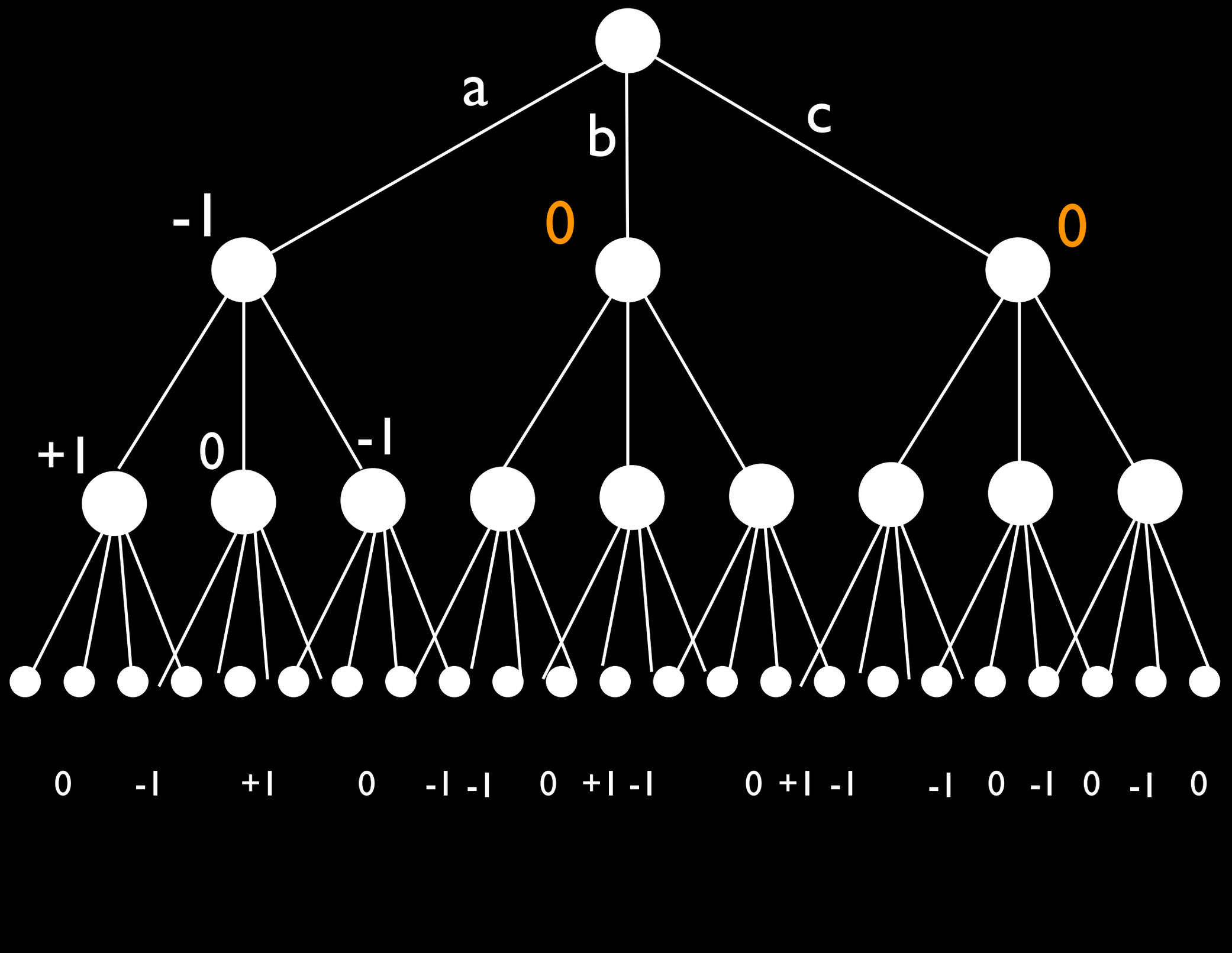
MAX

MIN

MAX

Term.

Utility



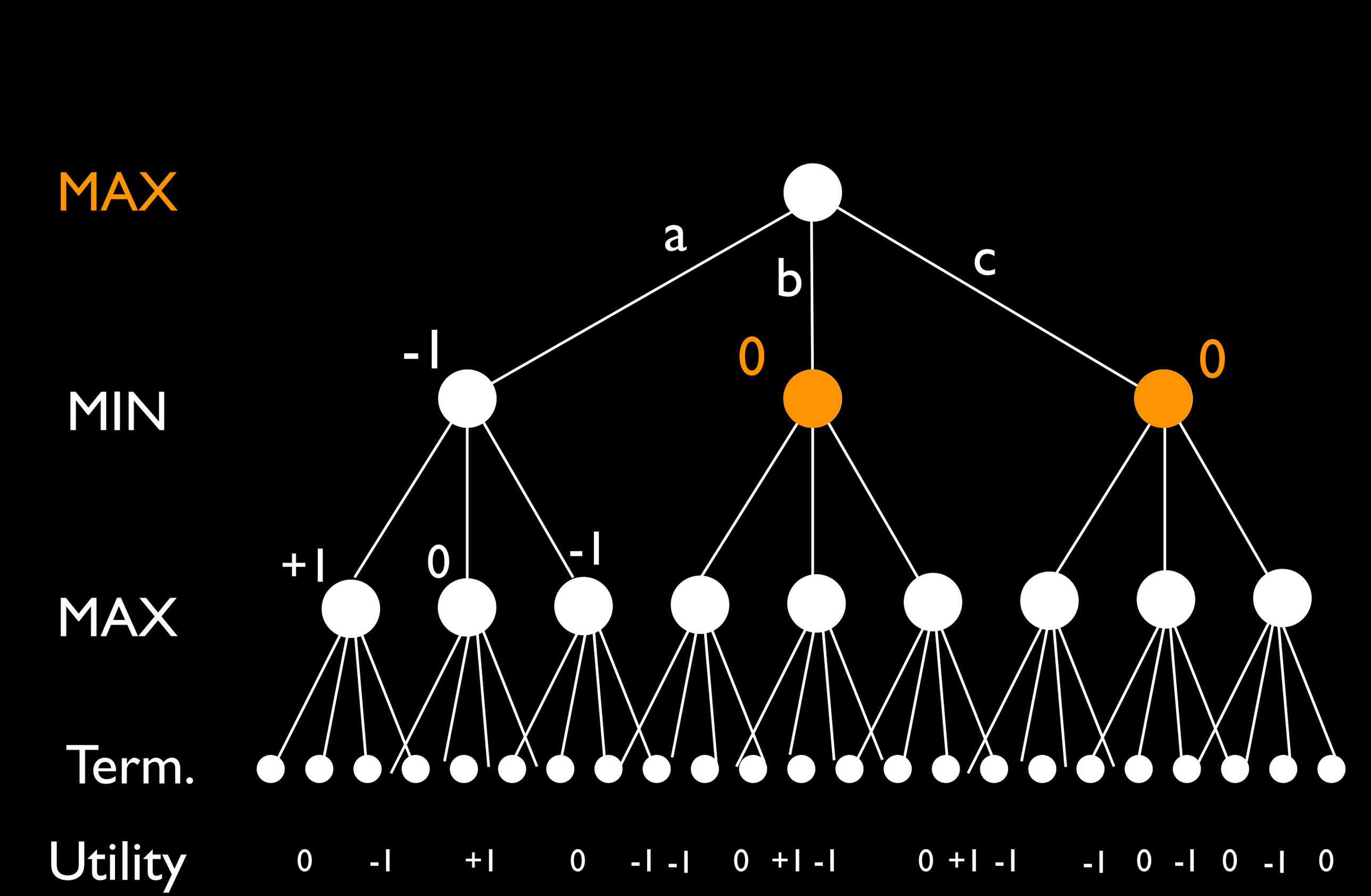
MAX

MIN

MAX

Term.

Utility



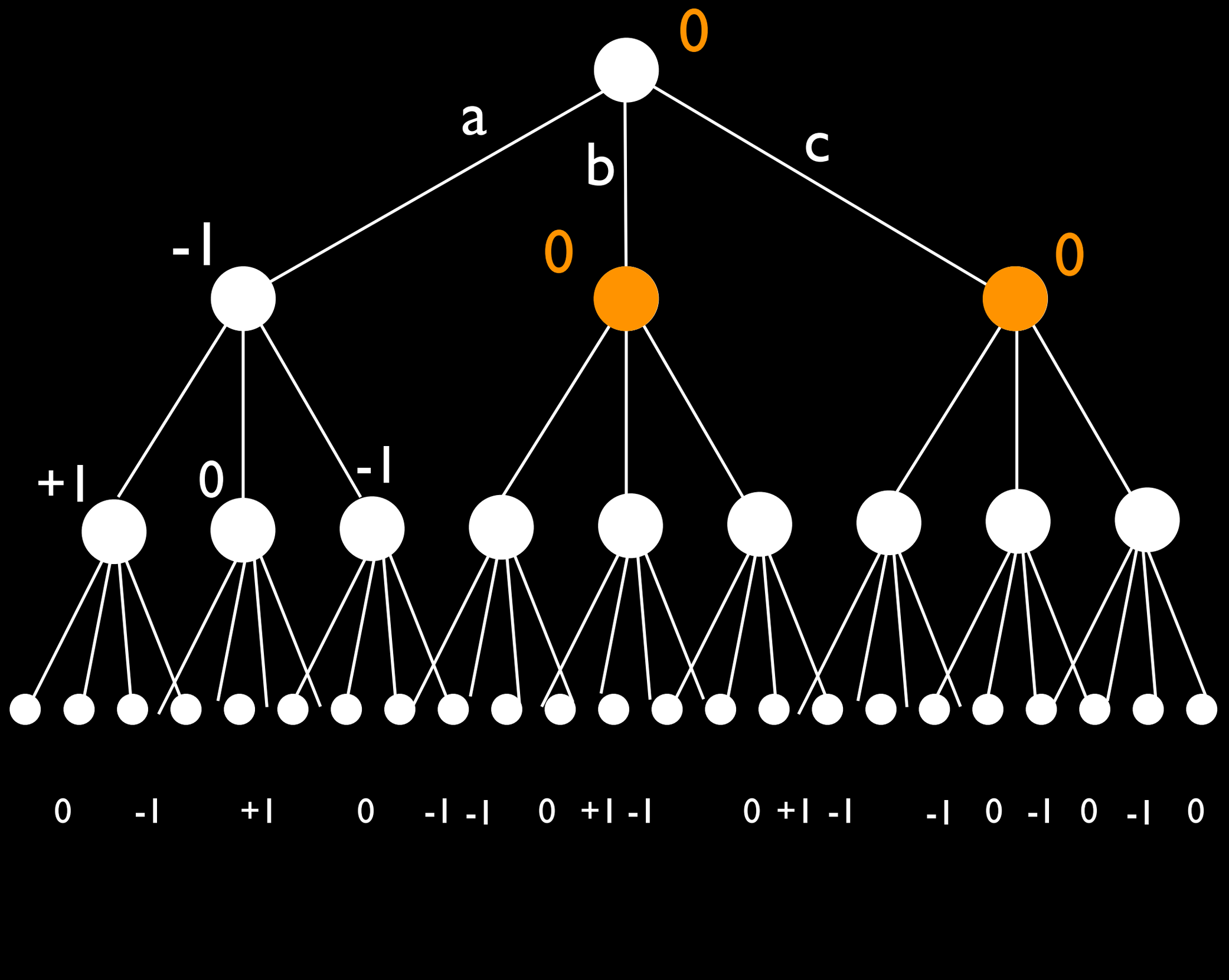
MAX

MIN

MAX

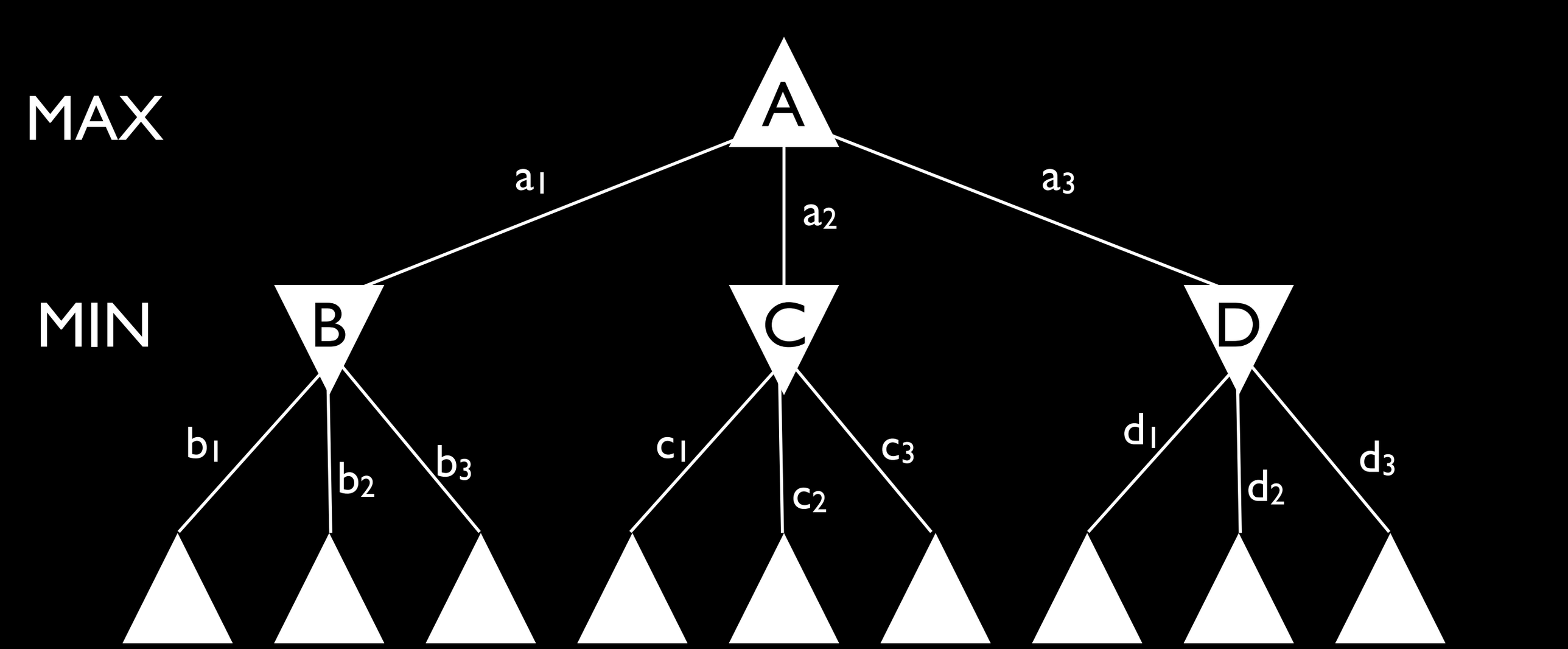
Term.

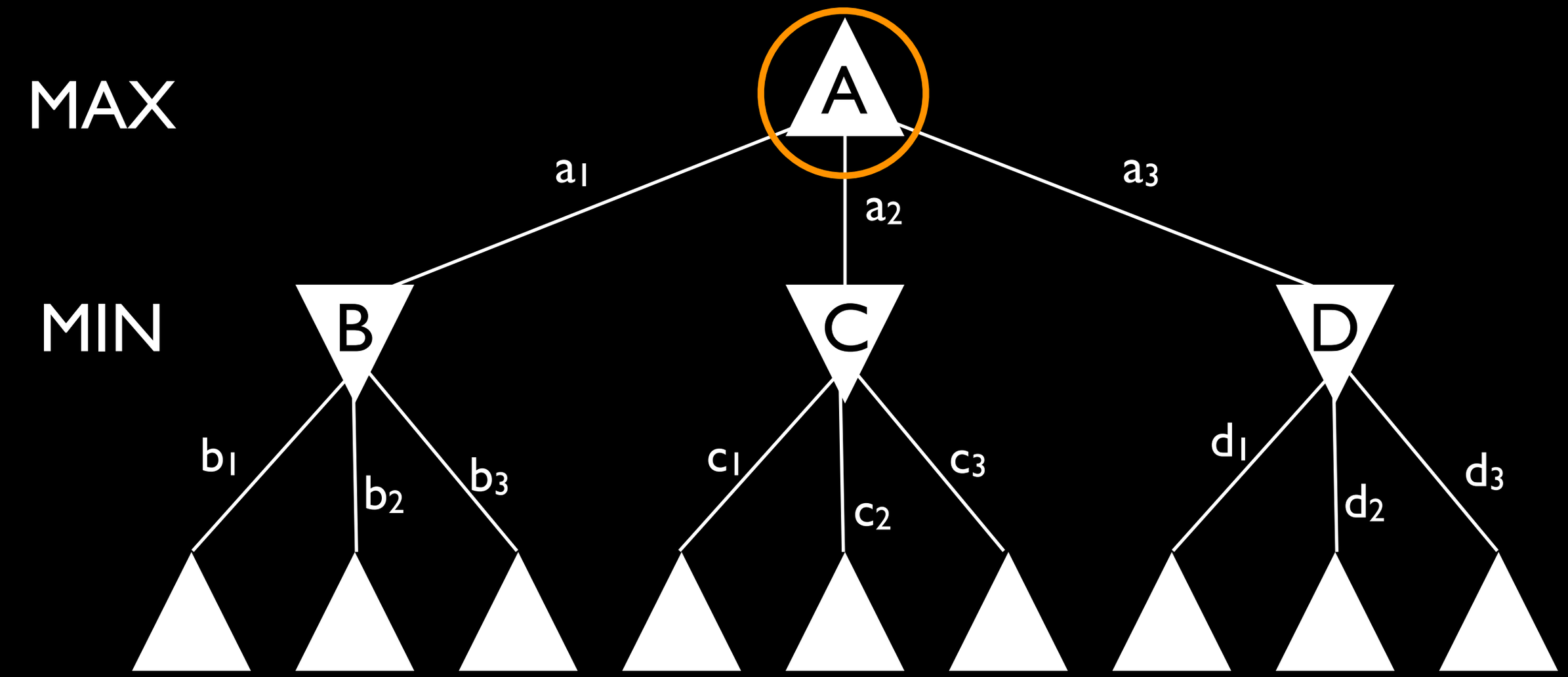
Utility

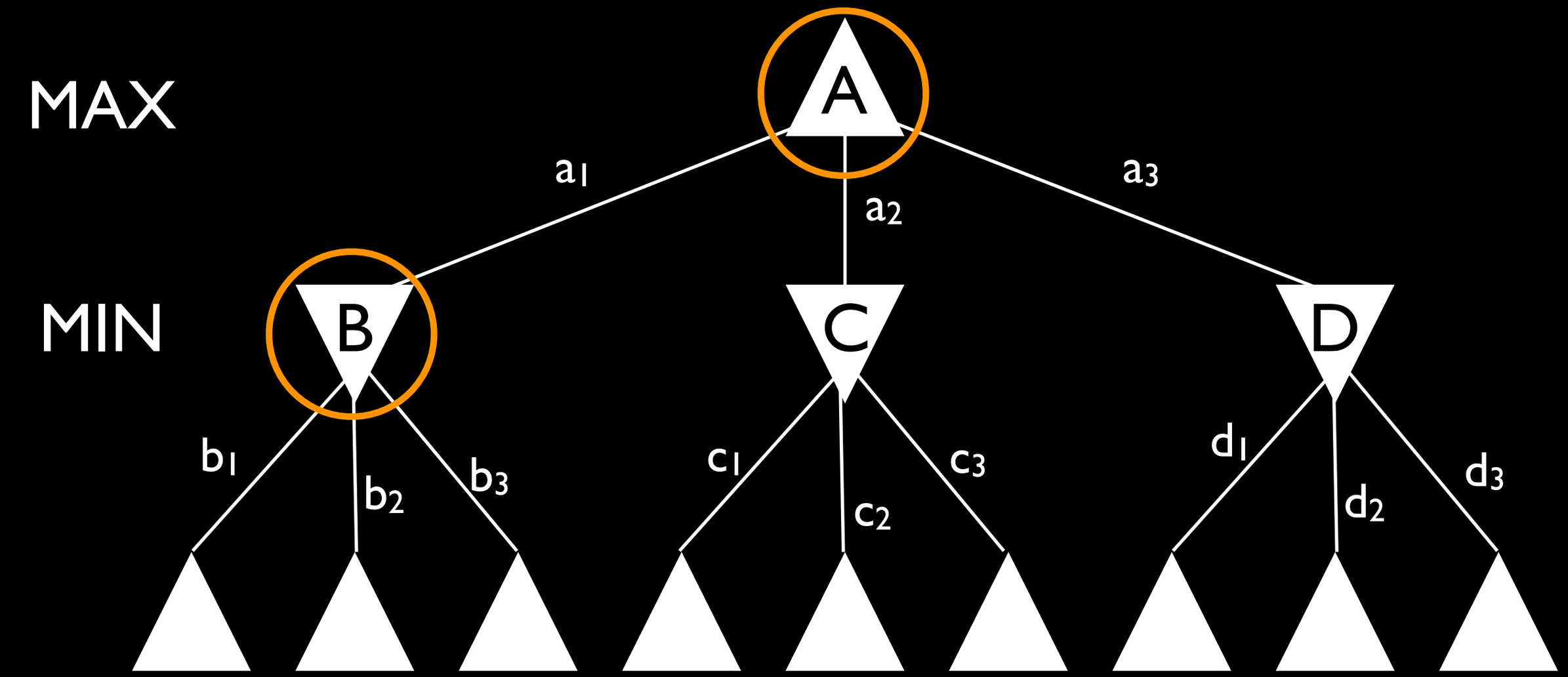


Minimax Algorithm

$$\textsc{Minimax}(s) = \begin{cases} \textsc{Utility}(s) & \text{if } \textsc{Terminal-Test}(s) \\ \max_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{if } \textsc{Player}(s) = \textsc{max} \\ \min_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{if } \textsc{Player}(s) = \textsc{min} \end{cases}$$


MAX
A
a_1
a_2
a_3
MIN
B
C
D
b_1
b_2
b_3
c_1
c_2
c_3
d_1
d_2
d_3


MAX
MIN
A
a1
a2
a3
B
C
D
b1
b2
b3
c1
c2
c3
d1
d2
d3


MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3

MAX

MIN

A

B

C

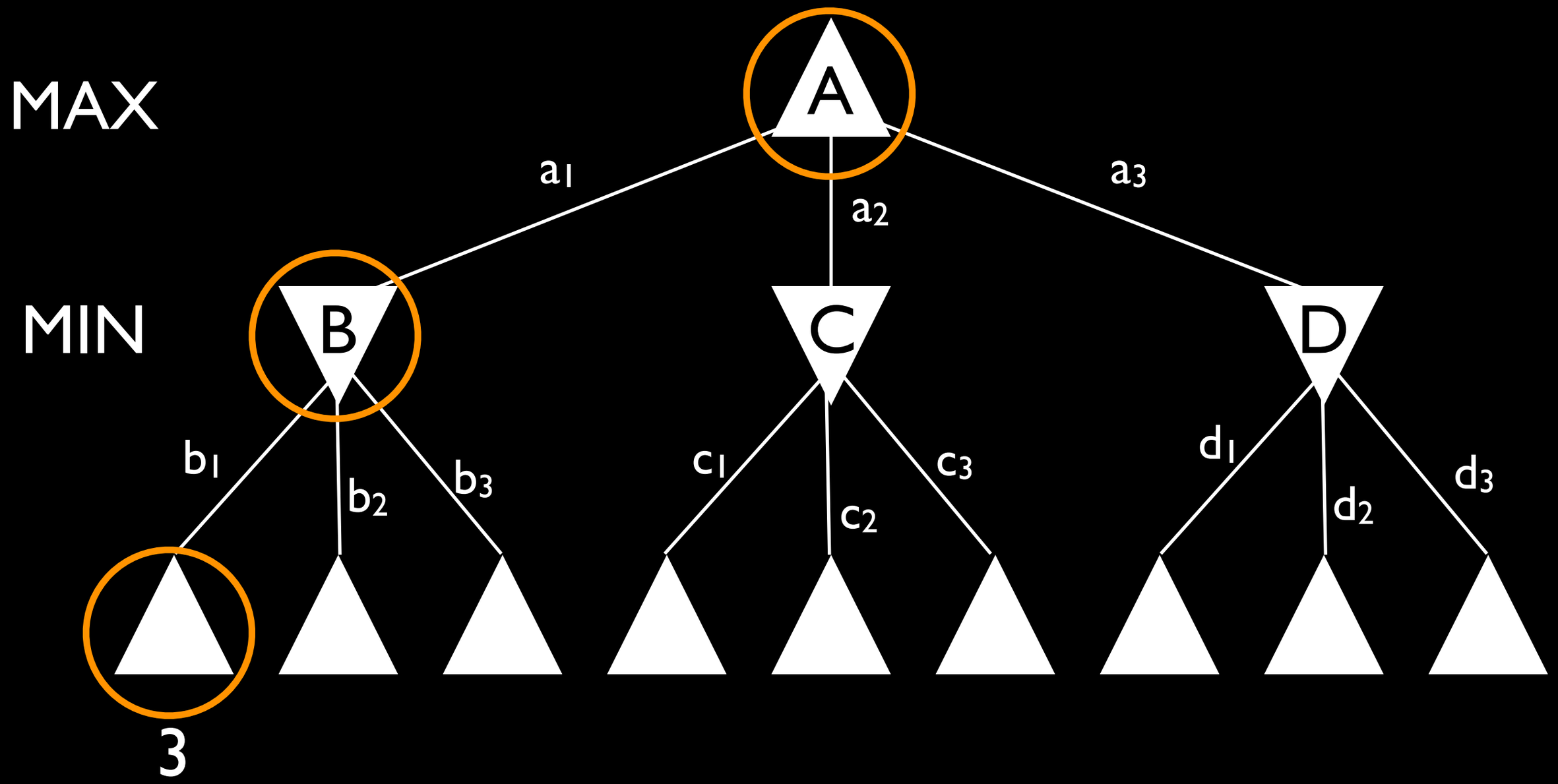
D

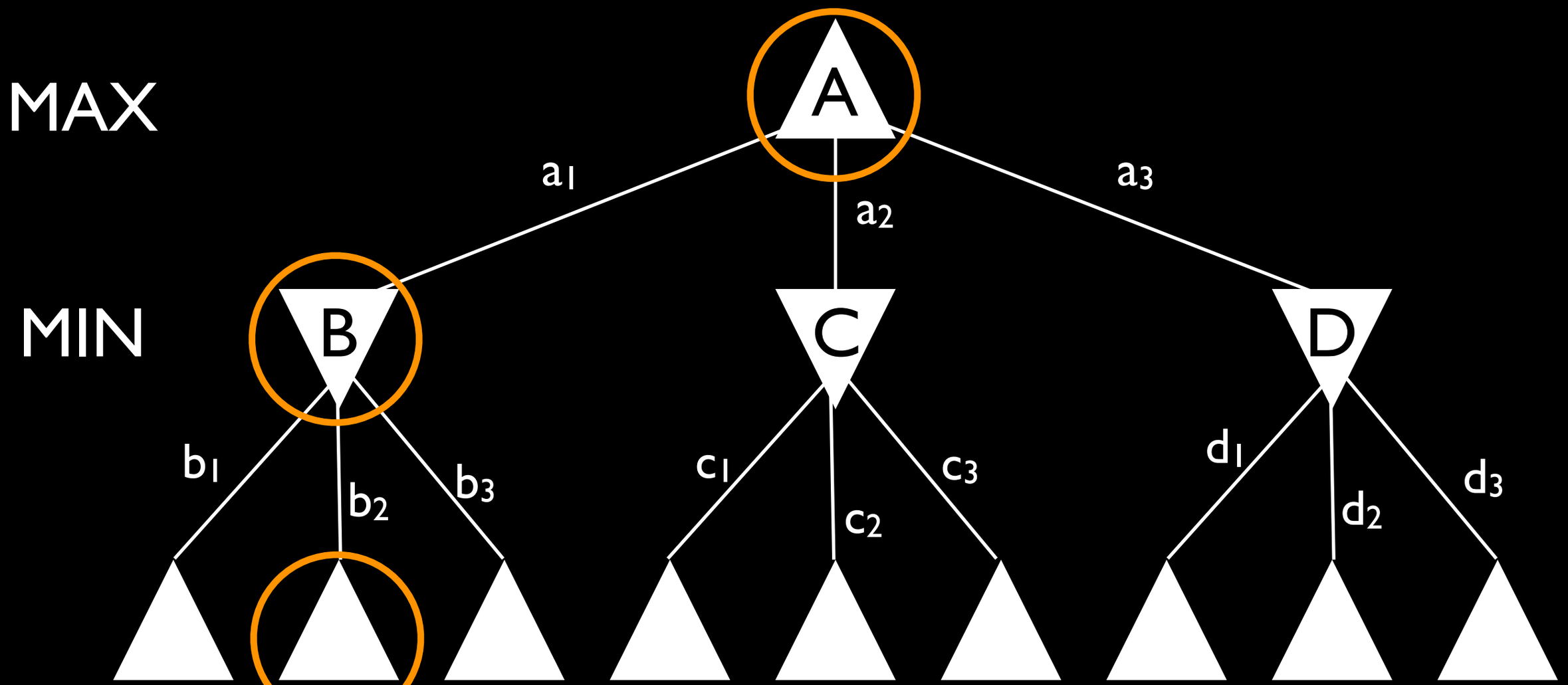
a_1 a_2 a_3

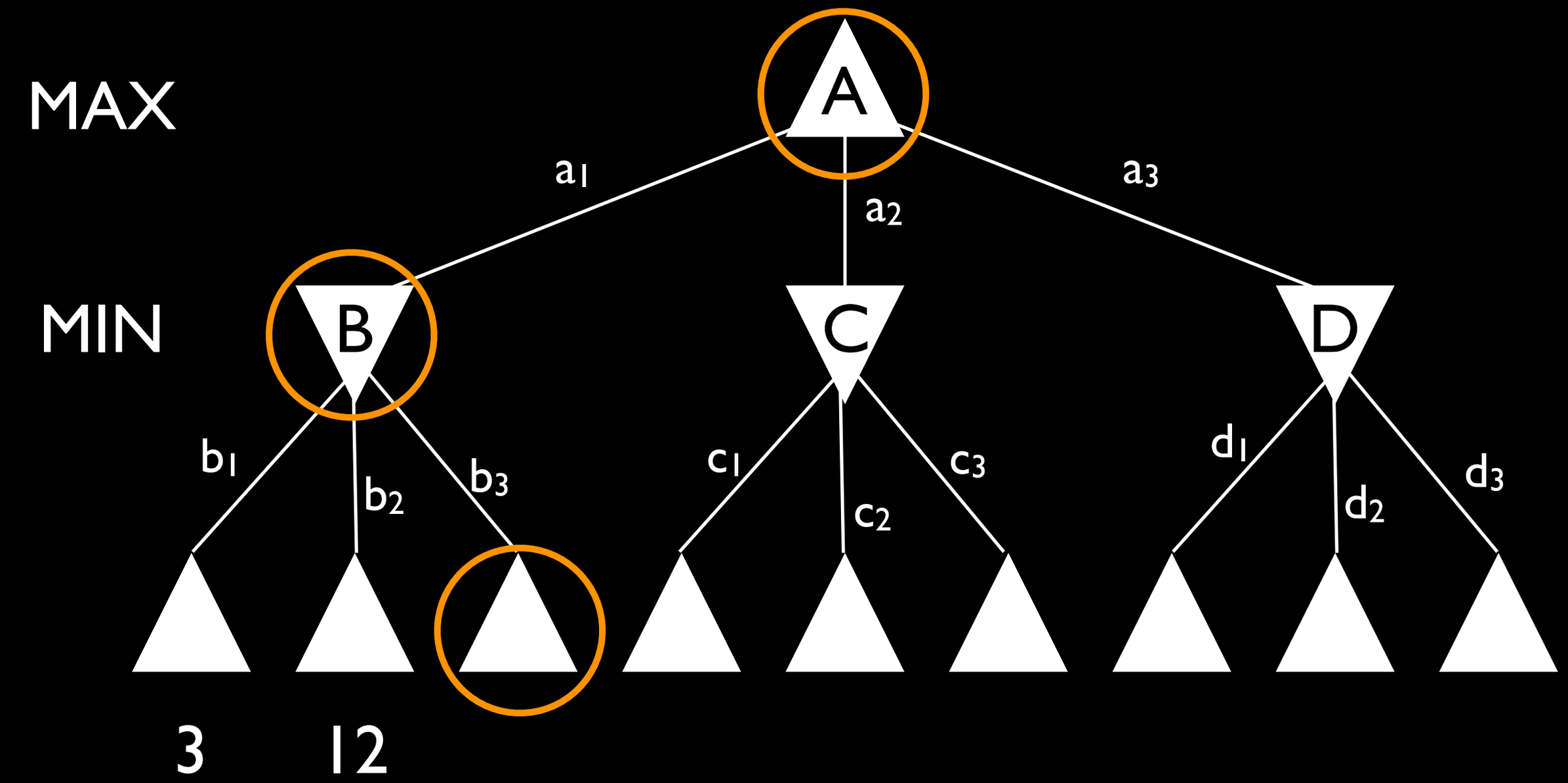
b_1 b_2 b_3

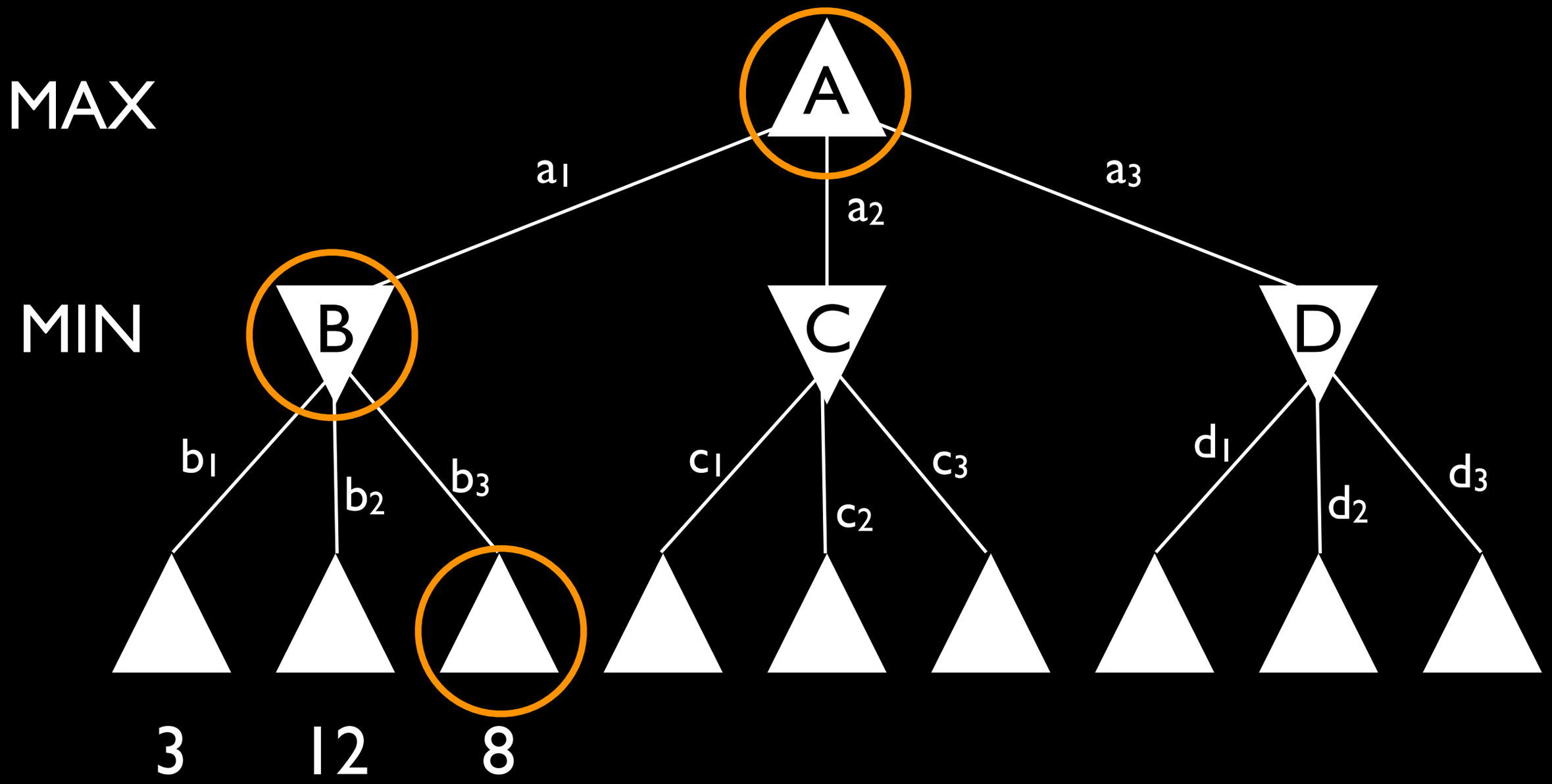
c_1 c_2 c_3

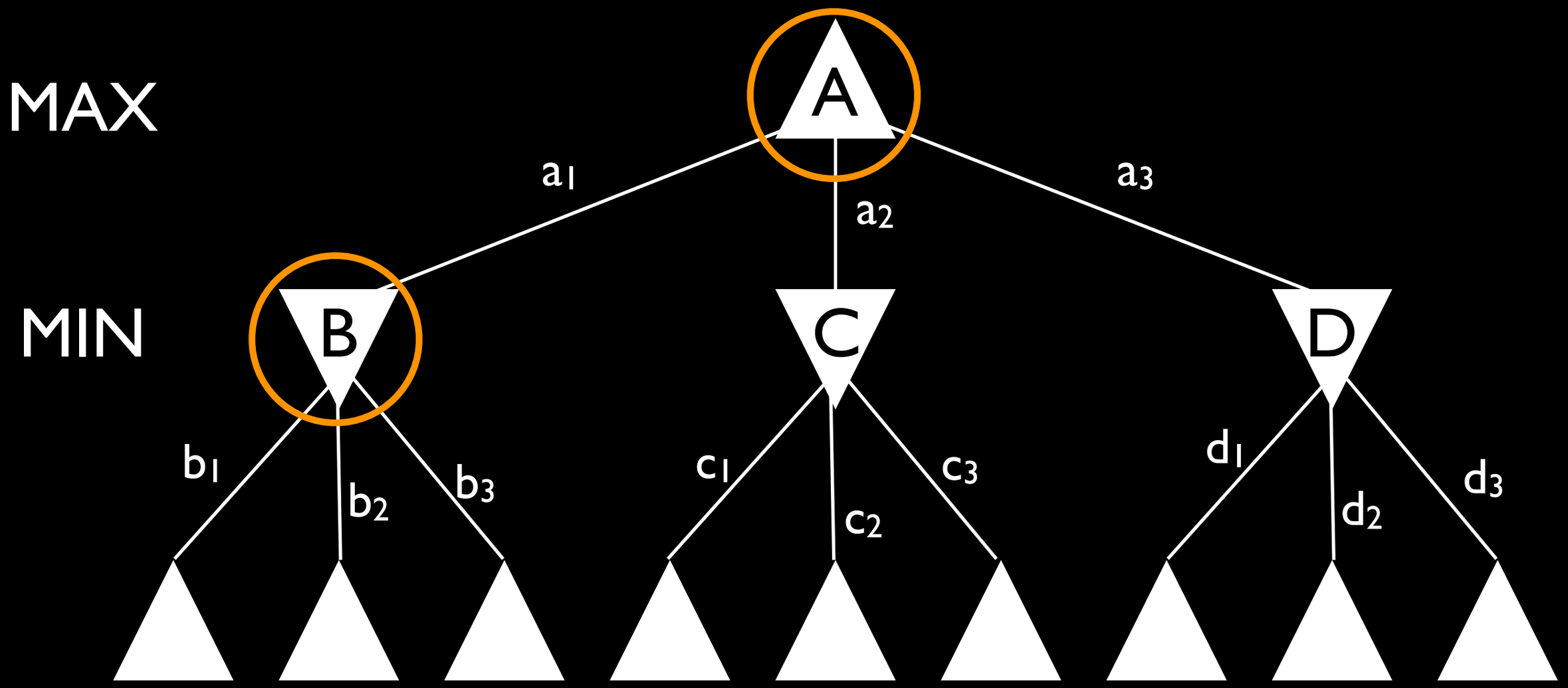
d_1 d_2 d_3

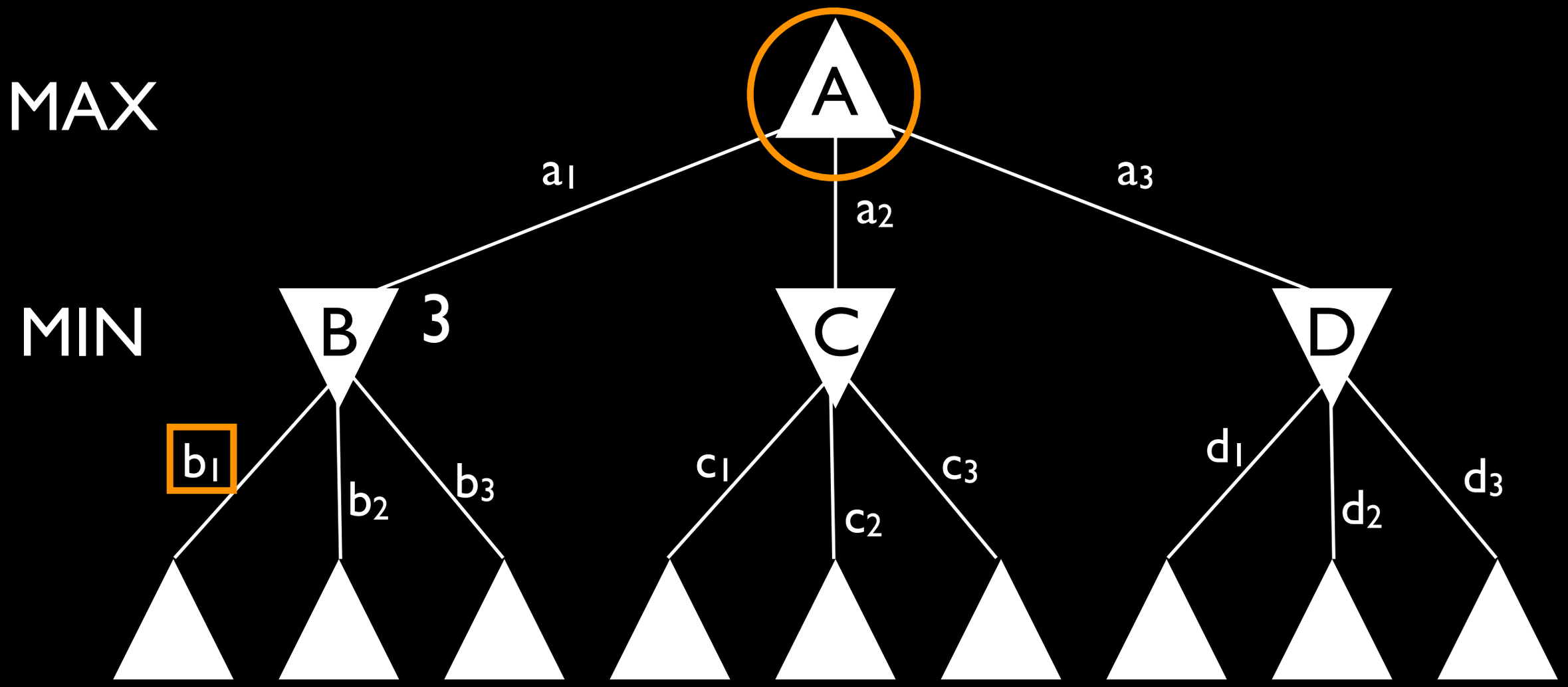

MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3

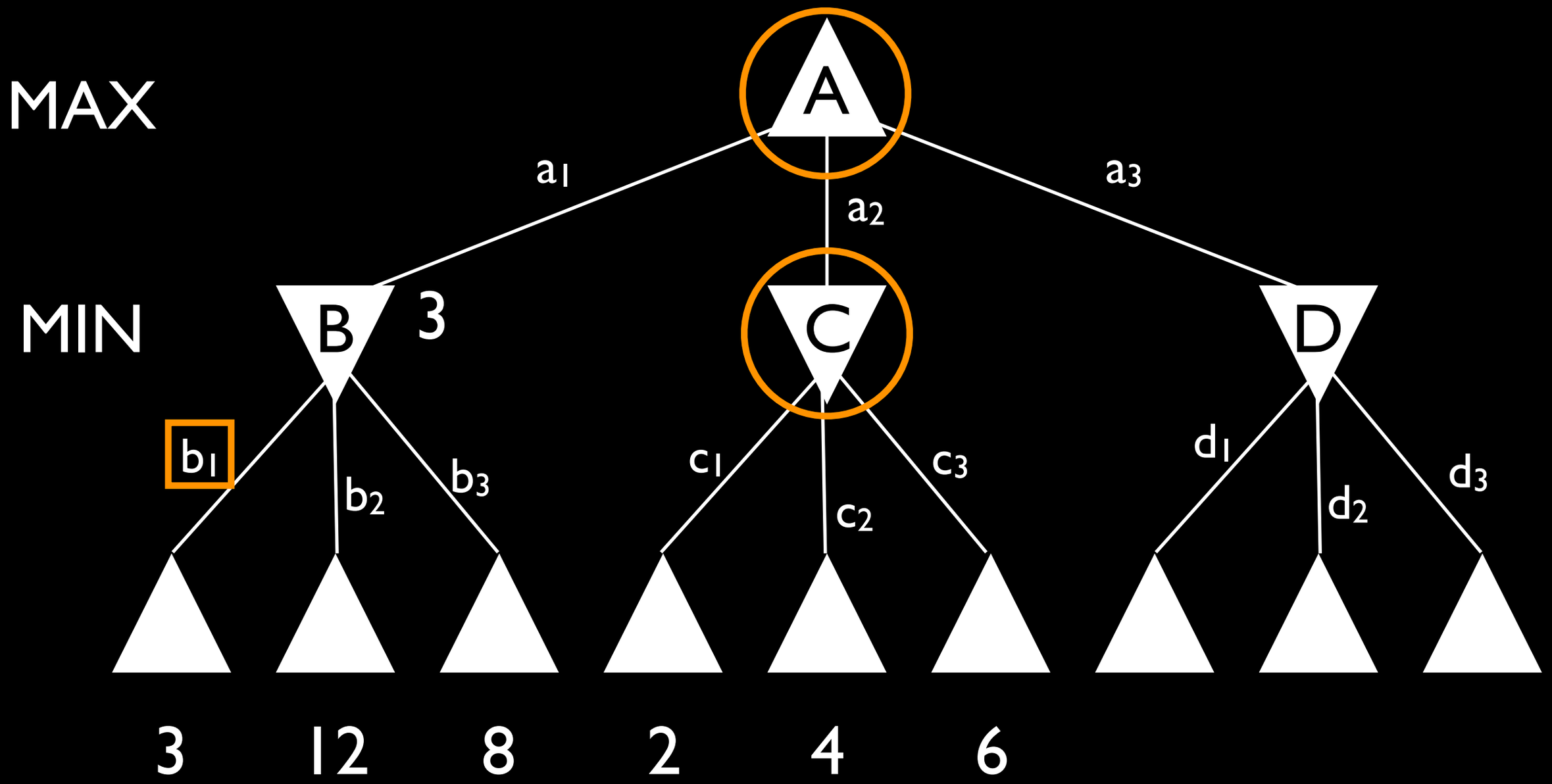

MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3

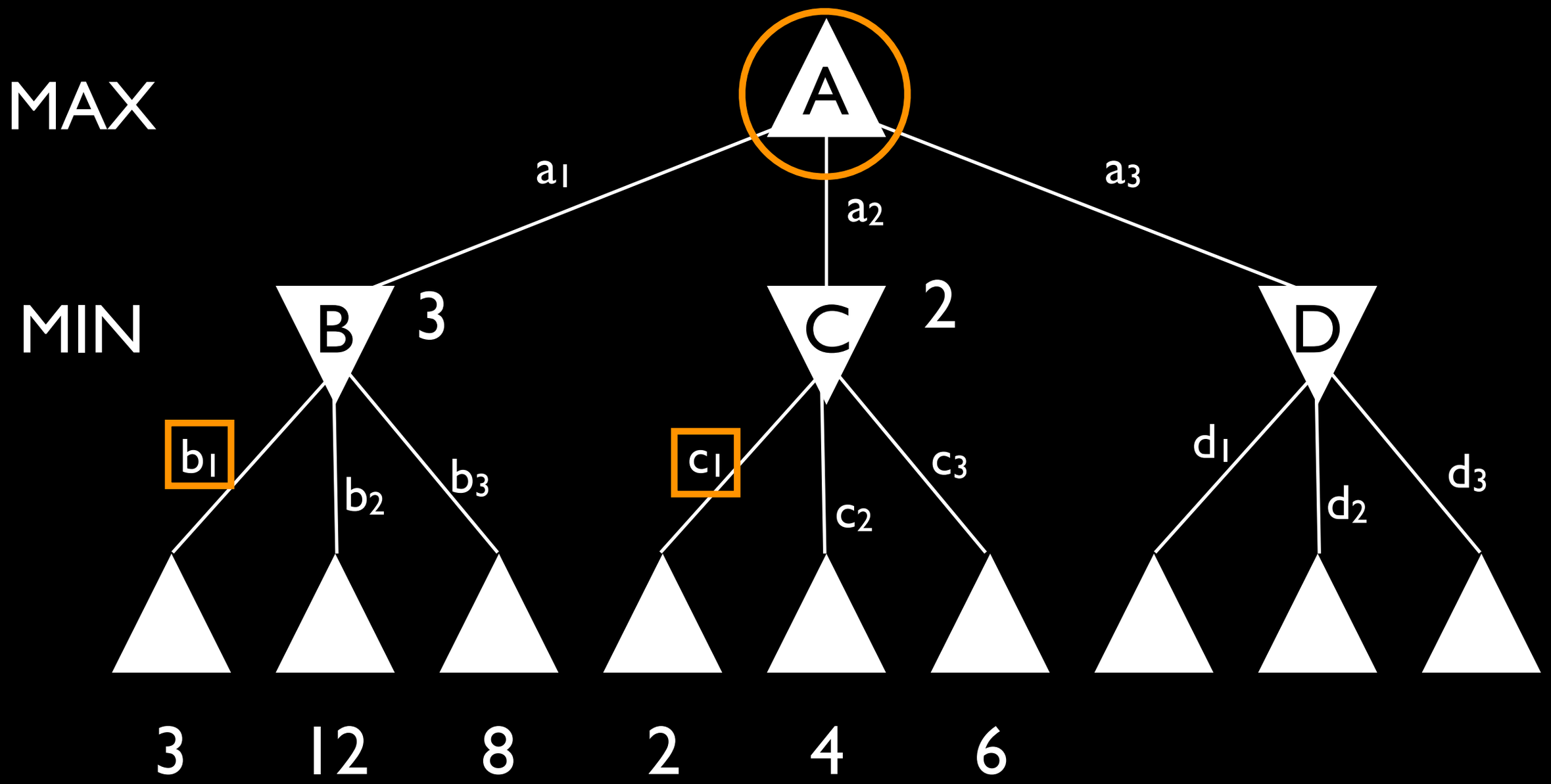

MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12

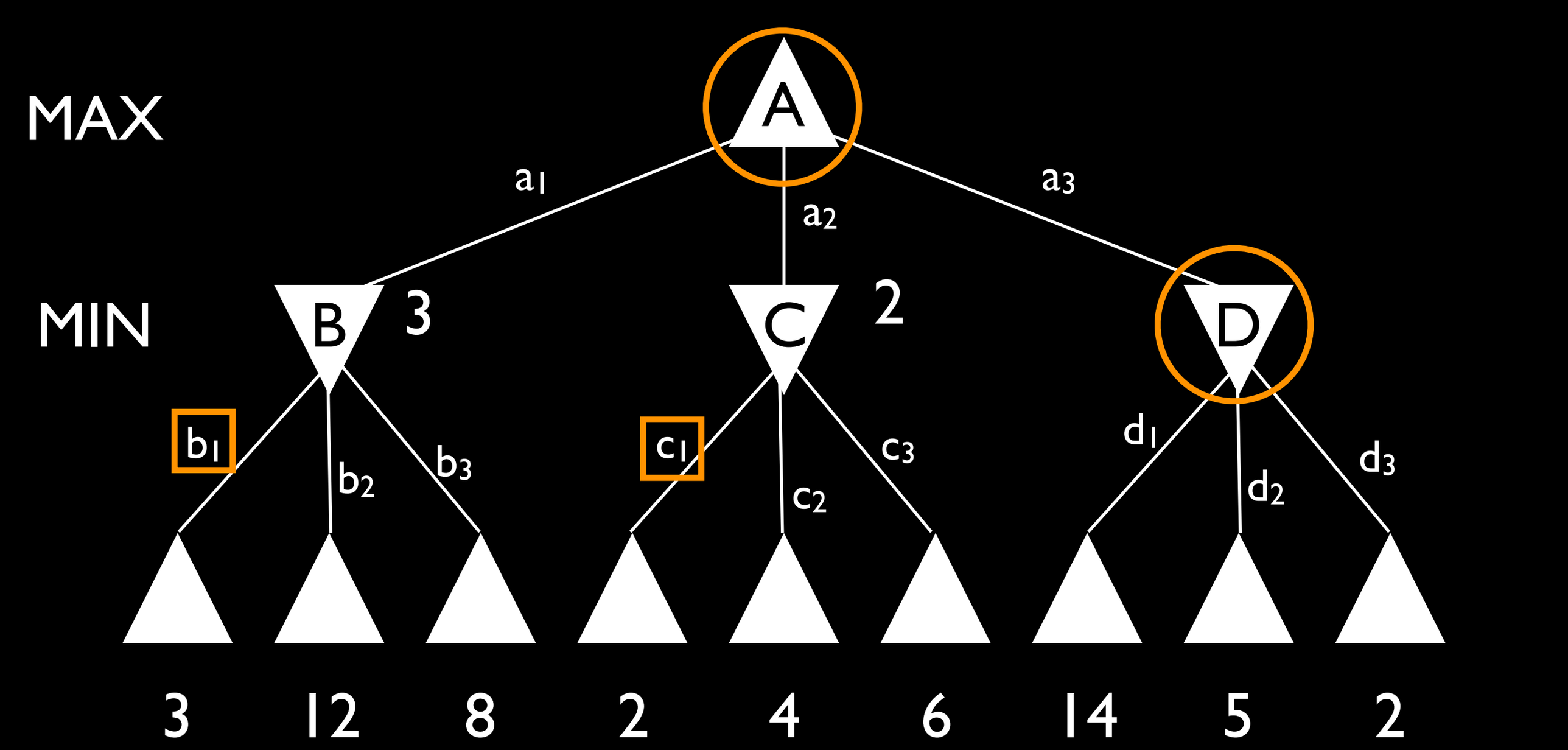

MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8

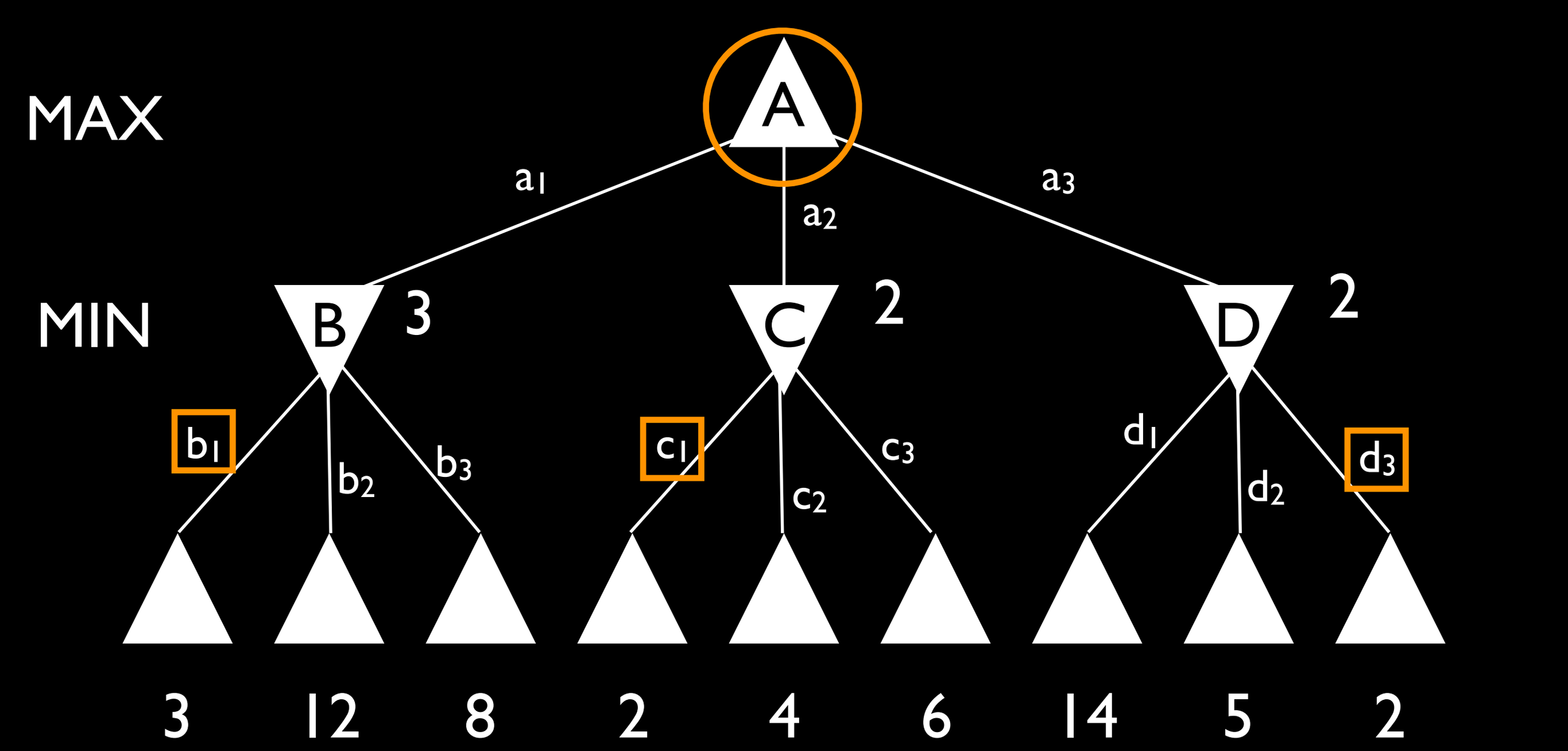

MAX
MIN
A
B
C
D
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8

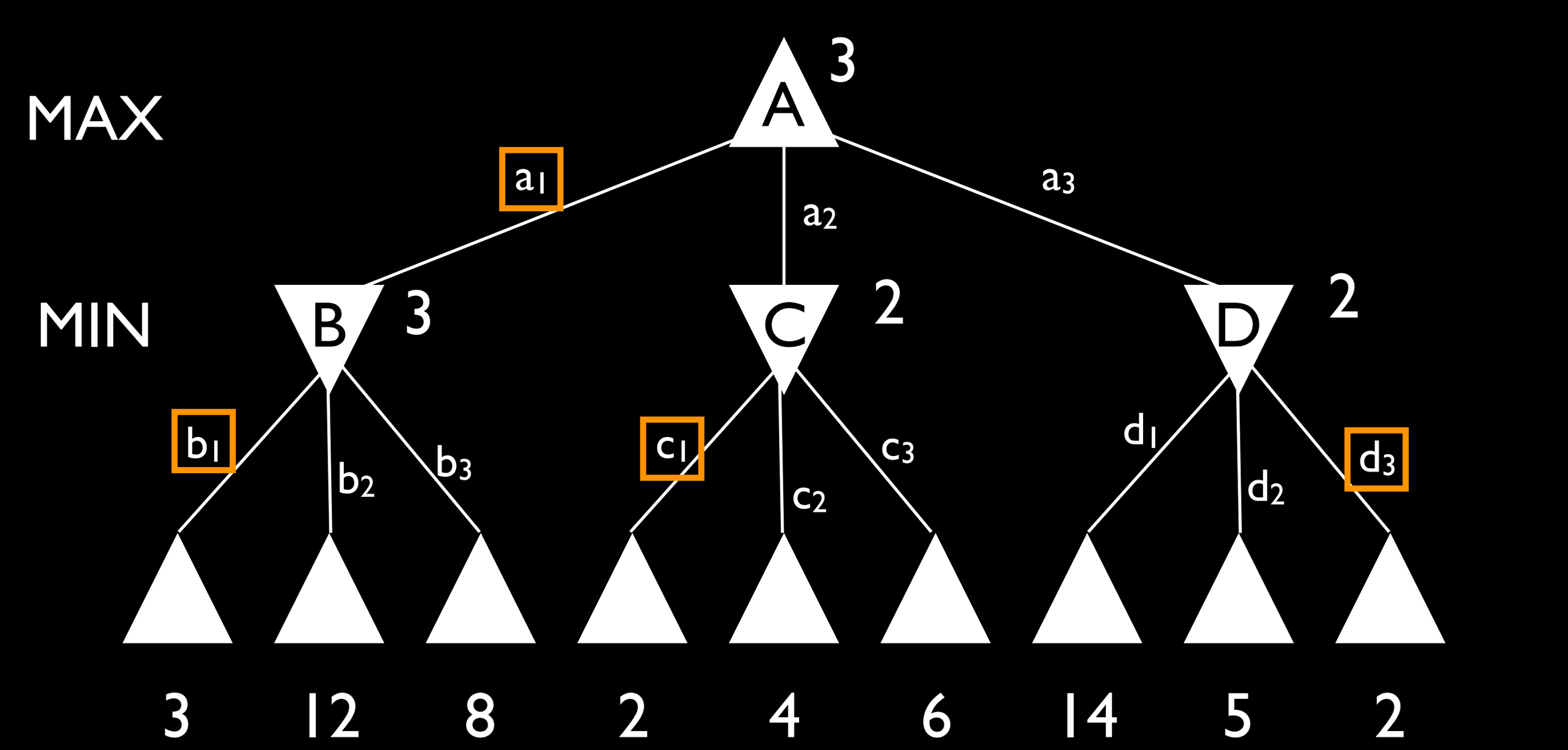

MAX
A
a1
a2
a3
MIN
B
3
C
D
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8

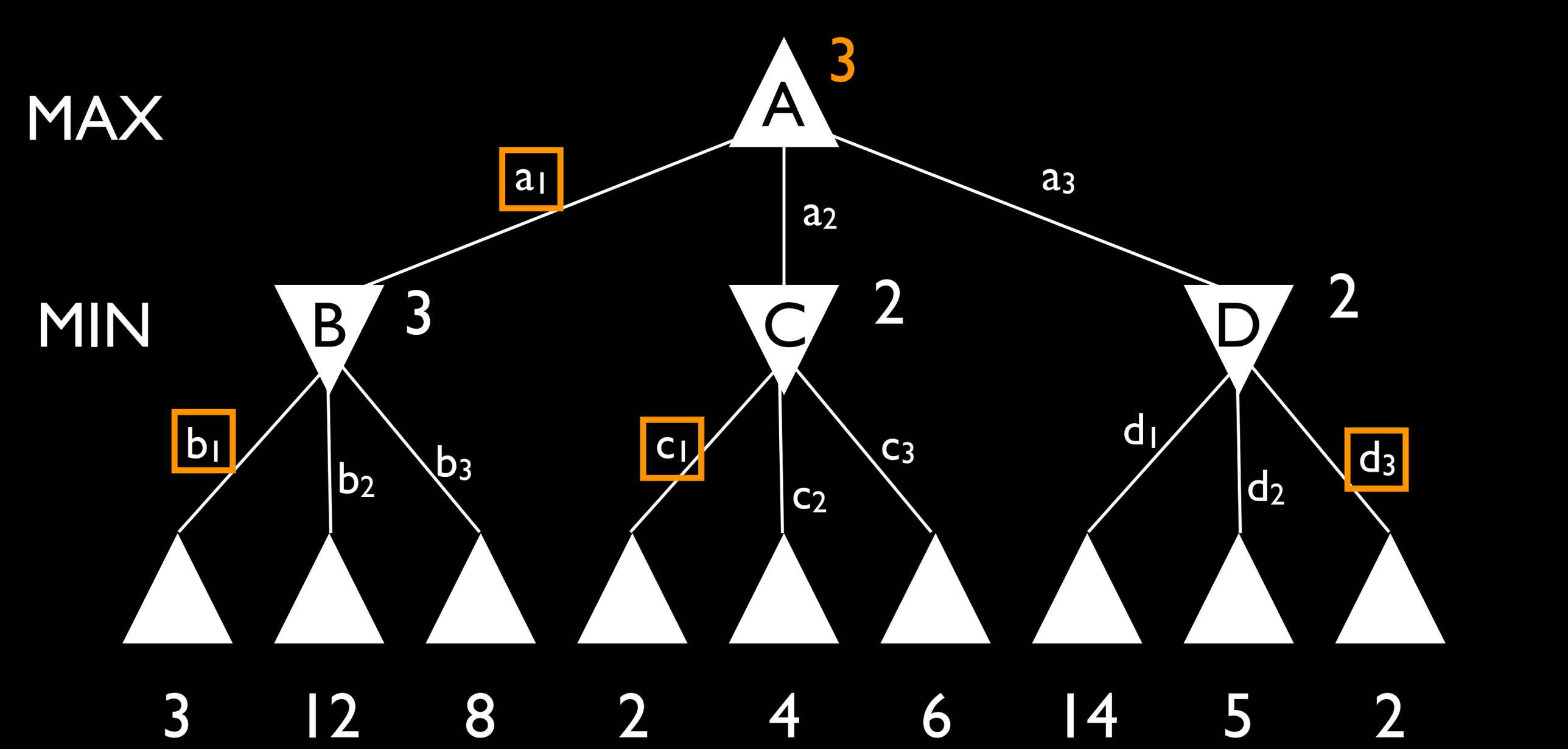

MAX
MIN
A
B
C
D
3
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6

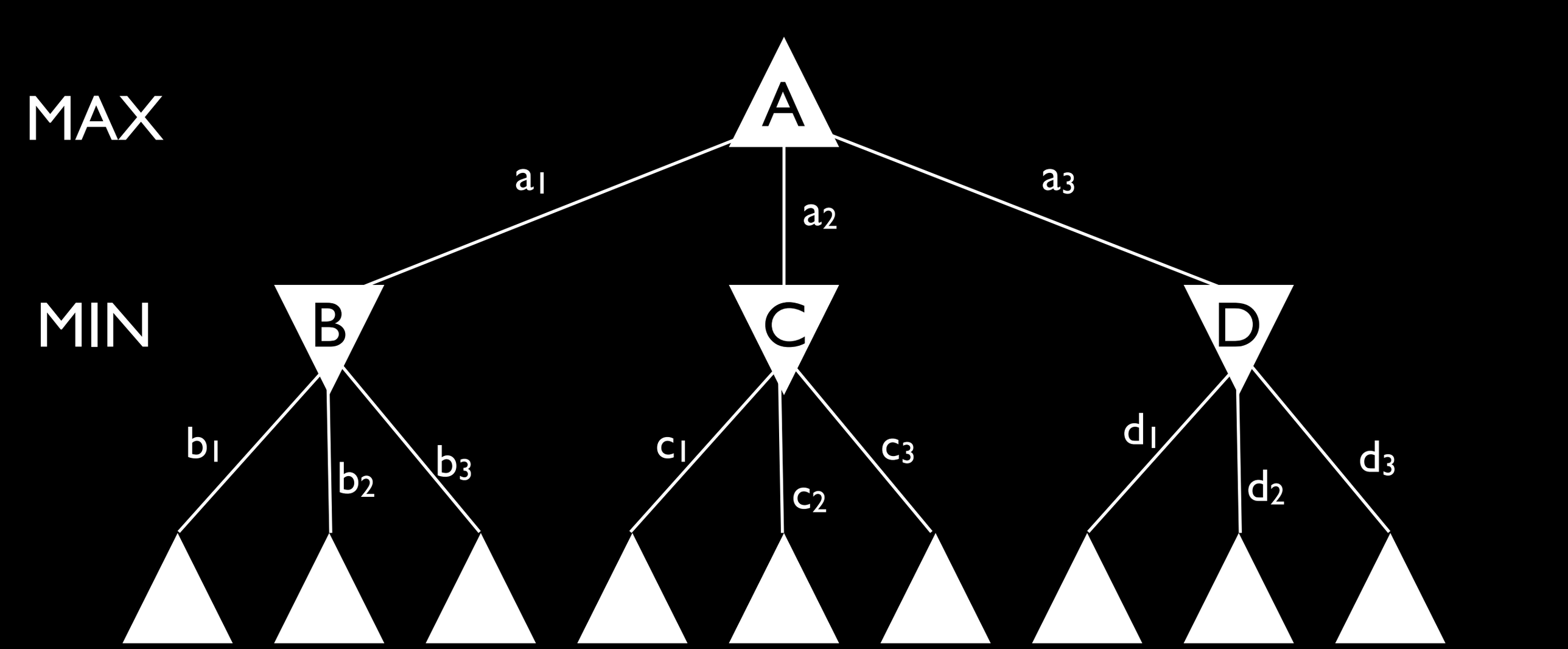

MAX
MIN
A
a1
a2
a3
B
3
C
2
D
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6

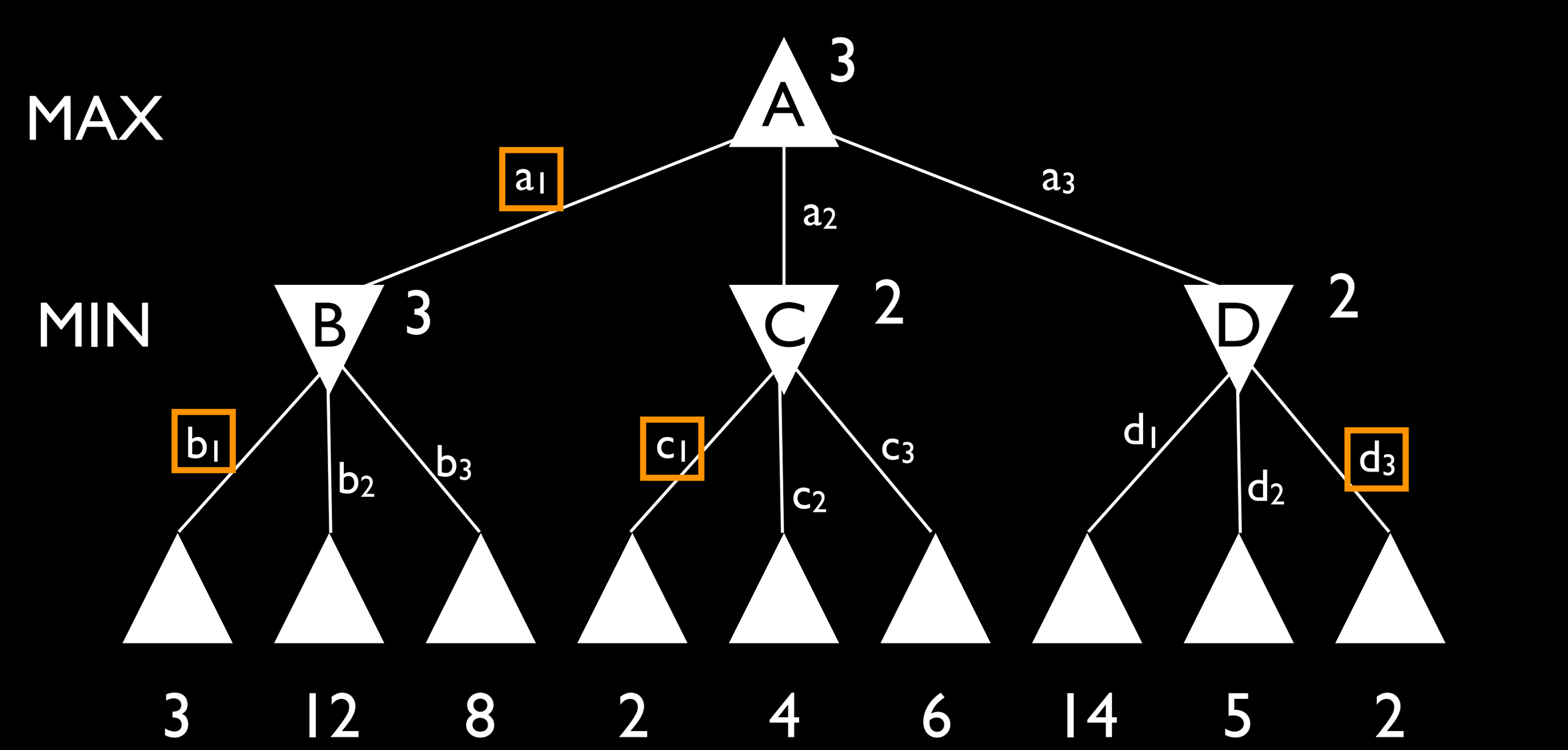

MAX
MIN
A
B
C
D
3
2
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6
14
5
2


MAX
MIN
A
a1
a2
a3
B
3
C
2
D
2
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6
14
5
2

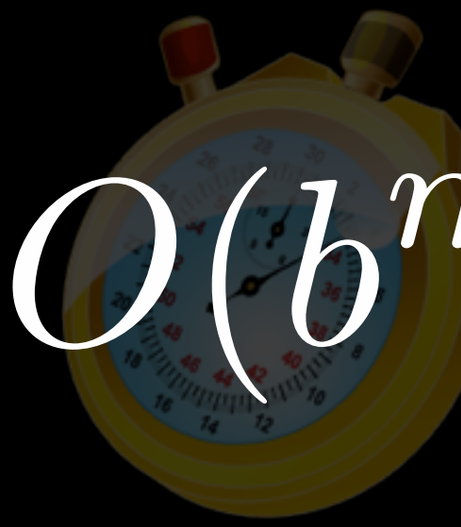

MAX
MIN
A
3
B
3
C
2
D
2
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6
14
5
2


MAX
MIN
A
3
B
3
C
2
D
2
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6
14
5
2


MAX
A
a_1
a_2
a_3
MIN
B
C
D
b_1
b_2
b_3
c_1
c_2
c_3
d_1
d_2
d_3


MAX
MIN
A
3
B
3
C
2
D
2
a1
a2
a3
b1
b2
b3
c1
c2
c3
d1
d2
d3
3
12
8
2
4
6
14
5
2

Minimax Analysis



$O(b^m)$

Time Complexity

$O(bm)$

Space Complexity

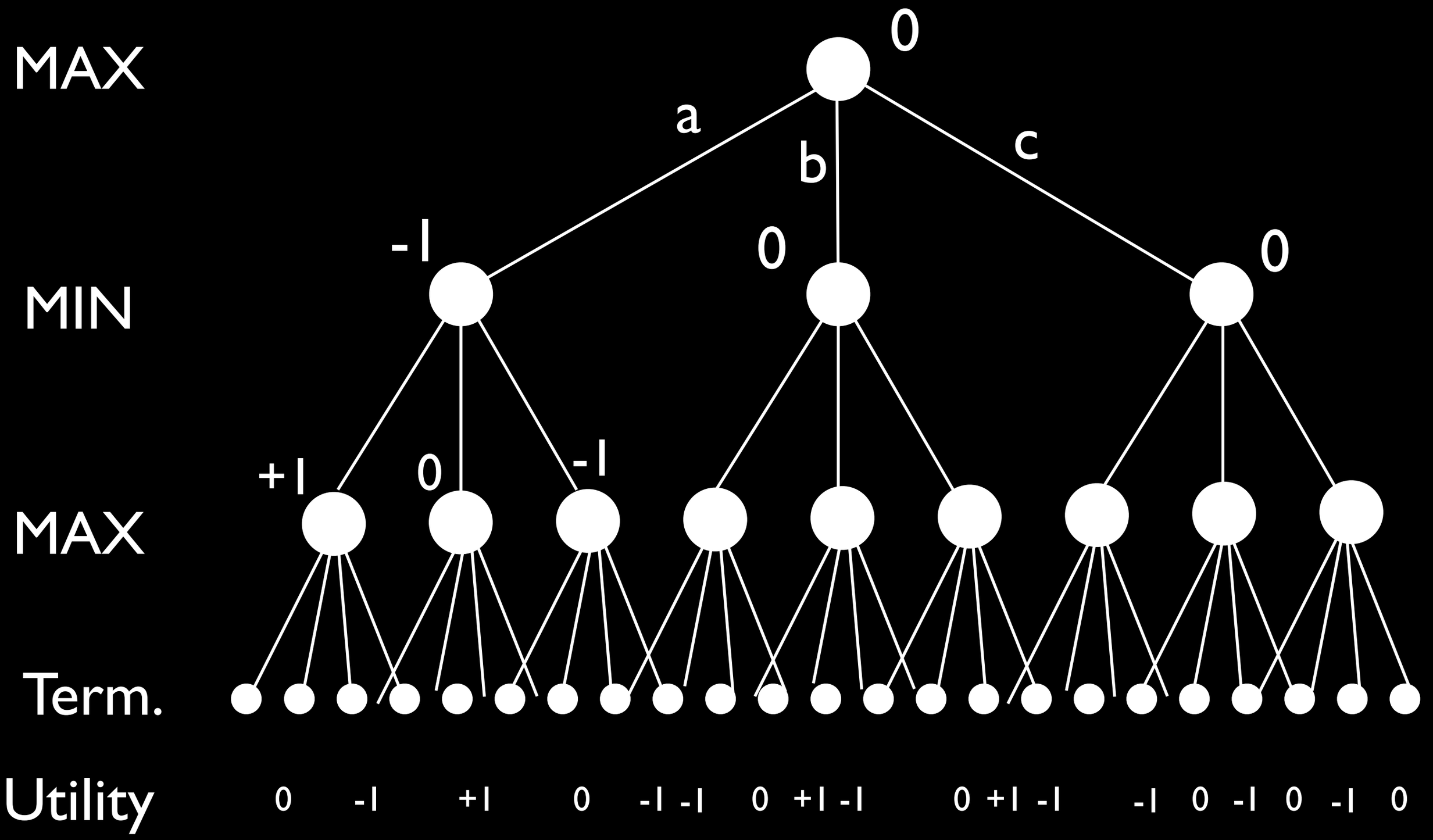
Minimax Summary

- Computes the optimal move assuming opponent also plays optimally (i.e., worst-case outcome)
- Explores game tree depth-first all the way to terminal states (end of game)
- Backs up utility values through alternating MIN and MAX (what's best for me is worst for you, and vice-versa)

Minimax Code

AIMA 5.2.1 and Figure 5.3 (page 166)

Code template will appear on course page


MAX
MIN
MAX
Term.
Utility
0
a
b
c
-1
0
0
+1
0
-1
0 -1 +1 0 -1 -1 0 +1 -1 0 +1 -1 -1 0 -1 0 -1 0

A Problem for Minimax

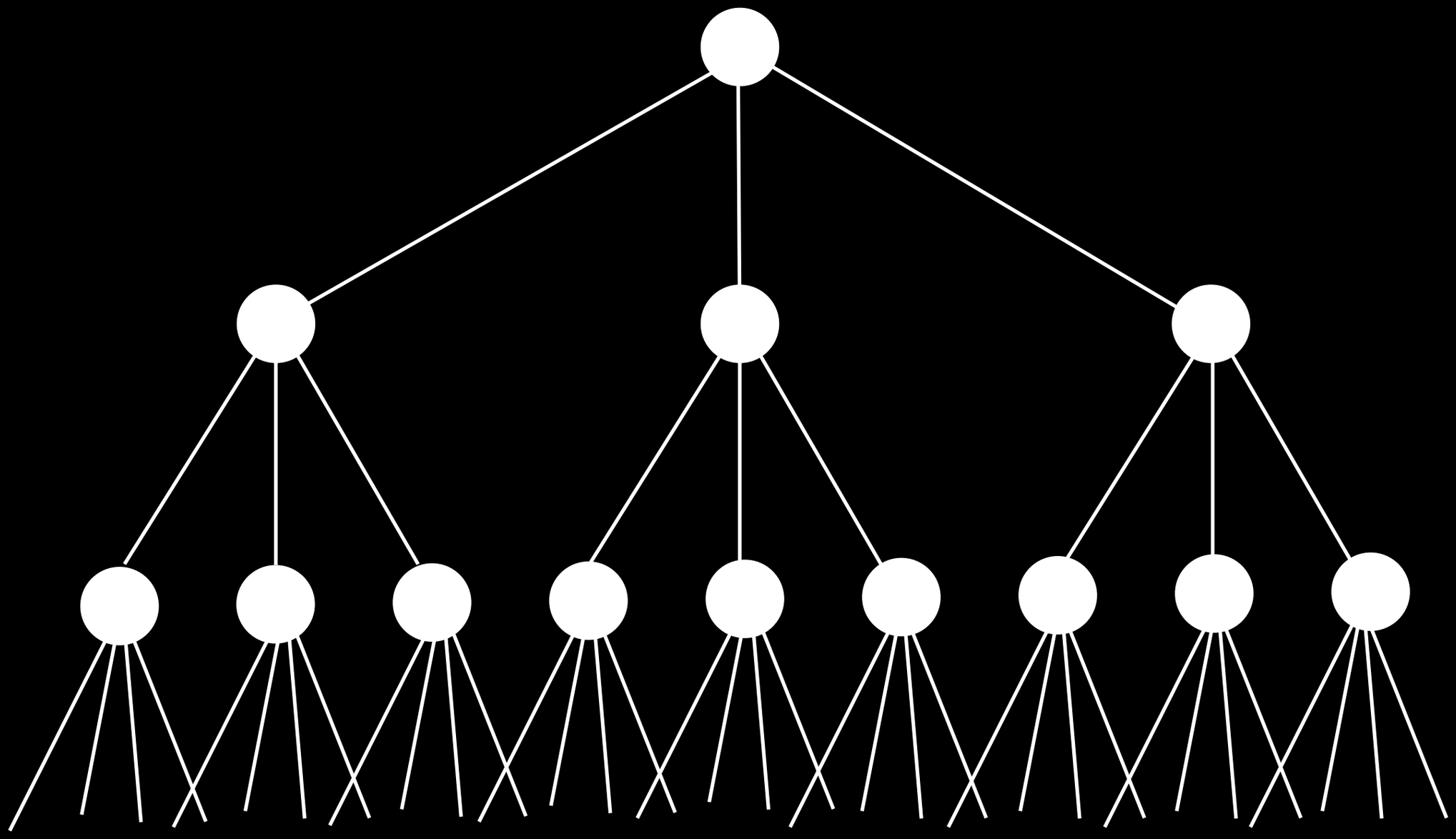
- You can't search all the way to the terminal nodes
- You can't evaluate a node (state) unless you're at a terminal node
- Utility function is defined on terminal states

Imperfect Real-Time Decisions

MAX

MIN

MAX

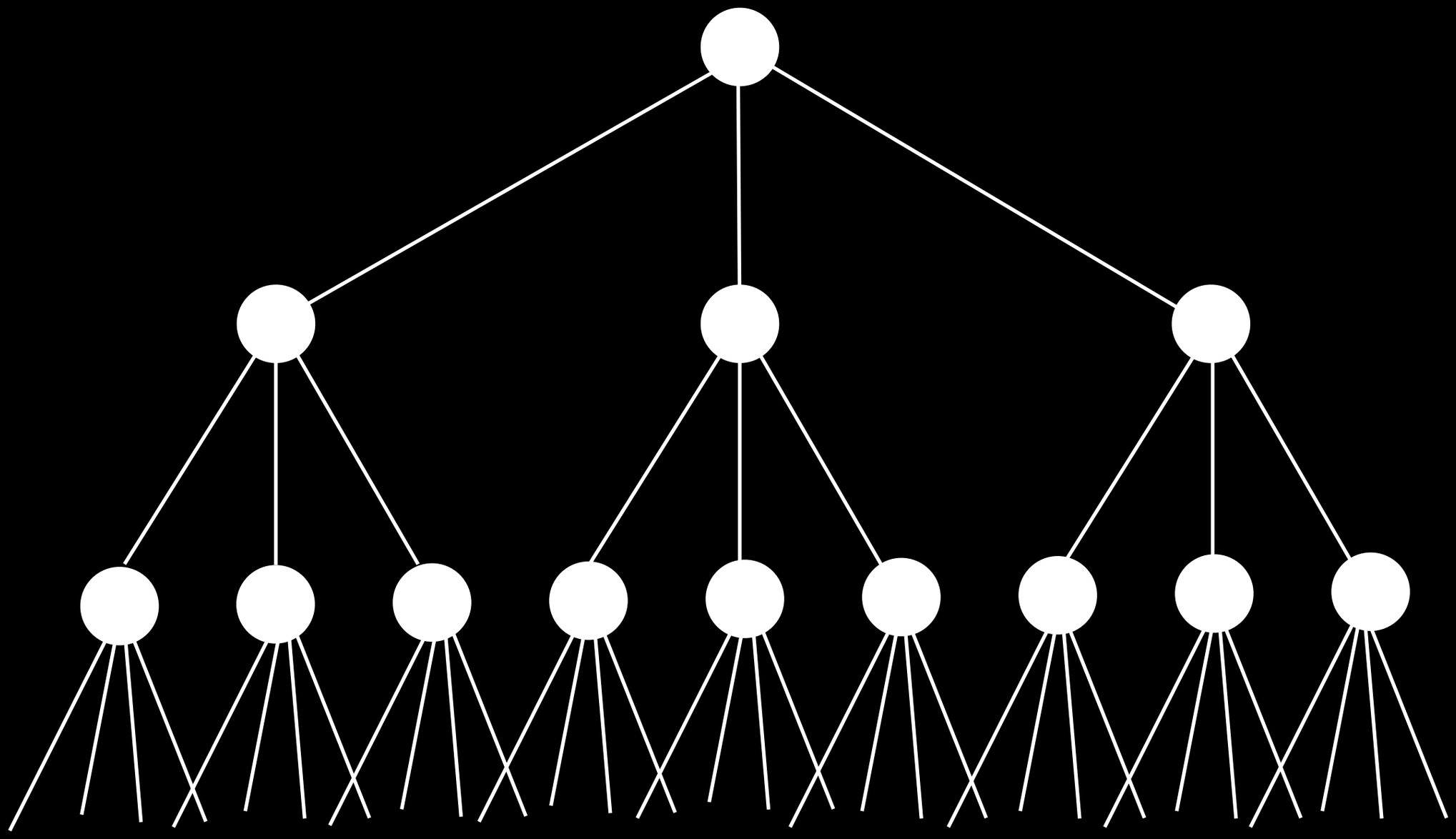


Term.

MAX

MIN

MAX

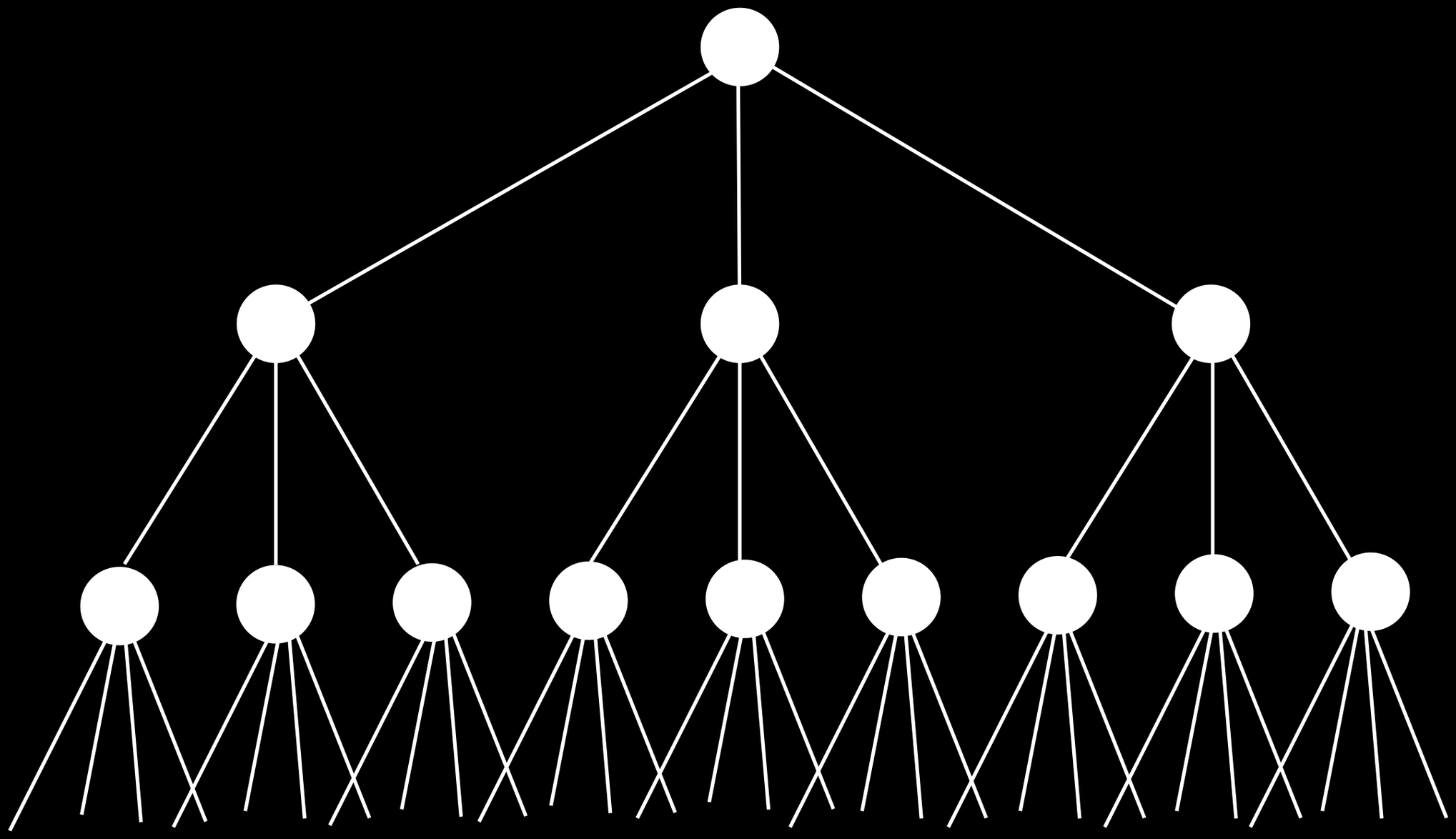


Term.

MAX

MIN

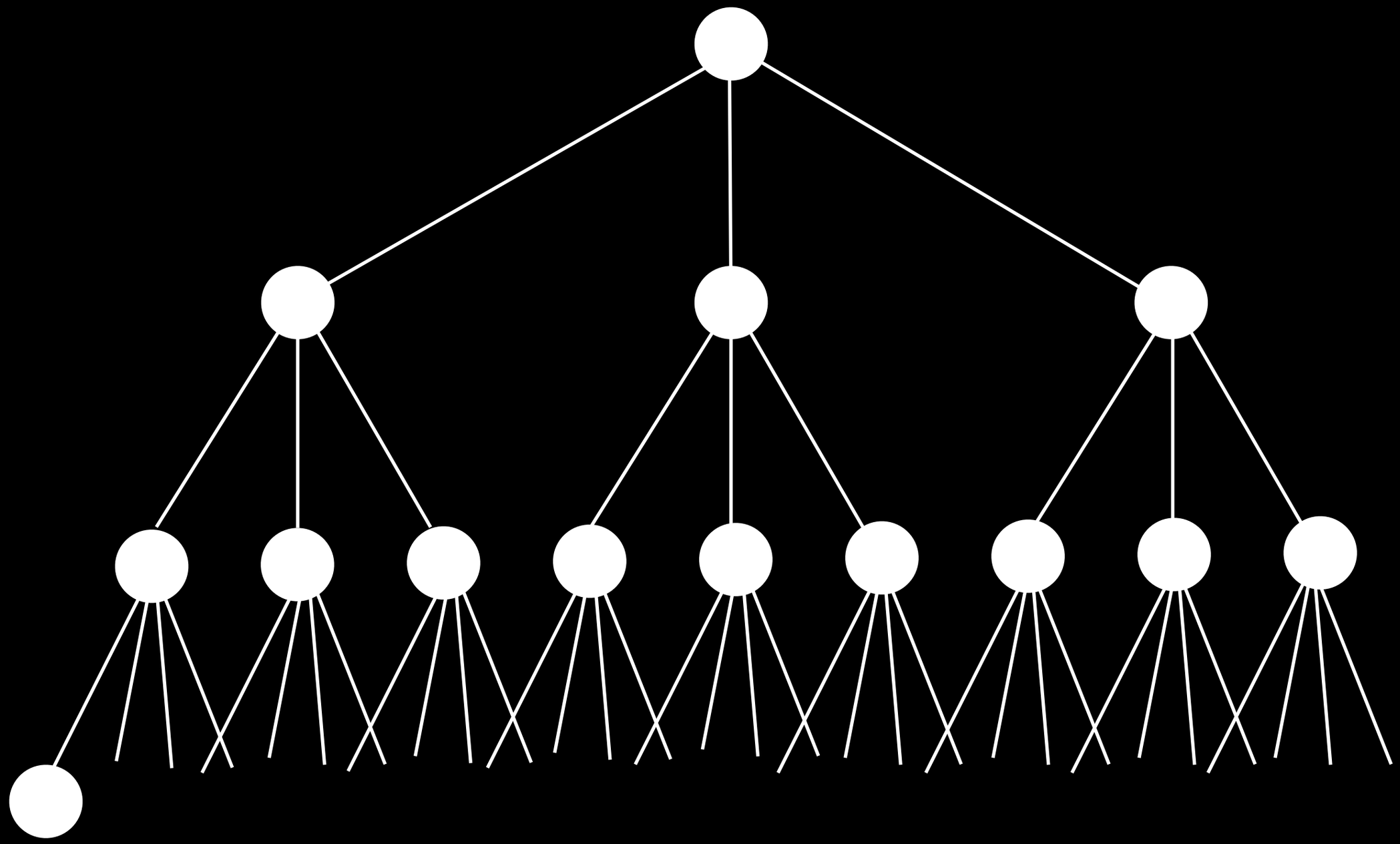
MAX



MAX

MIN

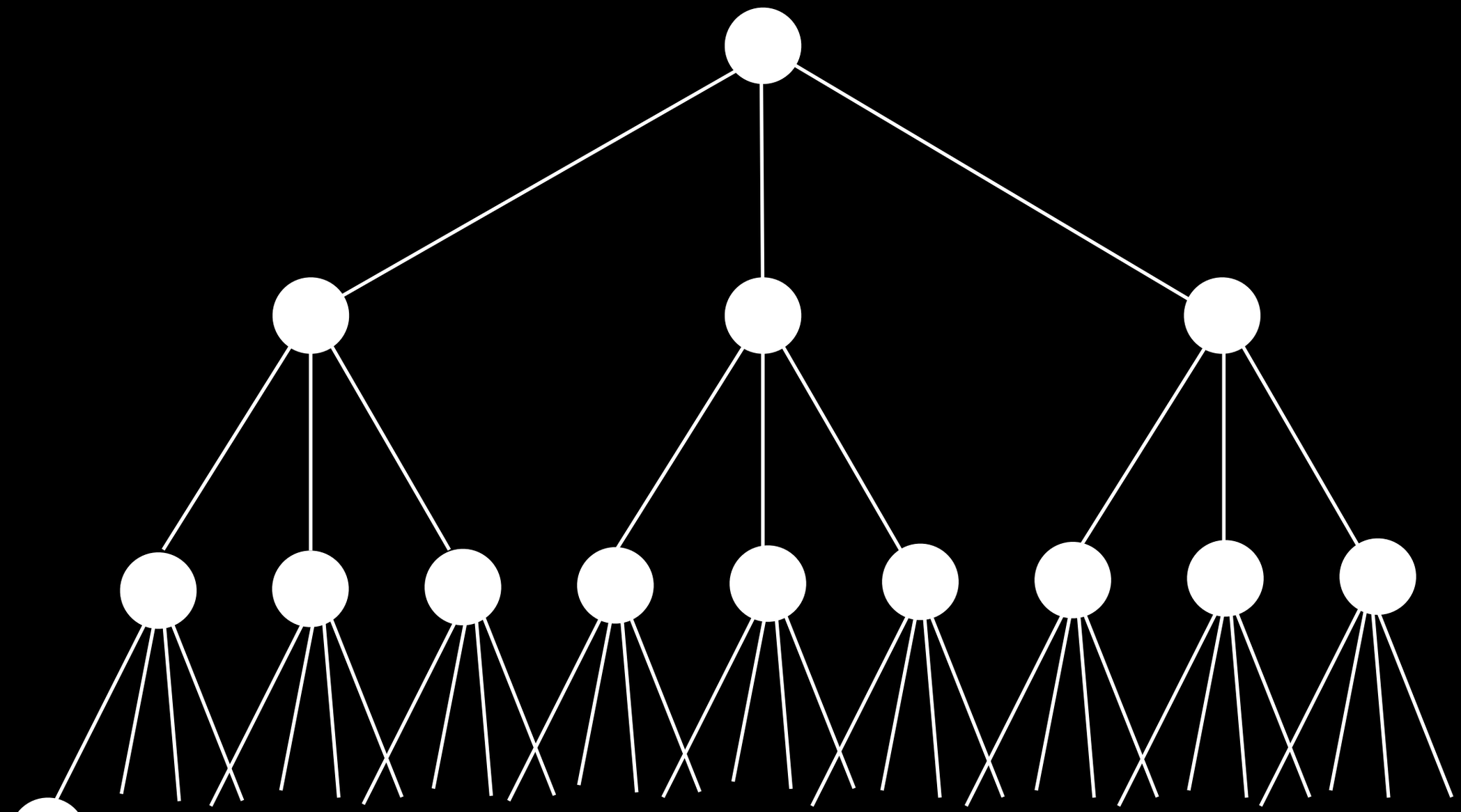
MAX



MAX

MIN

MAX



$h(n)$

Minimax Algorithm

$$\textsc{Minimax}(s) = \begin{cases} \textsc{Utility}(s) & \text{if } \textsc{Terminal-Test}(s) \\ \max_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{if } \textsc{Player}(s) = \textsc{max} \\ \min_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{if } \textsc{Player}(s) = \textsc{min} \end{cases}$$

Heuristic Minimax

$$\text{H-MINIMAX}(s) = \begin{cases} h(s) & \text{if CUTOFF-TEST}(s) \\ \max_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MAX} \\ \min_{a \in \text{ACTIONS}(s)} \text{MINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MIN} \end{cases}$$

H-Minimax: Cutoff

- When to cutoff search?
  - Time
  - Depth
  - “Quiescence”
- Can adjust dynamically
- AIMA 5.4.2

H-Minimax: Heuristic

- How to evaluate non-terminal state?
  - AIMA 5.4.1
  - Chess example
    - Material value
    - Weighted sum
    - Strategic considerations

Heuristic Minimax

- Cutoff search before reaching terminal nodes (time, depth, "quiescence")
- Use heuristic evaluation function to estimate state utility
- Backs up utility values through alternating MIN and MAX (what's best for me is worst for you, and vice-versa)

Adversarial Search: MINIMAX

MINIMAX

- Searches to terminal nodes
- Uses utility function

H-MINIMAX

- Cuts off search before terminals
- Uses heuristic function

Backs up utility values through alternating MIN and MAX (zero-sum game)

For Next Class

- Read rest of AIMA Chapter 5
- Complete Homework 01 - solutions will be given out
- Project 1 launch: Othello Tournament
- Lecture 06: Games of chance and partial information