

# Curs 8

Jocuri

# Regulile de joc

- Doi jucători: MAX și MIN
- Fiecare are ca obiectiv câștigarea jocului
- Doar unul poate câștiga sau se poate obține remiză
- În modelarea inițială nu intervine șansa
  - dar ea poate fi simulată
- Exemple:
  - șah
  - checkers
  - tic-tac-toe
  - ...

# Jocul tic-tac-toe

MAX joacă cu X-uri

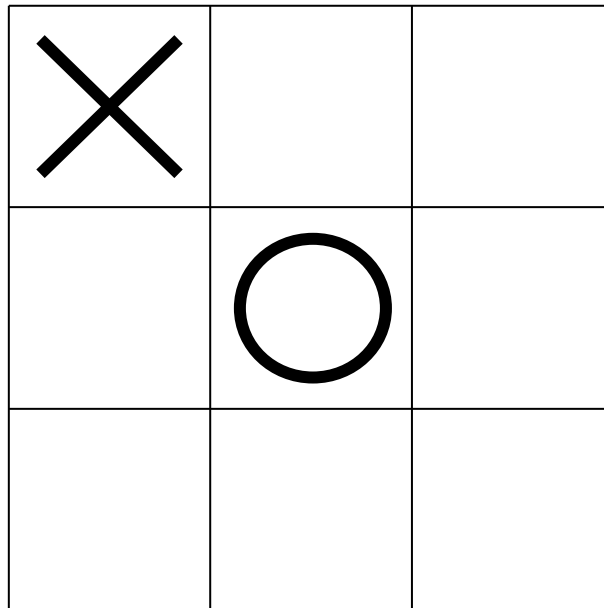

MIN joacă cu O-uri

# Jocul tic-tac-toe

MAX

X		

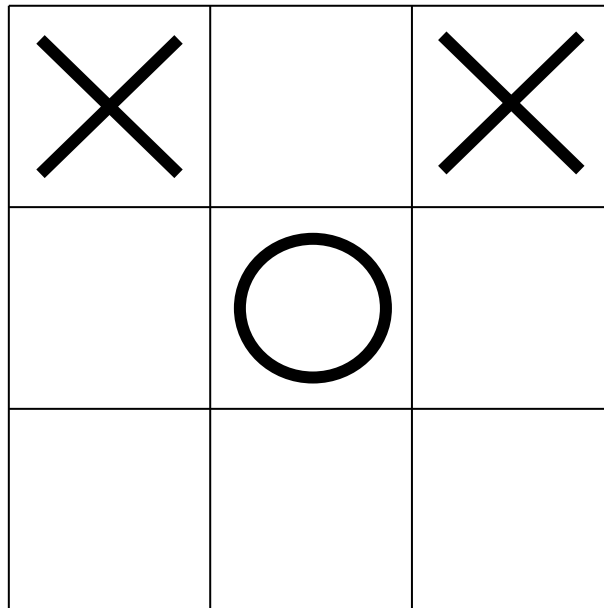
# Jocul tic-tac-toe



MIN

# Jocul tic-tac-toe

MAX



# Jocul tic-tac-toe

X	O	X
	O	

MIN

# Jocul tic-tac-toe

MAX

X	O	X
	O	
	X	



# Jocul tic-tac-toe

X	O	X
	O	O
	X	

MIN

# Jocul tic-tac-toe

MAX

X	O	X
X	O	O
	X	

# Jocul tic-tac-toe

X	O	X
X	O	O
O	X	

MIN

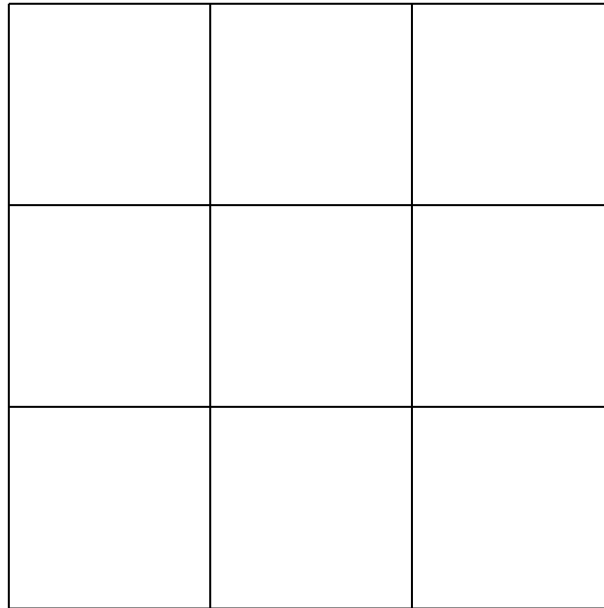
# Jocul tic-tac-toe

MAX

X	O	X
X	O	O
O	X	X

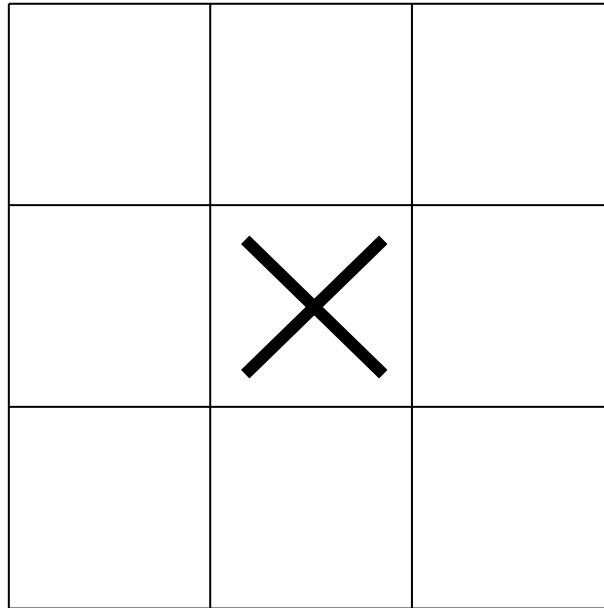
Remiză!

# Jocul tic-tac-toe



# Jocul tic-tac-toe

MAX



# Jocul tic-tac-toe

	○	
	×	

MIN

# Jocul tic-tac-toe

MAX

	○	
	×	
×		



# Jocul tic-tac-toe

	○	○
	×	
×		

MIN

# Jocul tic-tac-toe

MAX

X	O	O
	X	
X		

# Jocul tic-tac-toe

X	O	O
	X	
X		O

MIN

# Jocul tic-tac-toe

MAX

X	O	O
X	X	
X		O

# Jocul tic-tac-toe

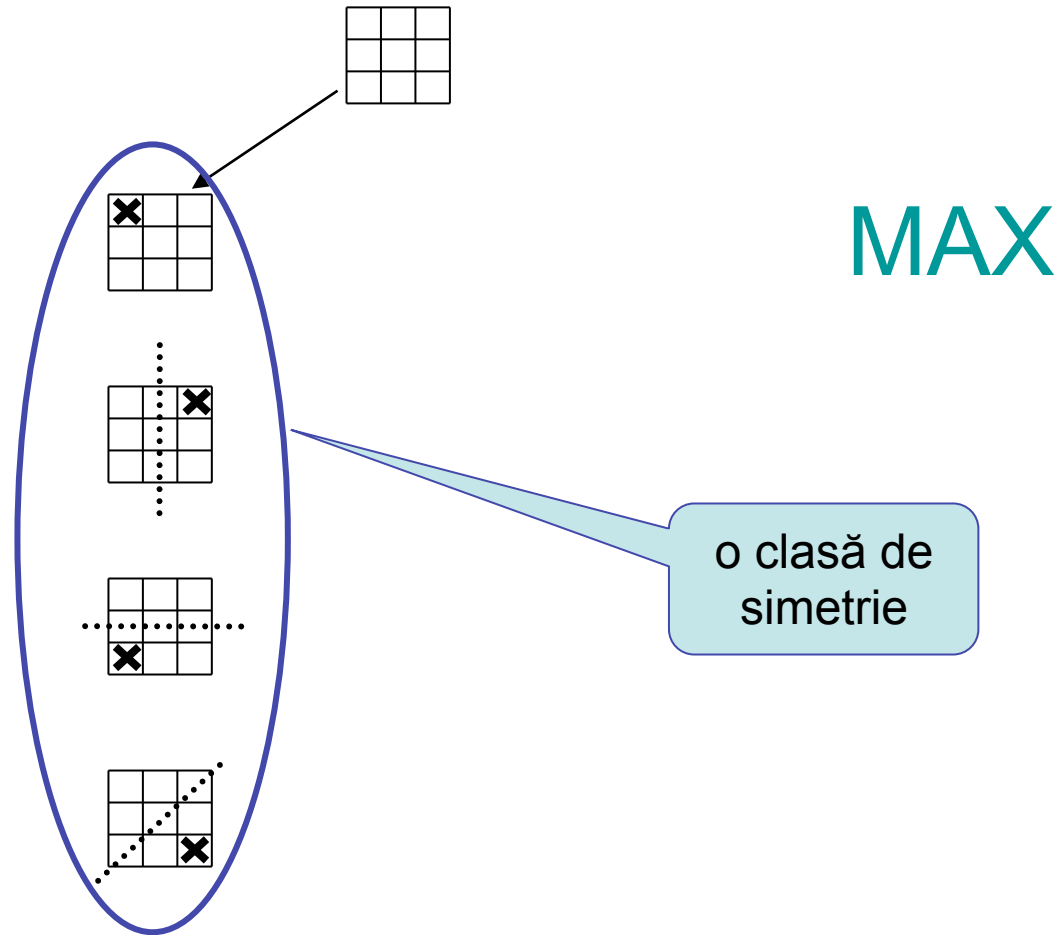
MAX  
câștigă

X	O	O
X	X	
X		O

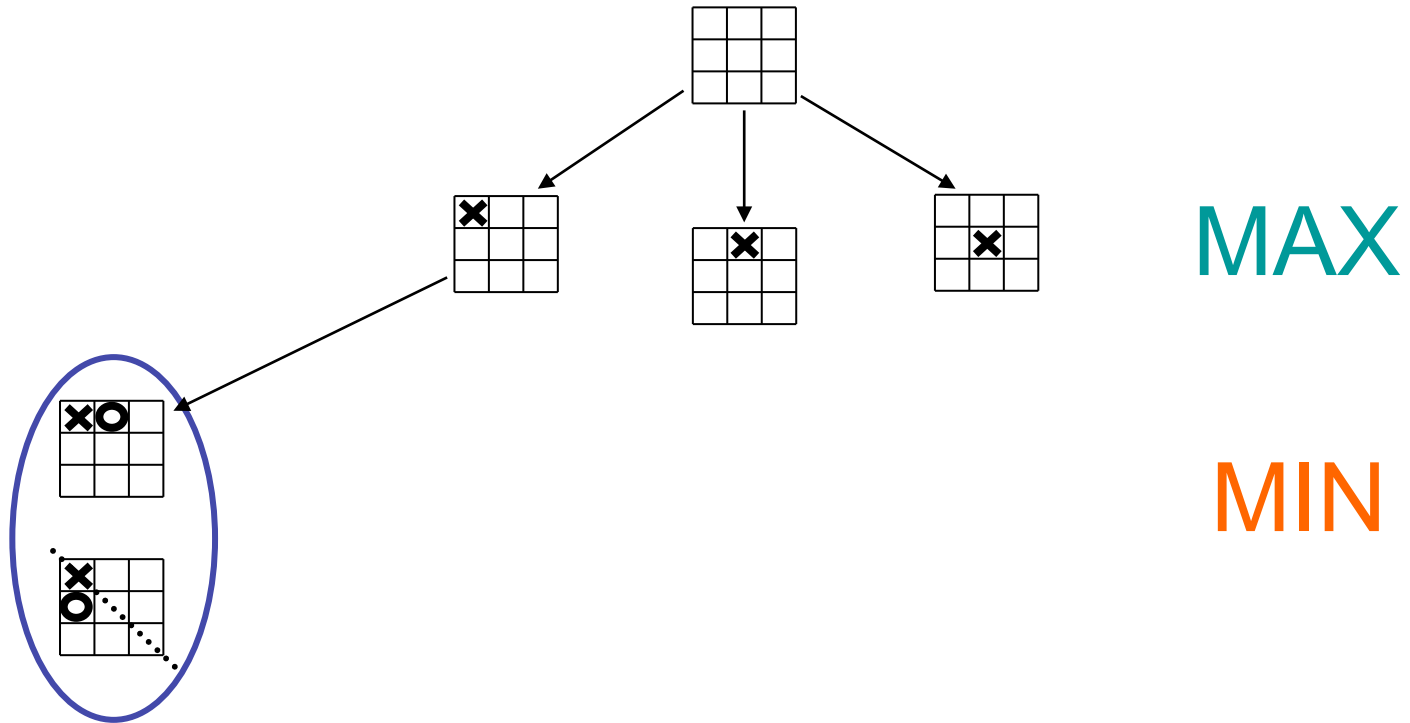
# Reprezentarea ca o problemă de IA

1. Problemă versus instanță
2. Spațiul stărilor:
  - o stare: poziția pe tabla a semnelor între două mutări
  - dimensiunea spațiului:  $3^9$
3. Reprezentarea unei stări:
  - o matrice 3x3
4. Reprezentarea unei tranziții
  - algoritmic (în abordarea de față)
5. Cum controlăm evoluția jocului?
  - metoda MIN-MAX
  - metoda ALPHA-BETA

# Arborele de joc

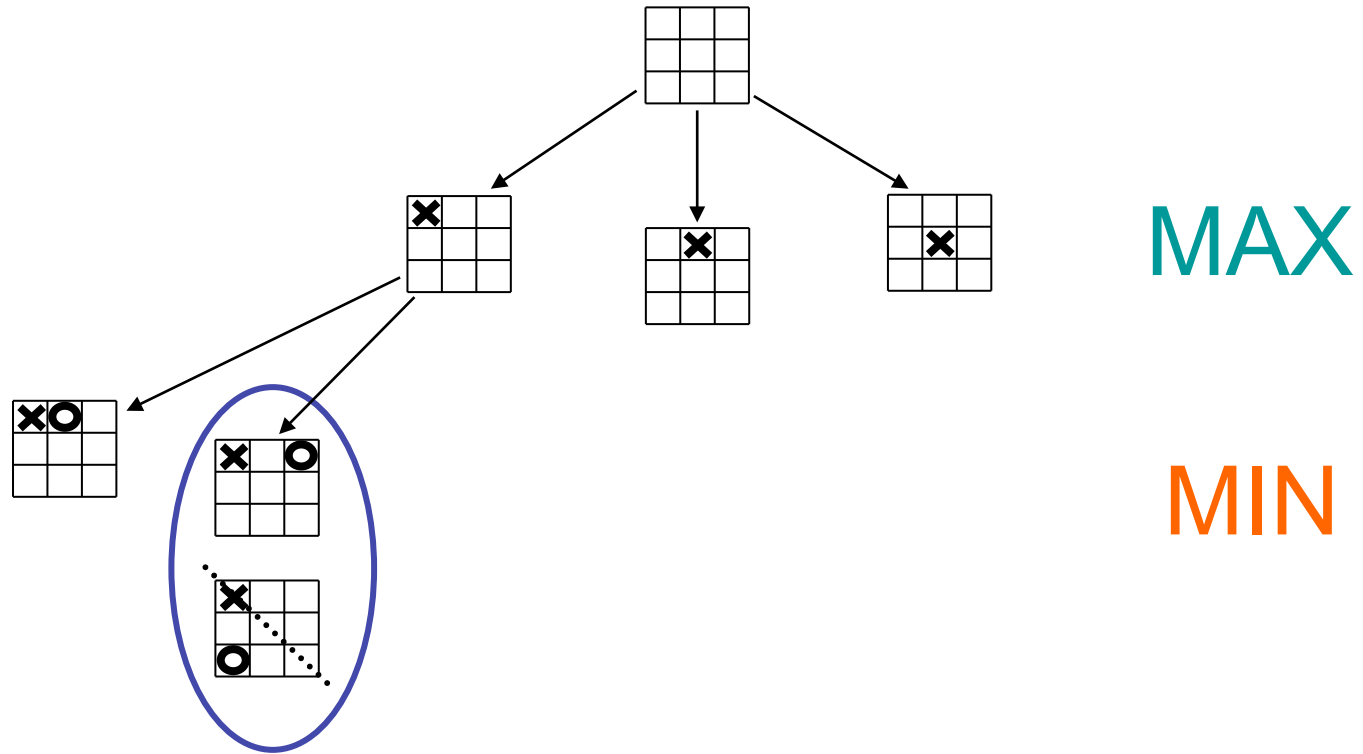


# Arborele de joc

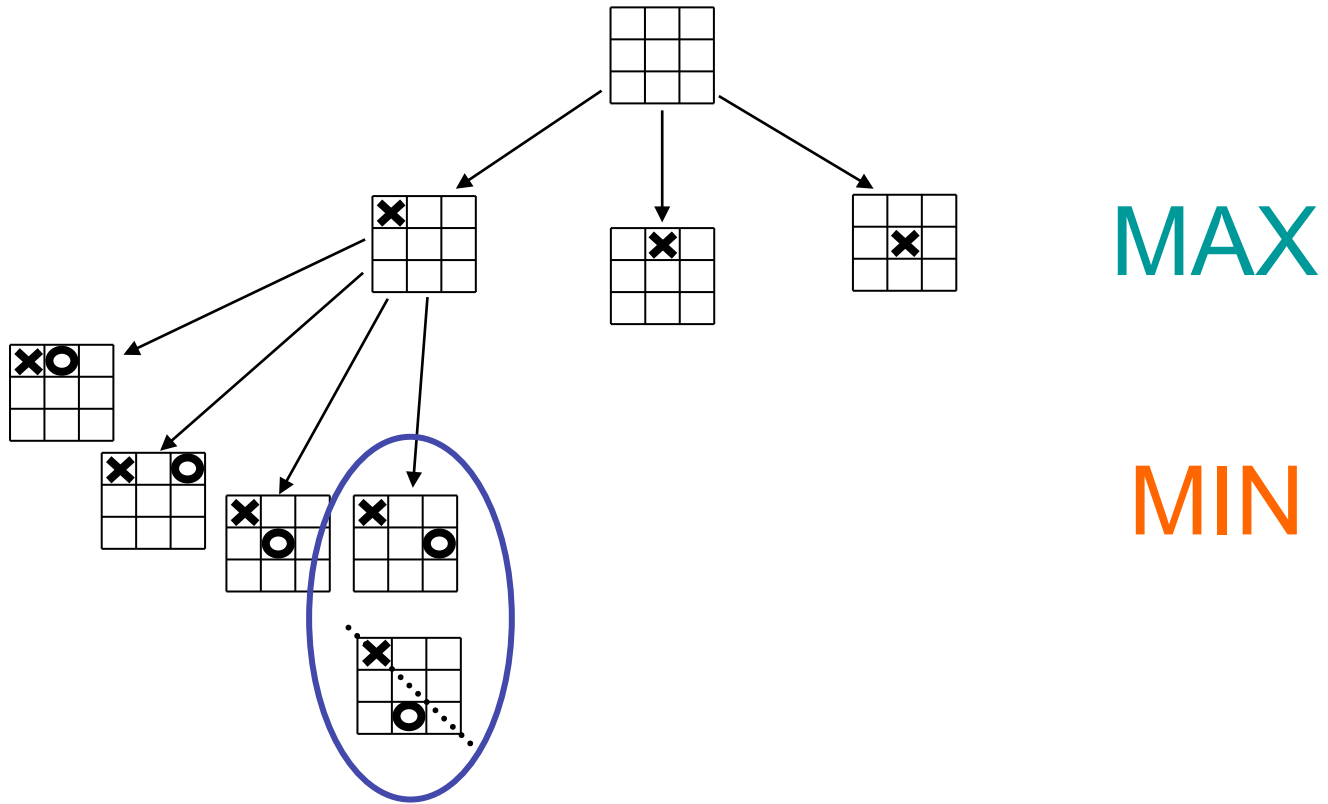




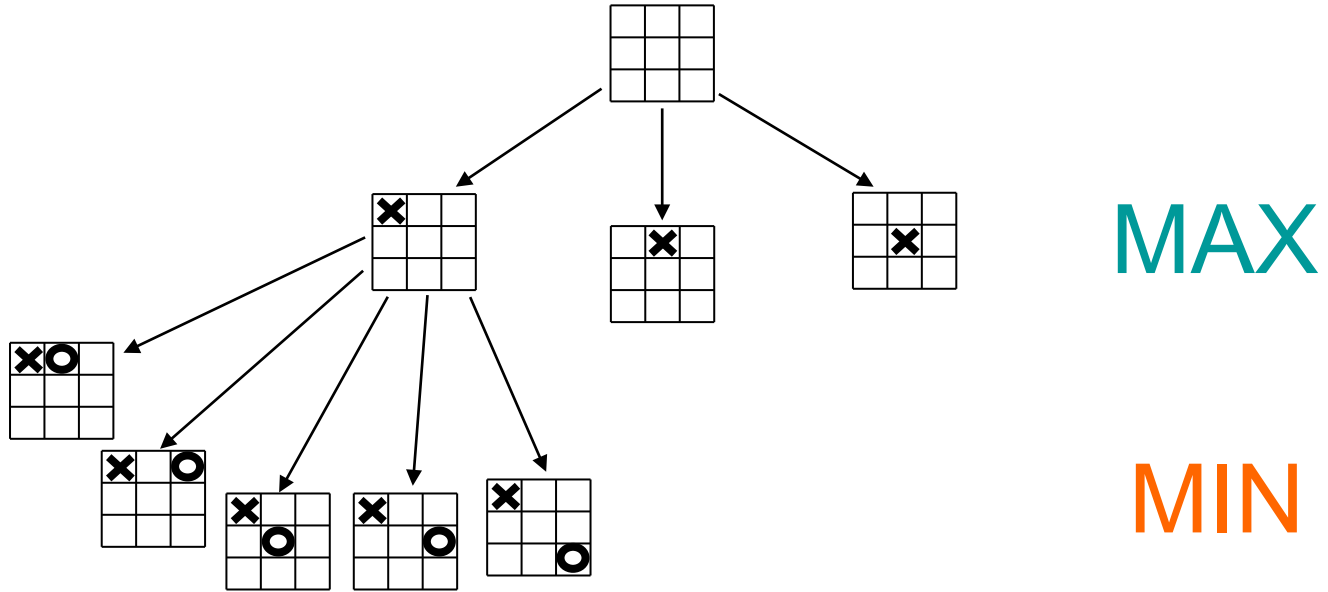
# Arborele de joc



# Arborele de joc



# Arborele de joc



# Valoarea unei stări

Câștig pentru MAX:  $+\infty$

X	O	
	X	O
O		X

# Valoarea unei stări

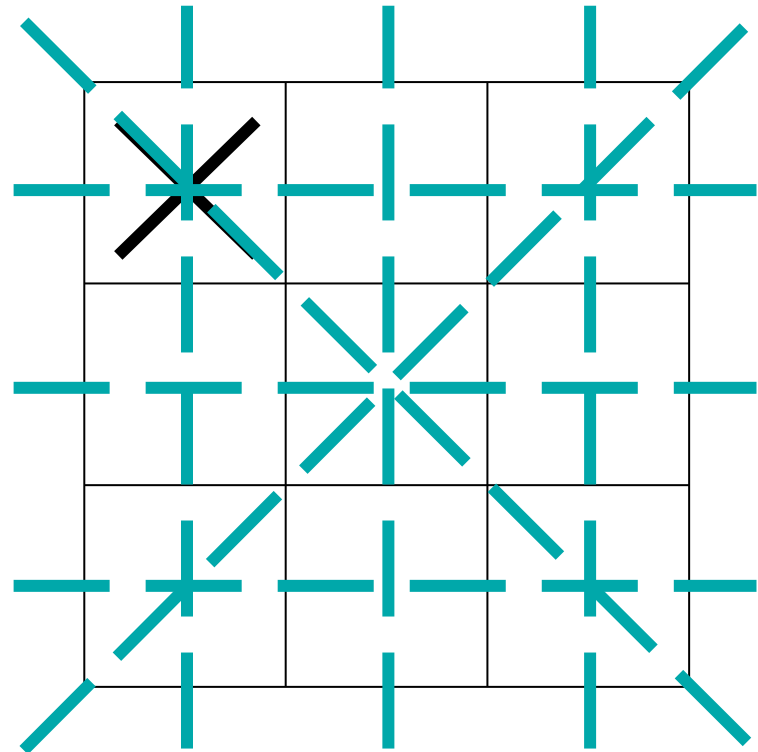
Câștig pentru MIN:  $-\infty$

X	X	O
	O	
O		X

# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

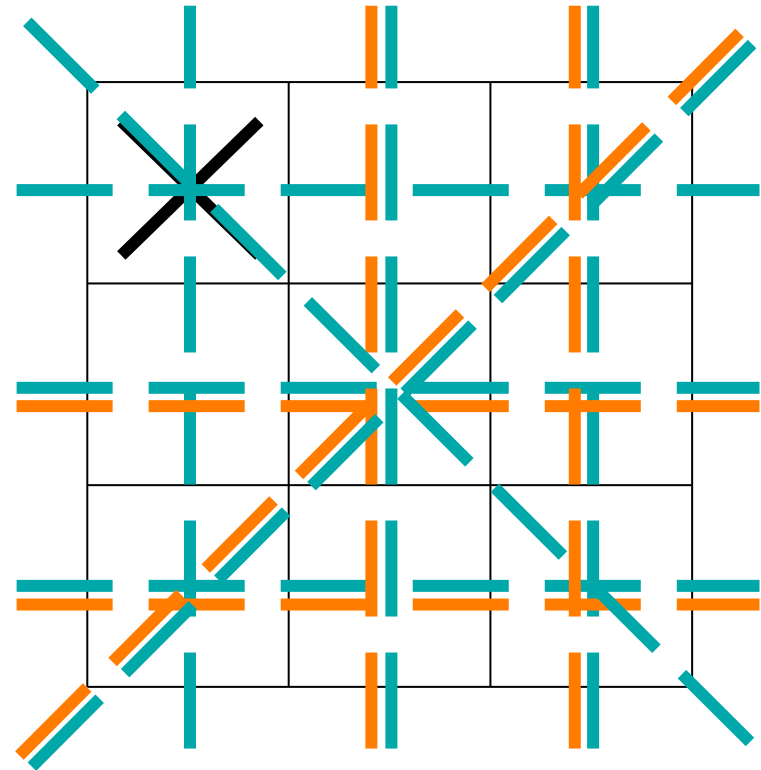
Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*



# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*



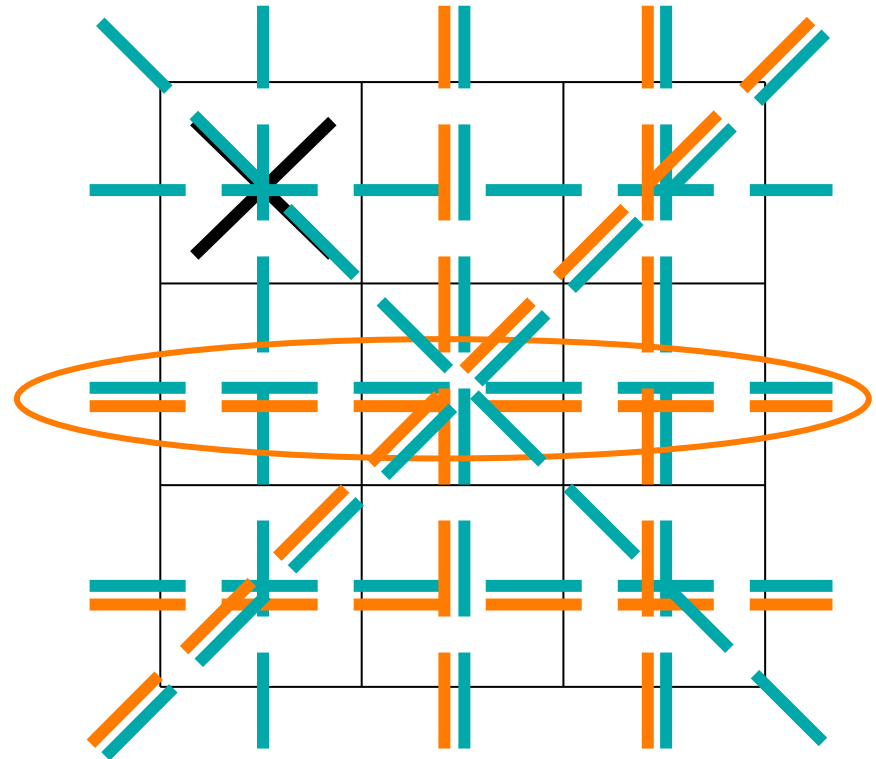
$$8 - 5 = 3$$

# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...



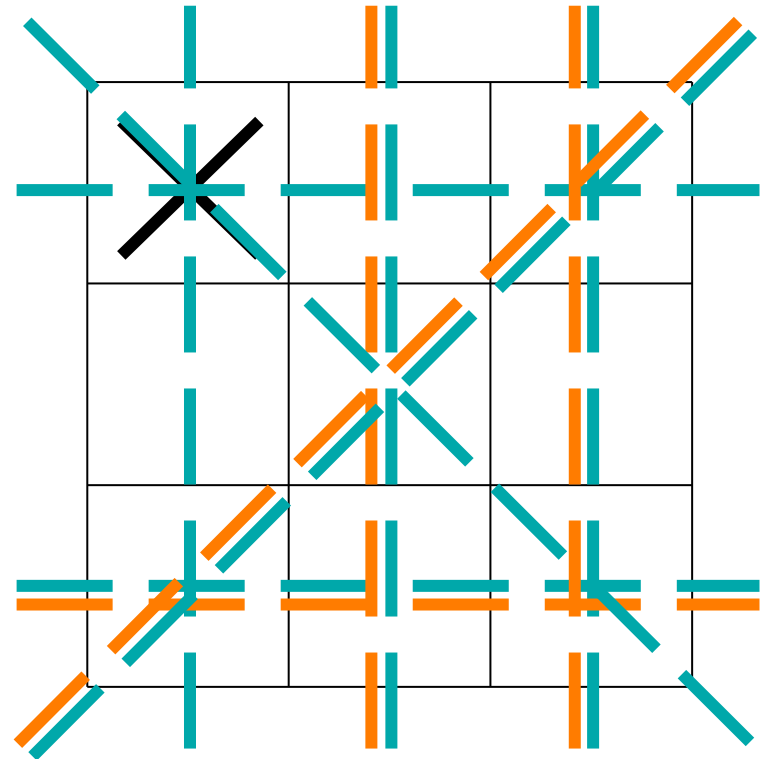


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

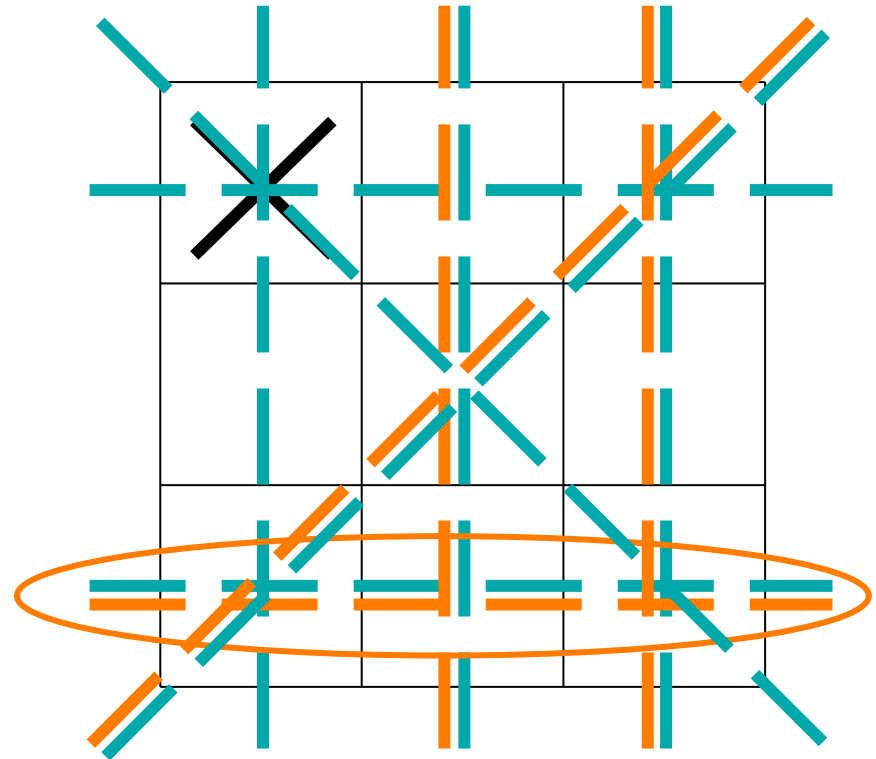


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

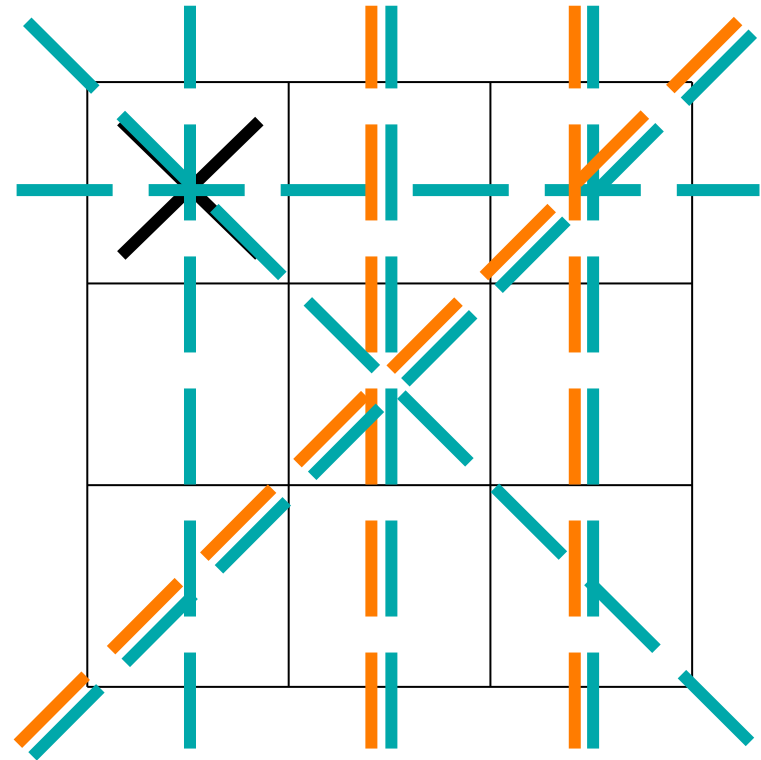


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

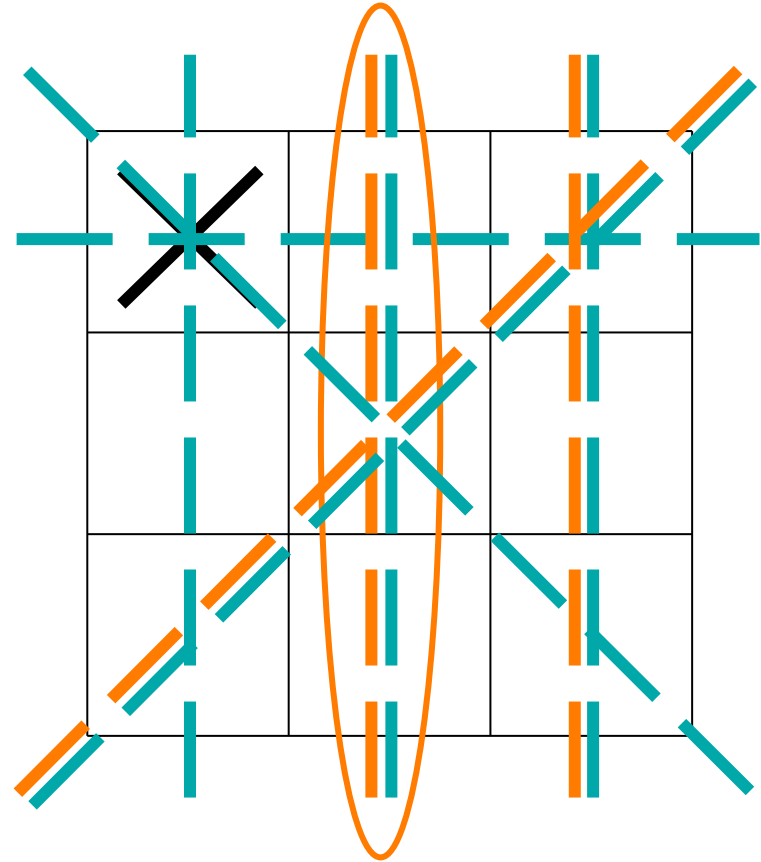


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

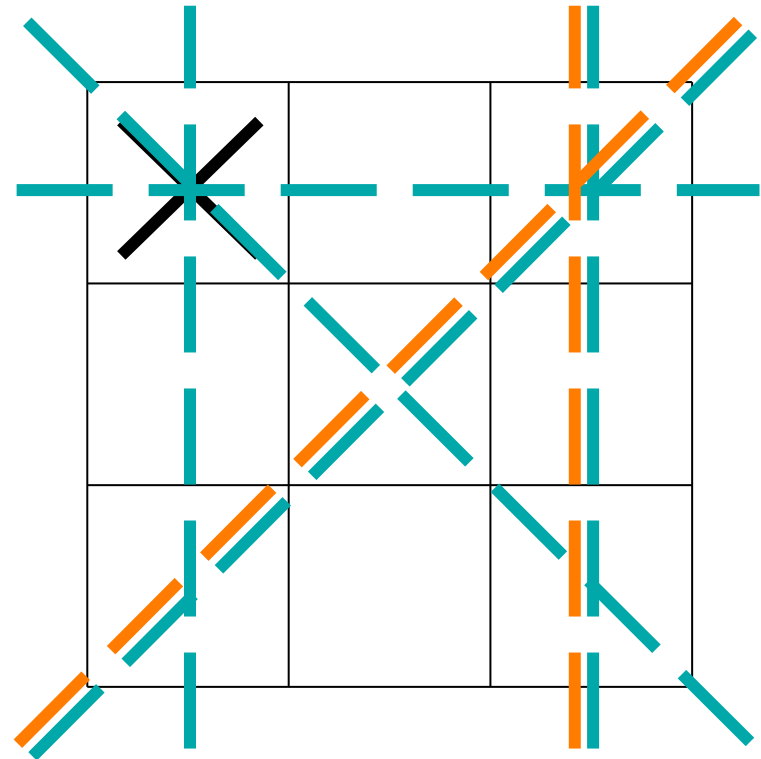


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

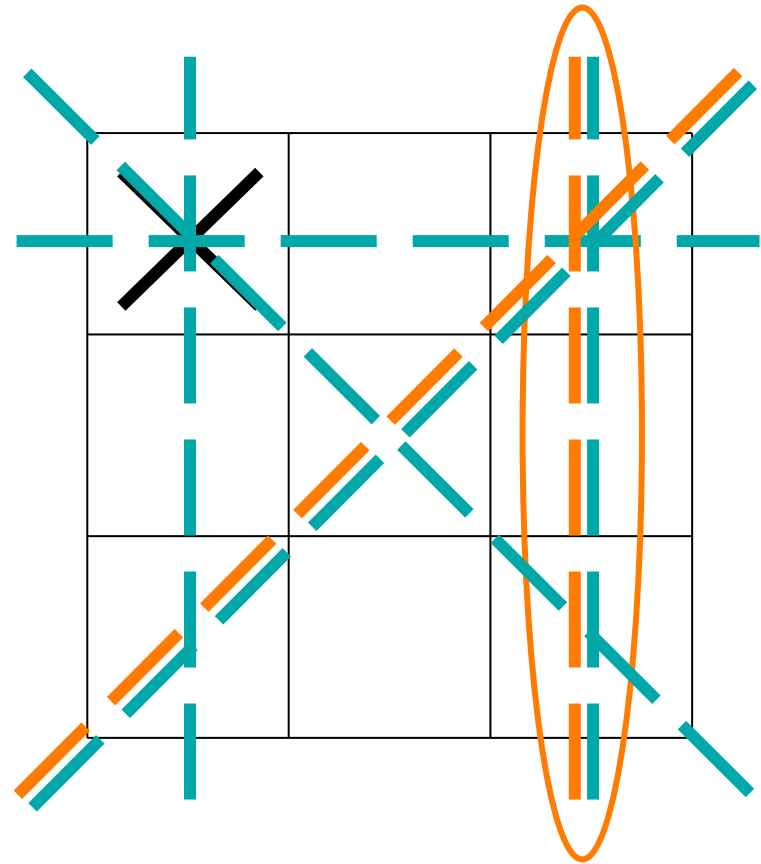


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

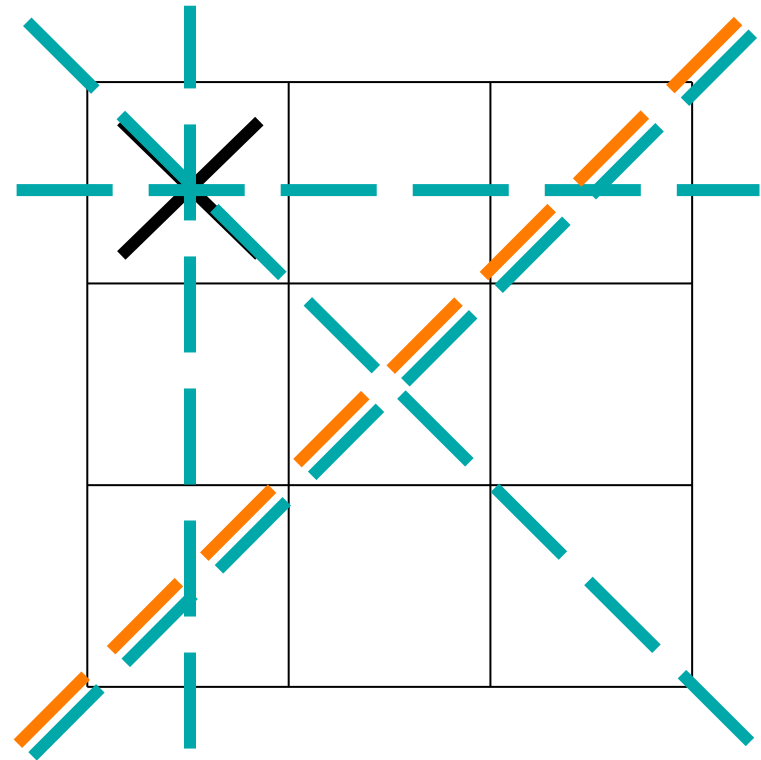


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

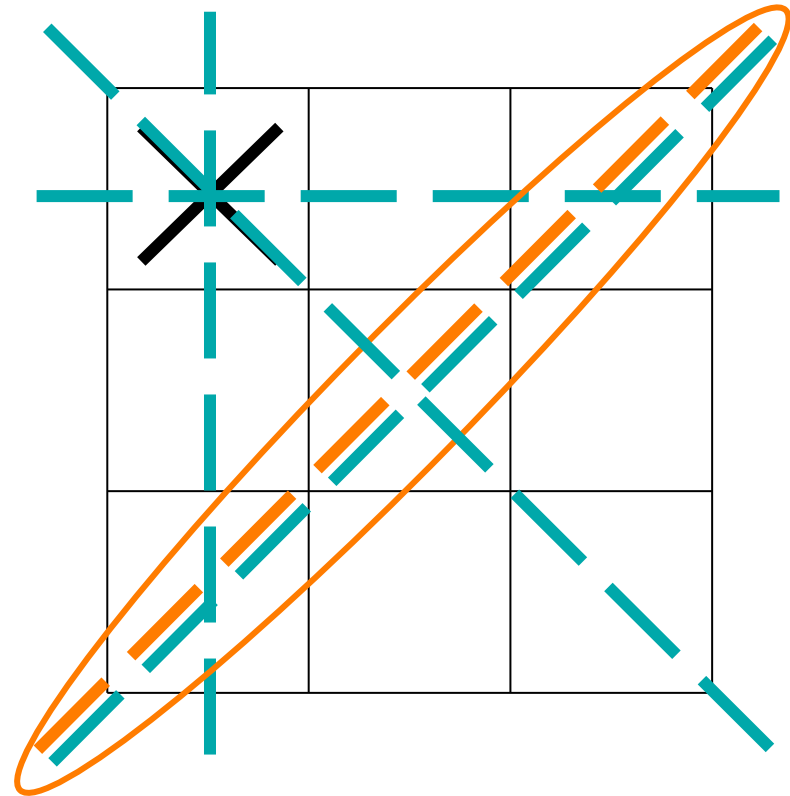


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...



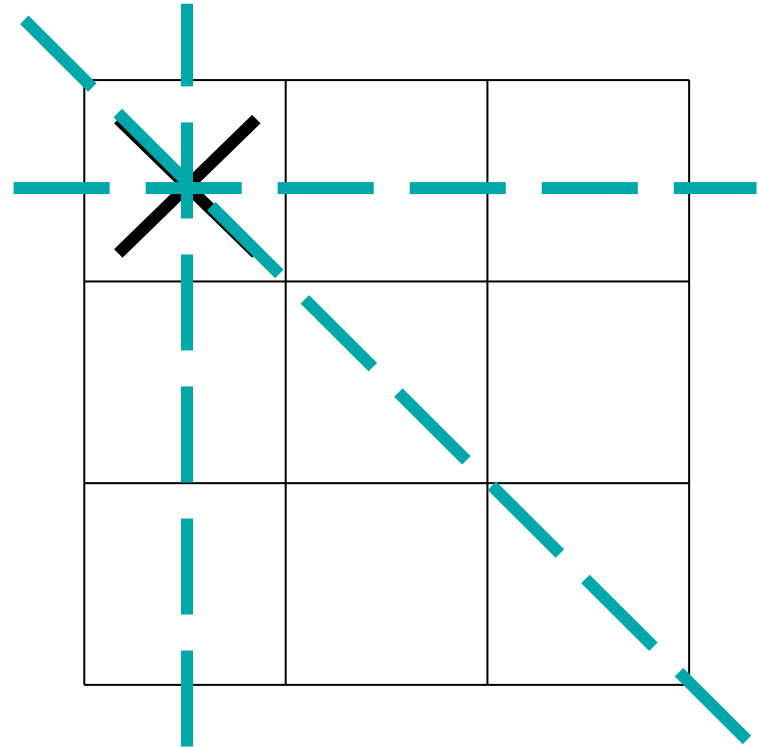


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...

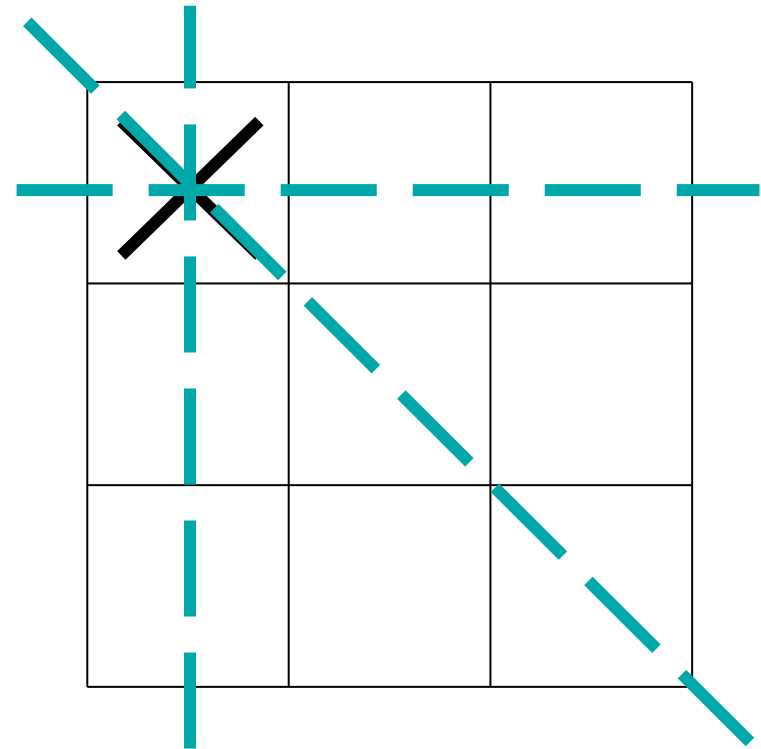


# Evaluarea unei stări

O stare este mai bună dacă deschide mai multe posibilități de câștig până la sfârșitul jocului.

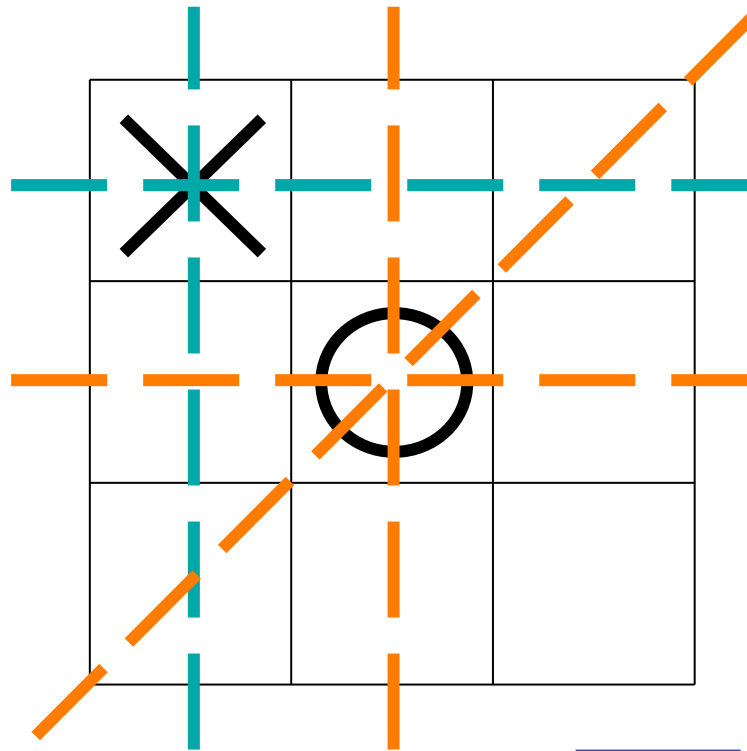
Un exemplu de funcție de evaluare:  
*valoarea stării este diferența dintre numărul de linii pe care le mai poate completa MAX și cele pe care le mai poate completa MIN.*

Liniile fără nici un semn pot fi luate de ambii jucători...



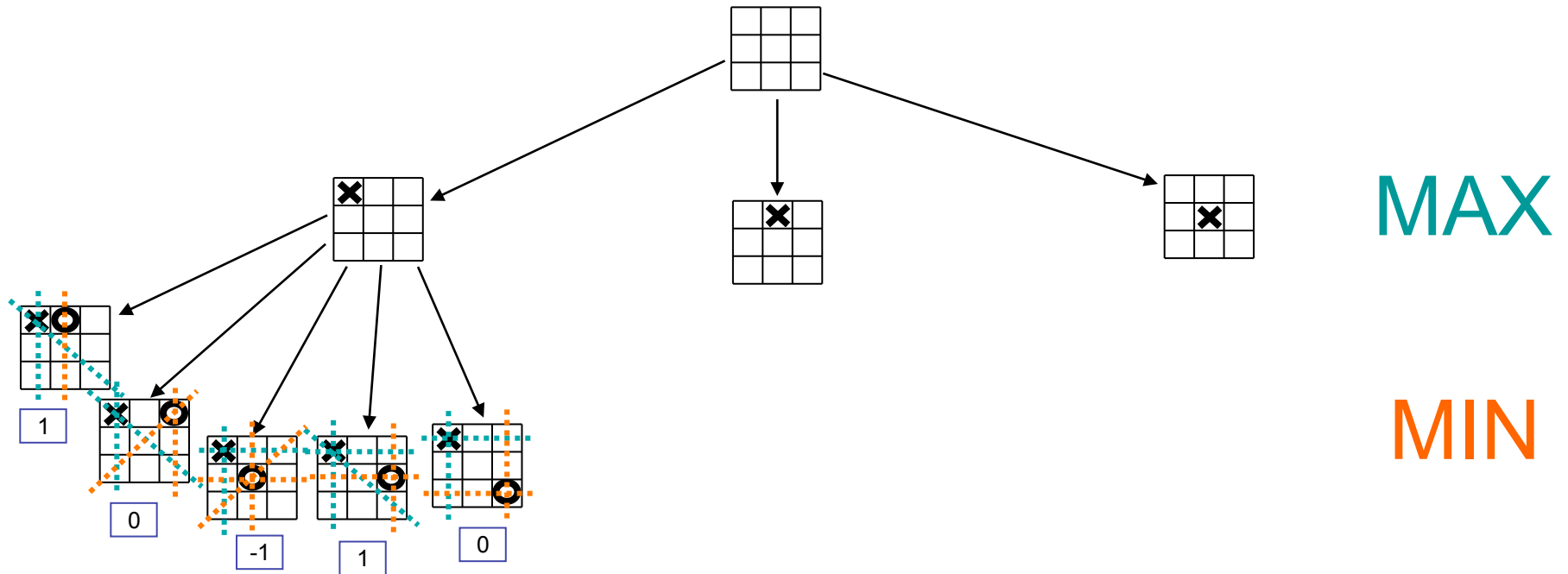
$$3 - 0 = 3$$

# Evaluarea unei stări

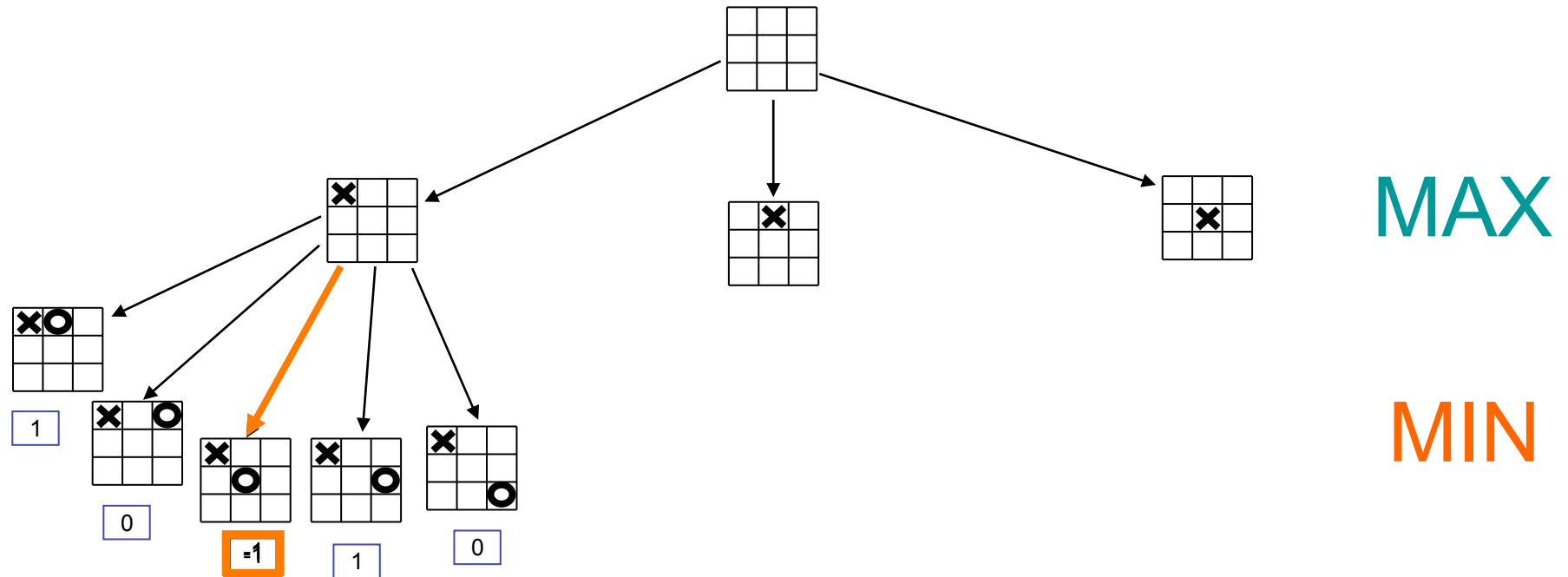


$$2 - 3 = -1$$

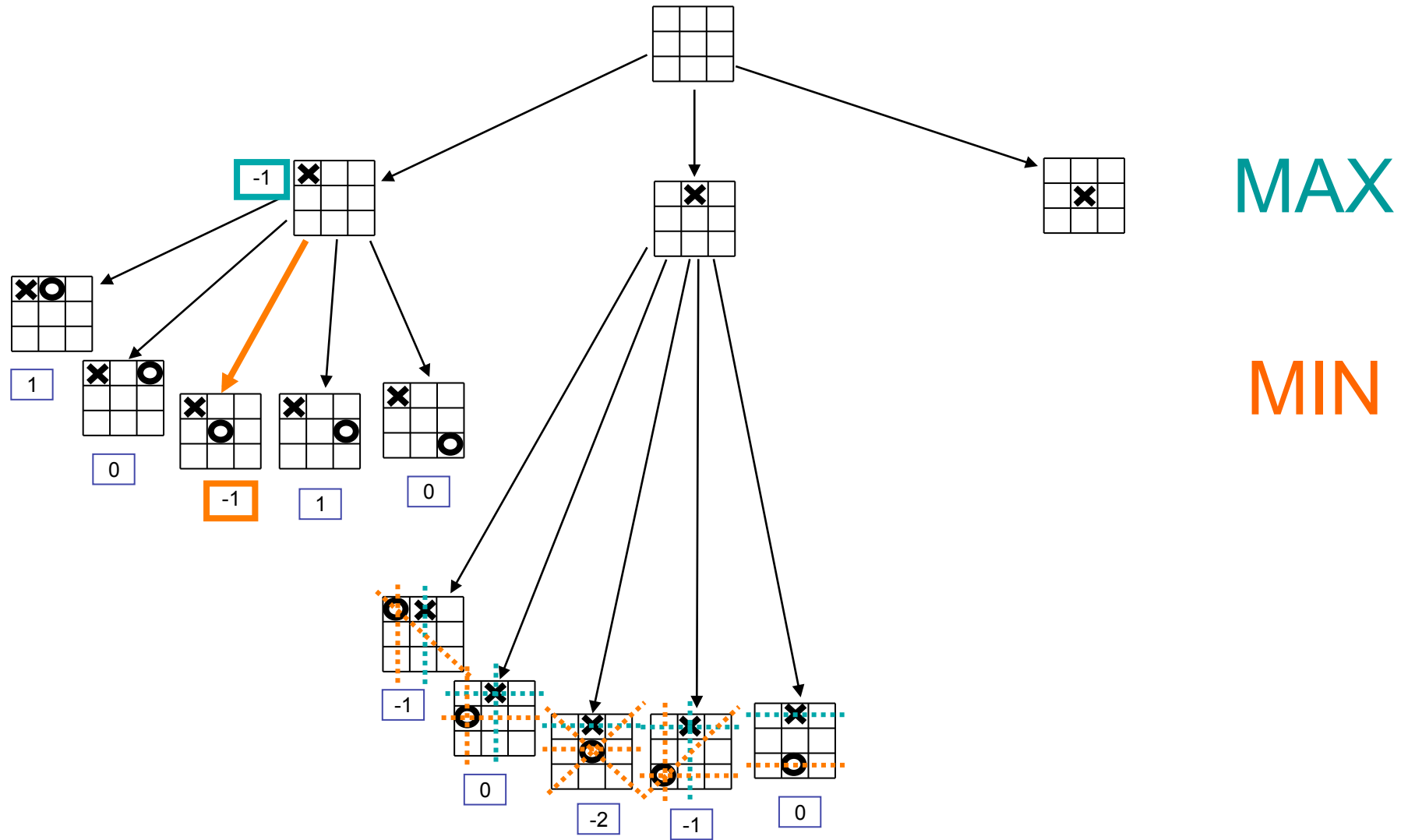
# Evaluarea: de jos în sus



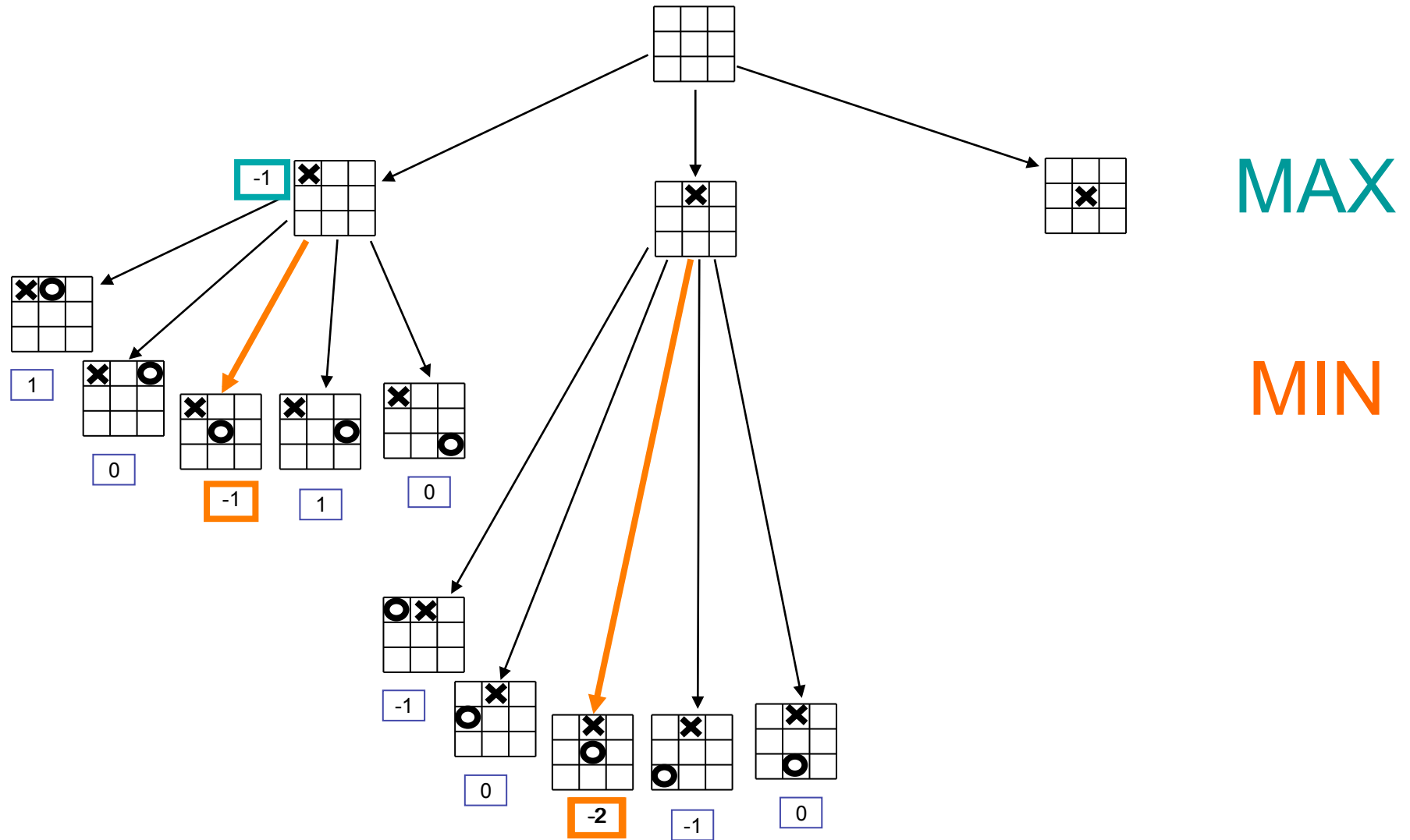
# Evaluarea: de jos în sus



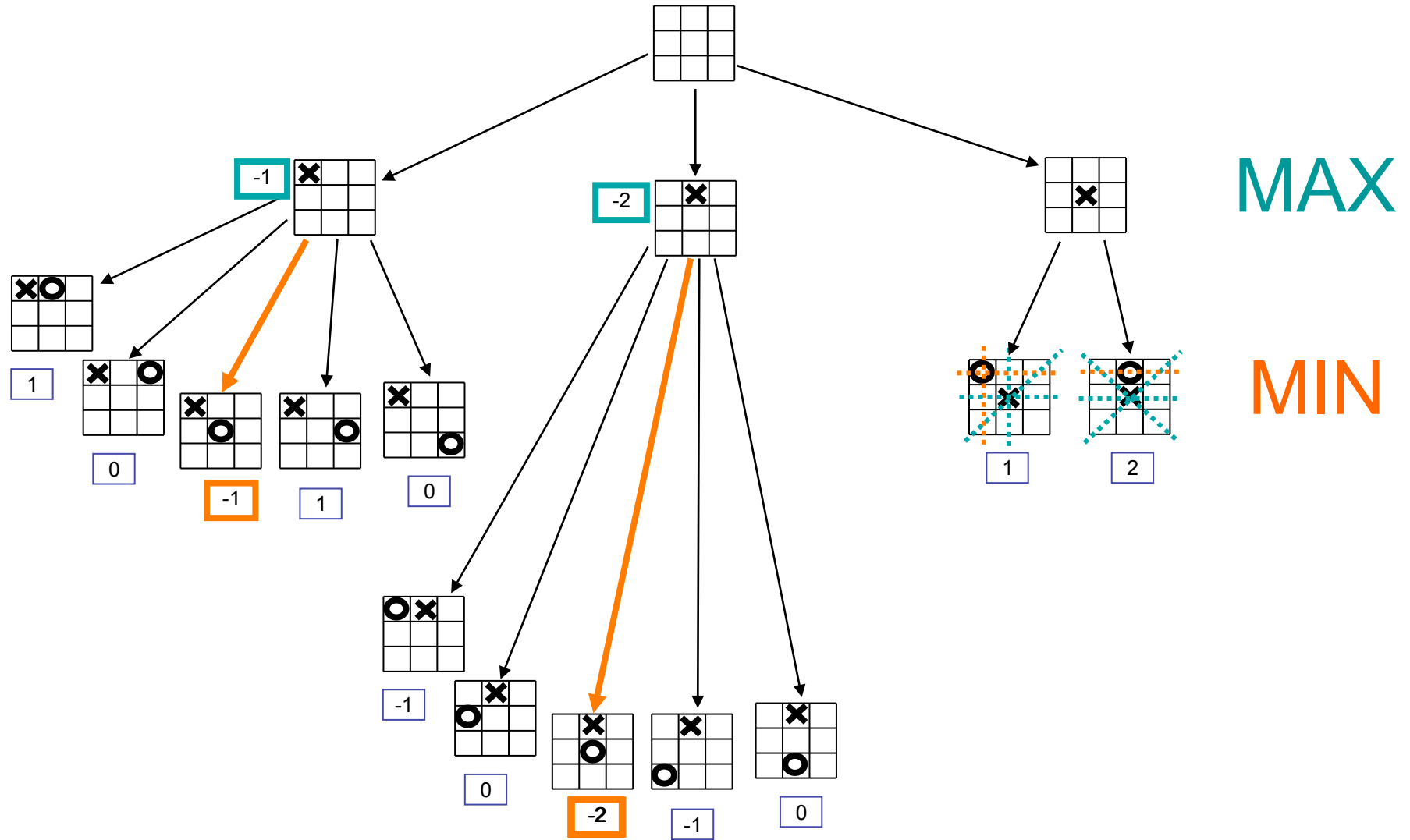
# Evaluarea: de jos în sus



# Evaluarea: de jos în sus

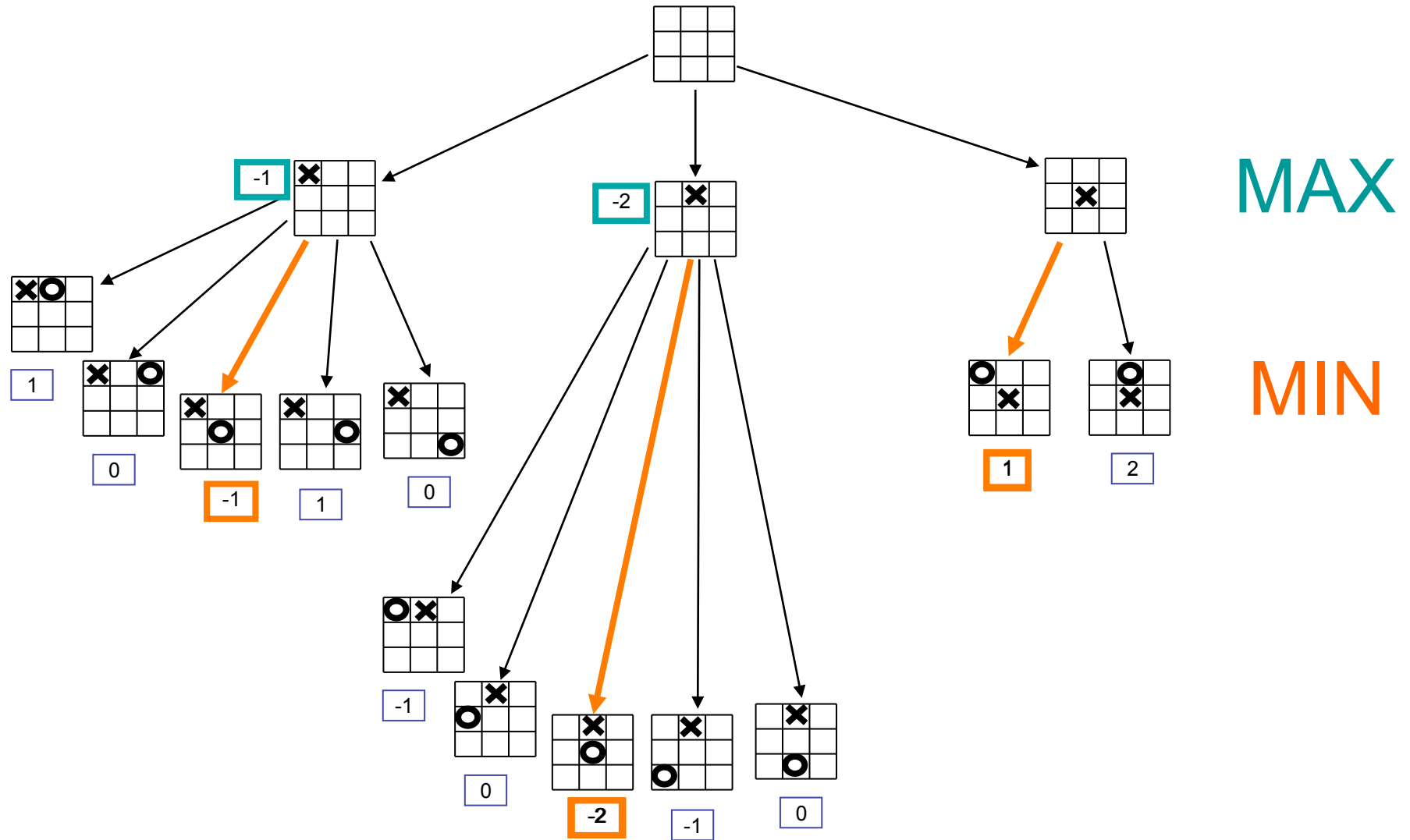


# Evaluarea: de jos în sus

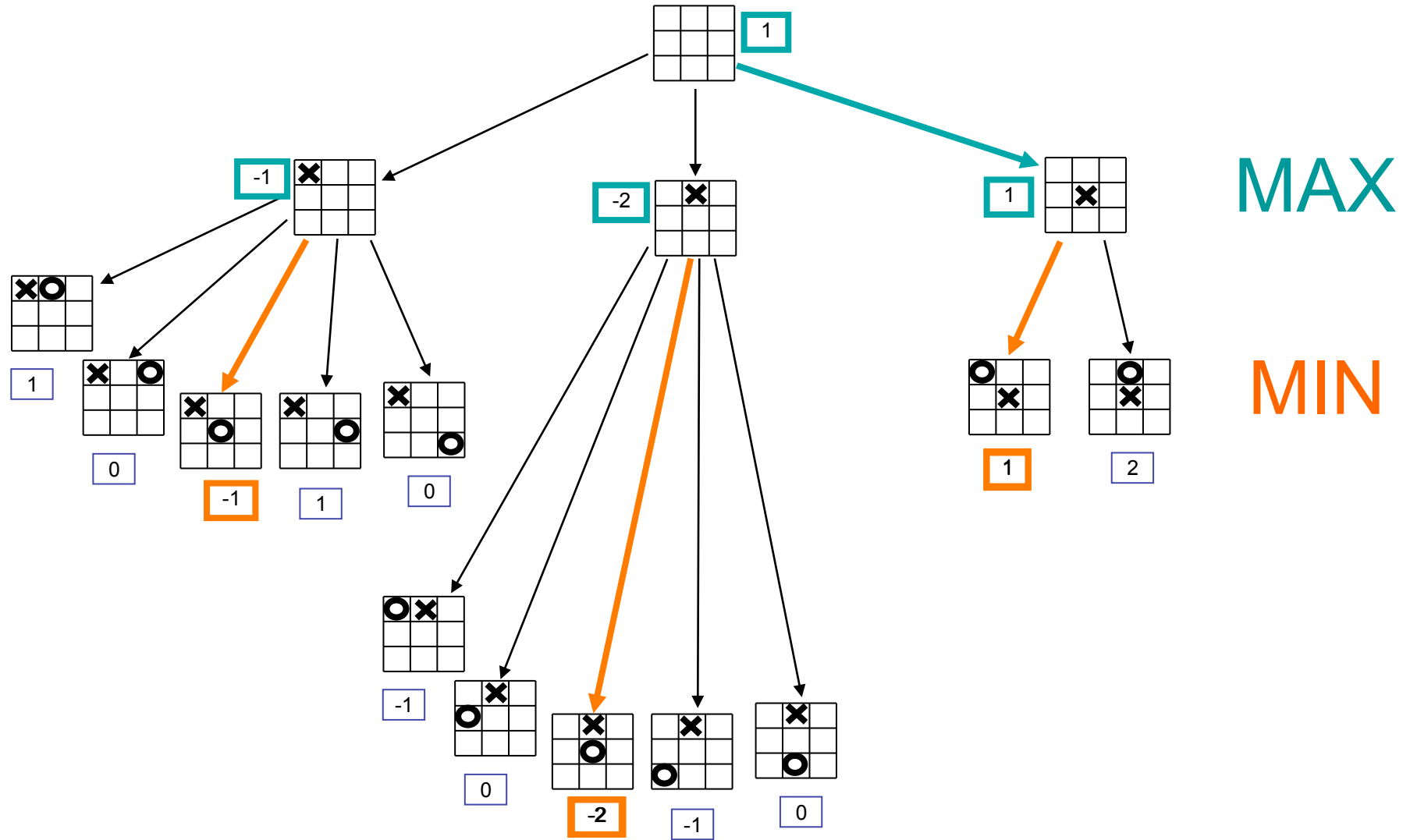




# Evaluarea: de jos în sus



# Evaluarea: de jos în sus






# Metoda MIN-MAX

```
function min-max(state, player, depth)  
begin  
  if (depth = 0) then return score(state);  
  val = worst(player);  
  while (mai sunt stări de generat) begin  
    generez o stare -> s;  
    val <- back-up-compare(val, min-max(s, not(player), depth-1), player);  
    // următoarea mișcare micșorează spațiul de căutare în cazul în care se obține poziția de câștig într-una  
    // din stările generate:  
    if (val = -worst(player)) return(val);  
  end  
  return(val);  
end
```

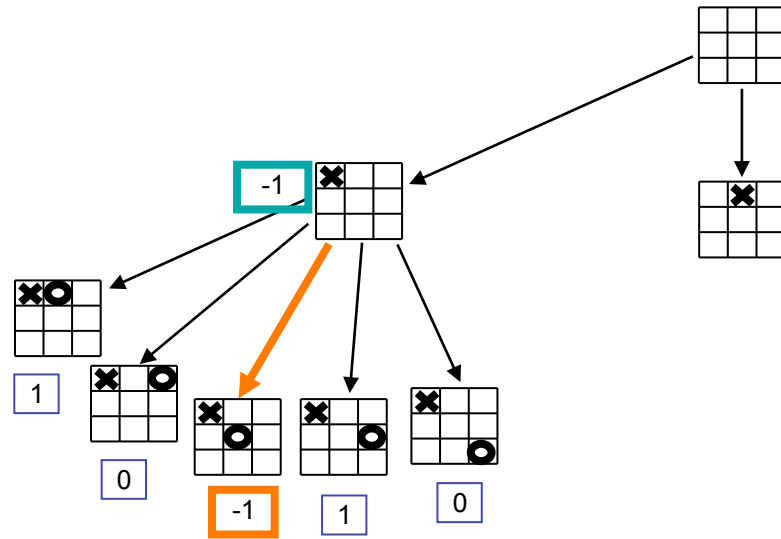
```
function worst(player)  
begin  
  if player = MAX then return  $-\infty$ ;  
  else return  $+\infty$ ;  
end
```

```
function back-up-compare(val1, val2, player)  
begin  
  if player = MAX then return max(val1, val2);  
  else return min(val1, val2);  
end
```

Apelul:

min-max(, MAX, 2)

# Evaluarea: de jos în sus



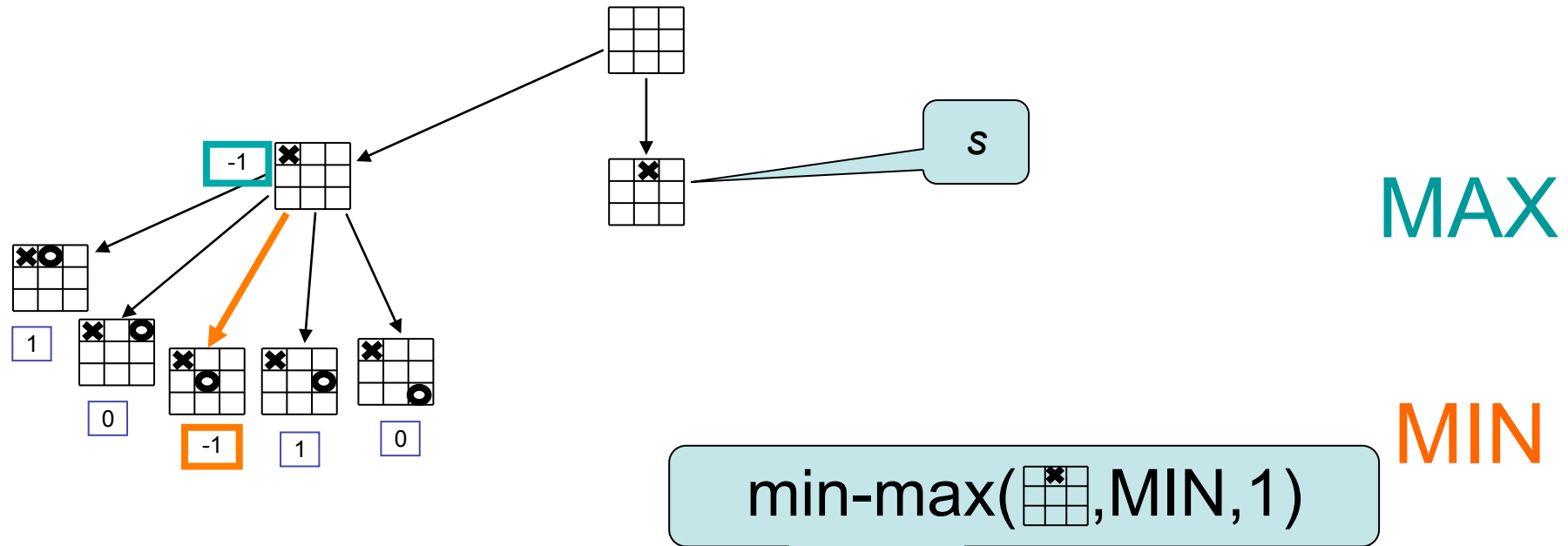
MAX

MIN

```
val=-1; player = MAX; depth=1;
```

```
while (mai sunt stări de generat) begin  
    generez o stare -> s;  
    ...  
end
```

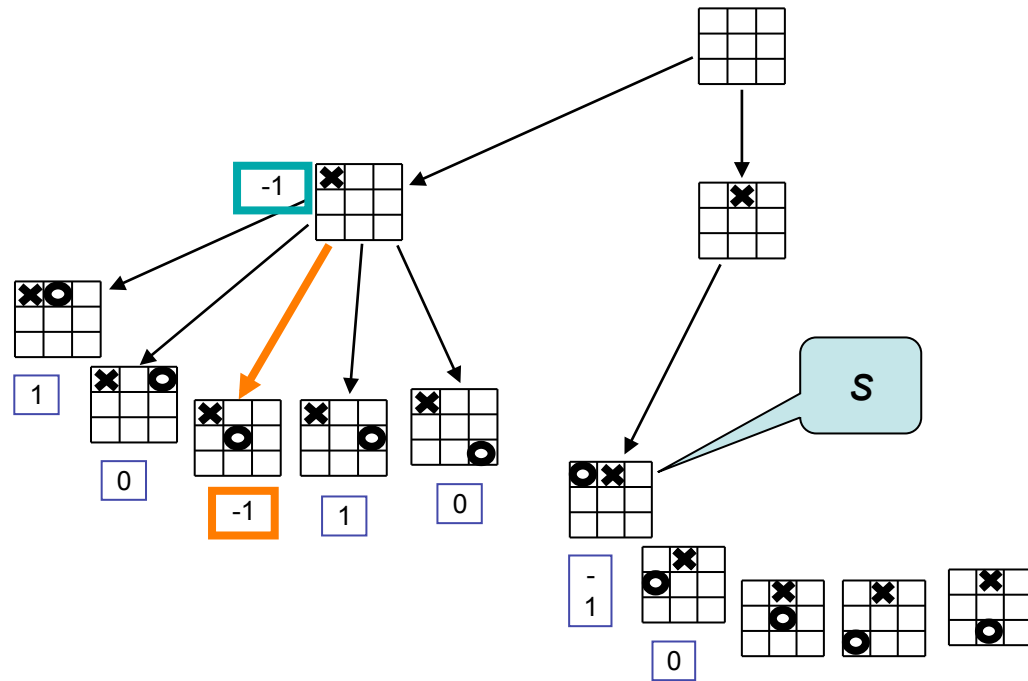
# Evaluarea: de jos în sus



`val=-1; player = MAX; depth=2;`

```
while (mai sunt stări de generat) begin  
  generează o stare -> s;  
  val <- back-up-compare(val, min-max(s, not(player), depth-1), player);  
  if (val = -worst(player)) return(val);  
end
```

# Evaluarea: de jos în sus




MAX

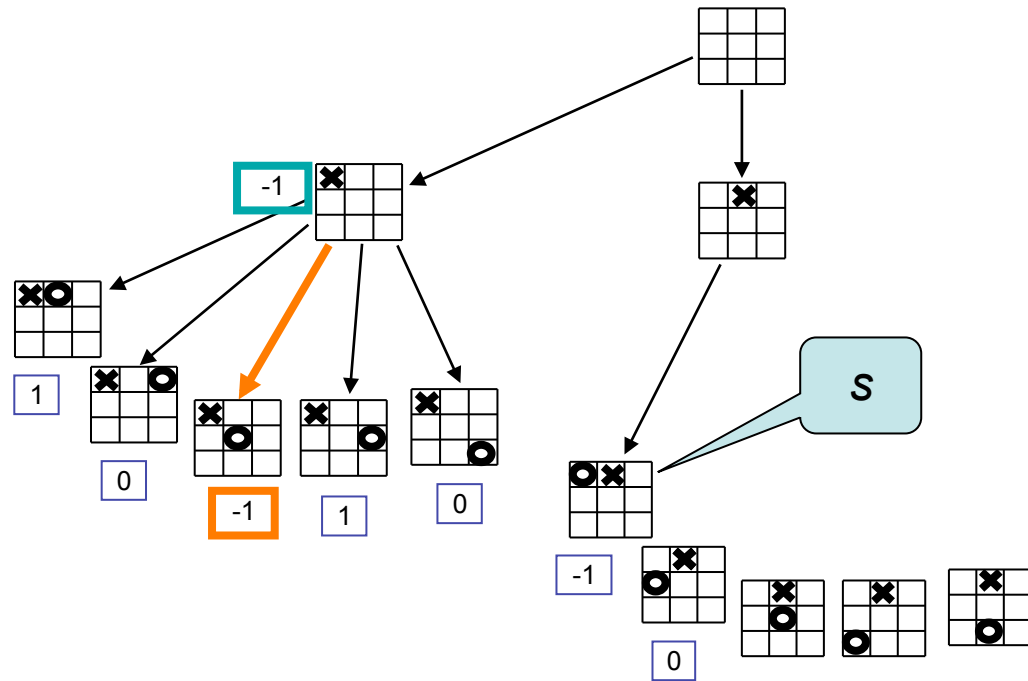
MIN

min-max(, MIN, 1)

```

val = worst(player);  val=∞
while (mai sunt stări de generat) begin
  generez o stare -> s;
  val <- back-up-compare(val, min-max(s, not(player), depth-1), player);
  if (val = -worst(player)) return(val);
end
    
```

# Evaluarea: de jos în sus



MAX

MIN

$\text{min-max}(\begin{array}{|c|c|c|} \hline \text{o} & \text{x} & \\ \hline & & \\ \hline & & \\ \hline \end{array}, \text{MAX}, 0)$

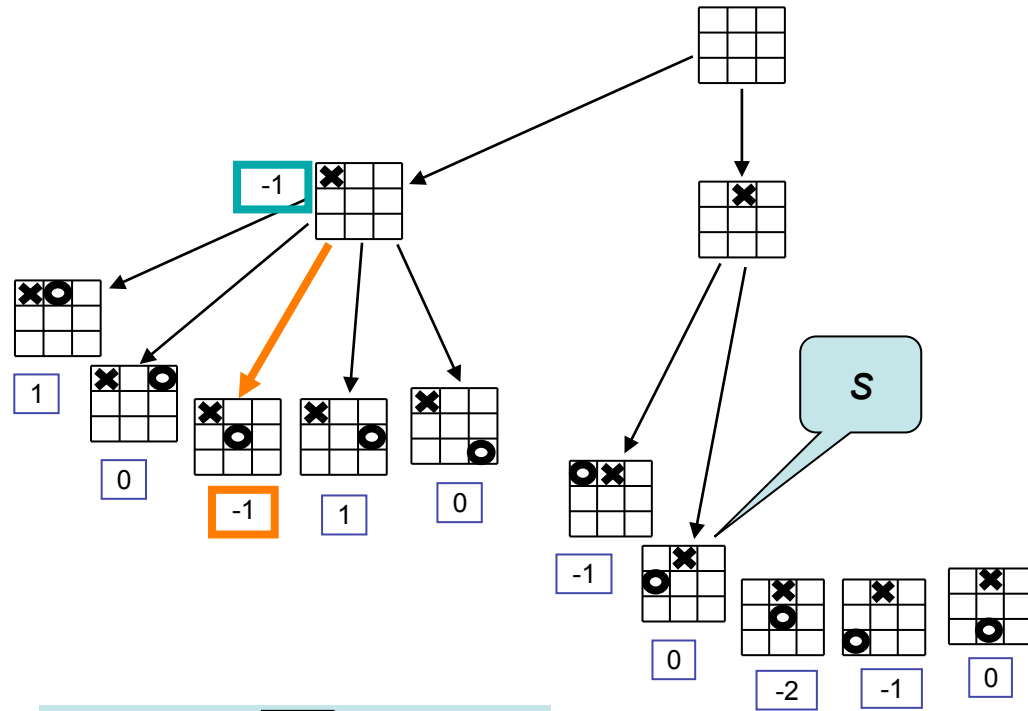
if ( $depth = 0$ ) then return  $score(state)$ ;



-1



# Evaluarea: de jos în sus



MAX

MIN

min-max( ,MIN,1)

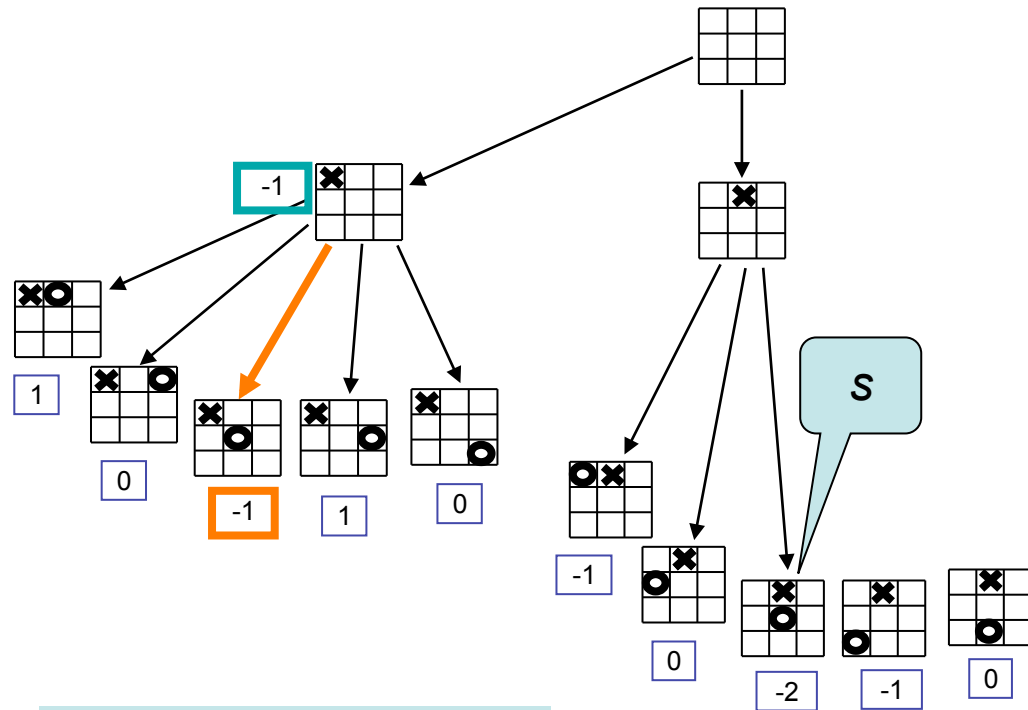
val= $\infty$ ; player=MIN;

```

val <- back-up-compare(val, -1, player)
if (val = -worst(player)) return(val);
end
    
```

-1

# Evaluarea: de jos în sus



MAX

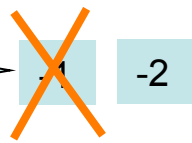
MIN

min-max( , MIN, 1)

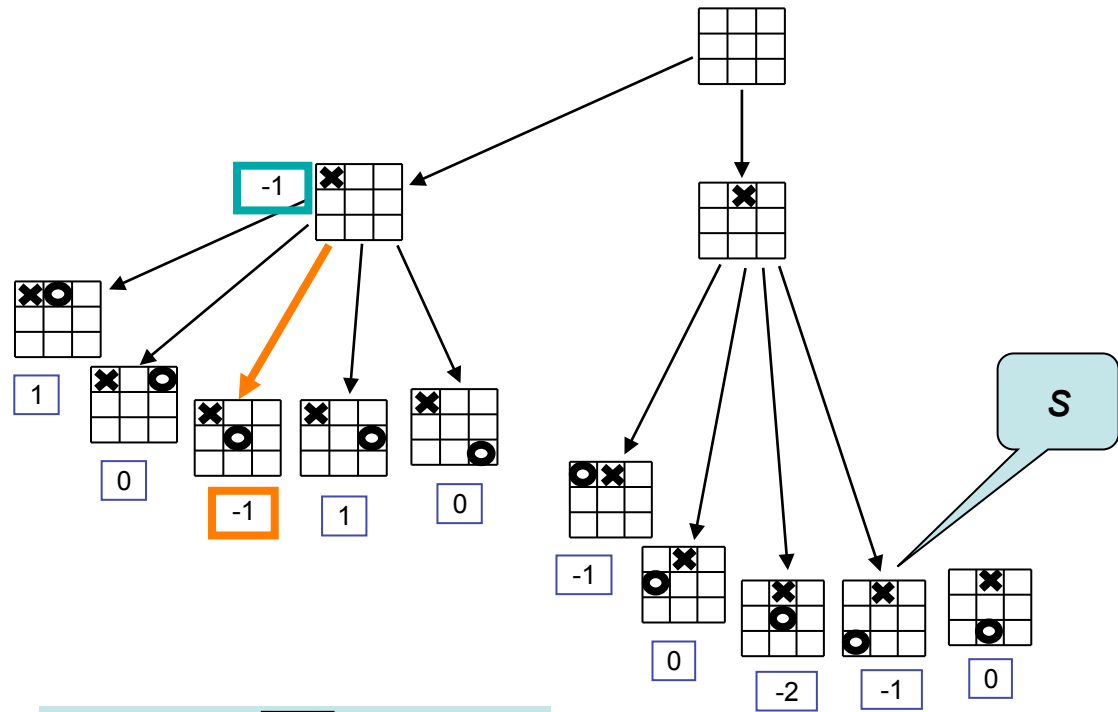
val= $\infty$ ; player=MIN;

```

val <- back-up-compare(val, -1, player)
if (val = -worst(player)) return(val);
end
    
```



# Evaluarea: de jos în sus



MAX

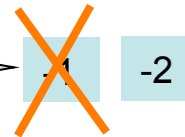
MIN

min-max( , MIN, 1)

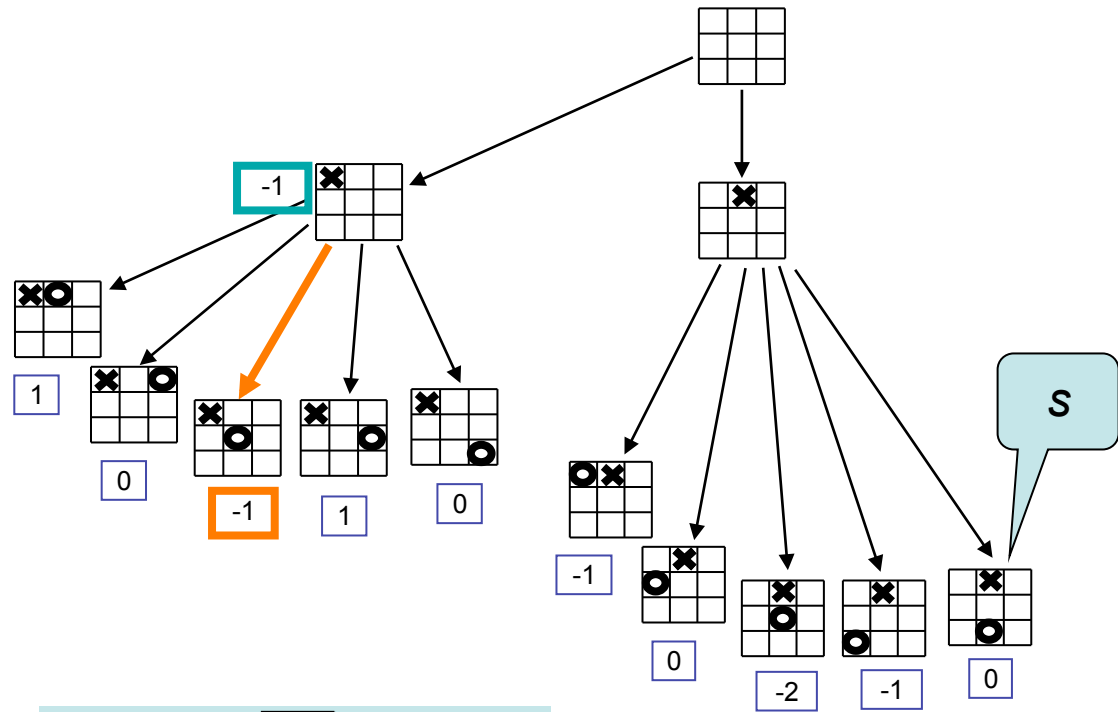
val = ∞; player = MIN;

```

val <- back-up-compare(val, -1, player)
if (val = -worst(player)) return(val);
end
    
```



# Evaluarea: de jos în sus



MAX

MIN

min-max( , MIN, 1)

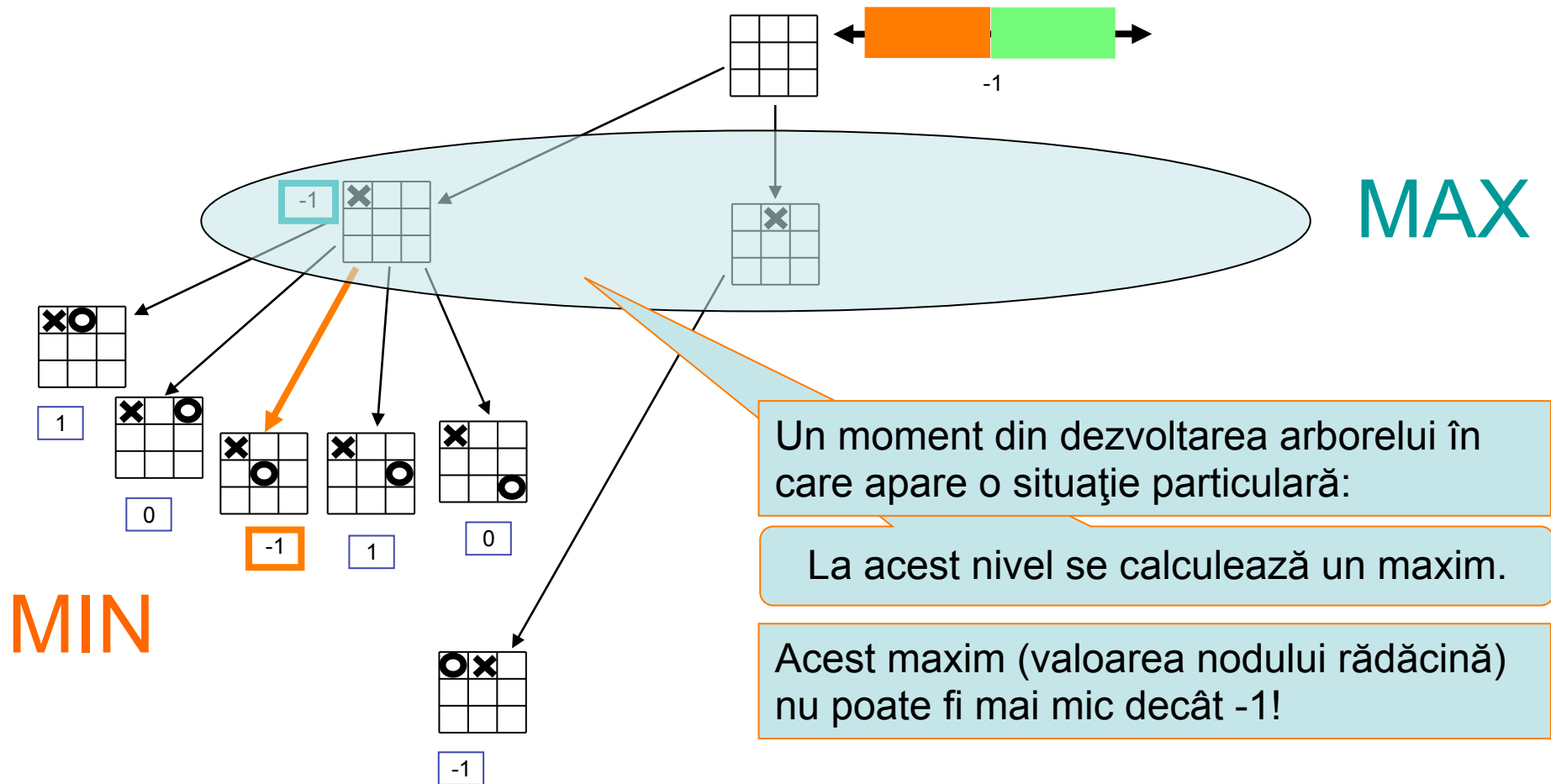
val = ∞; player = MIN;

```

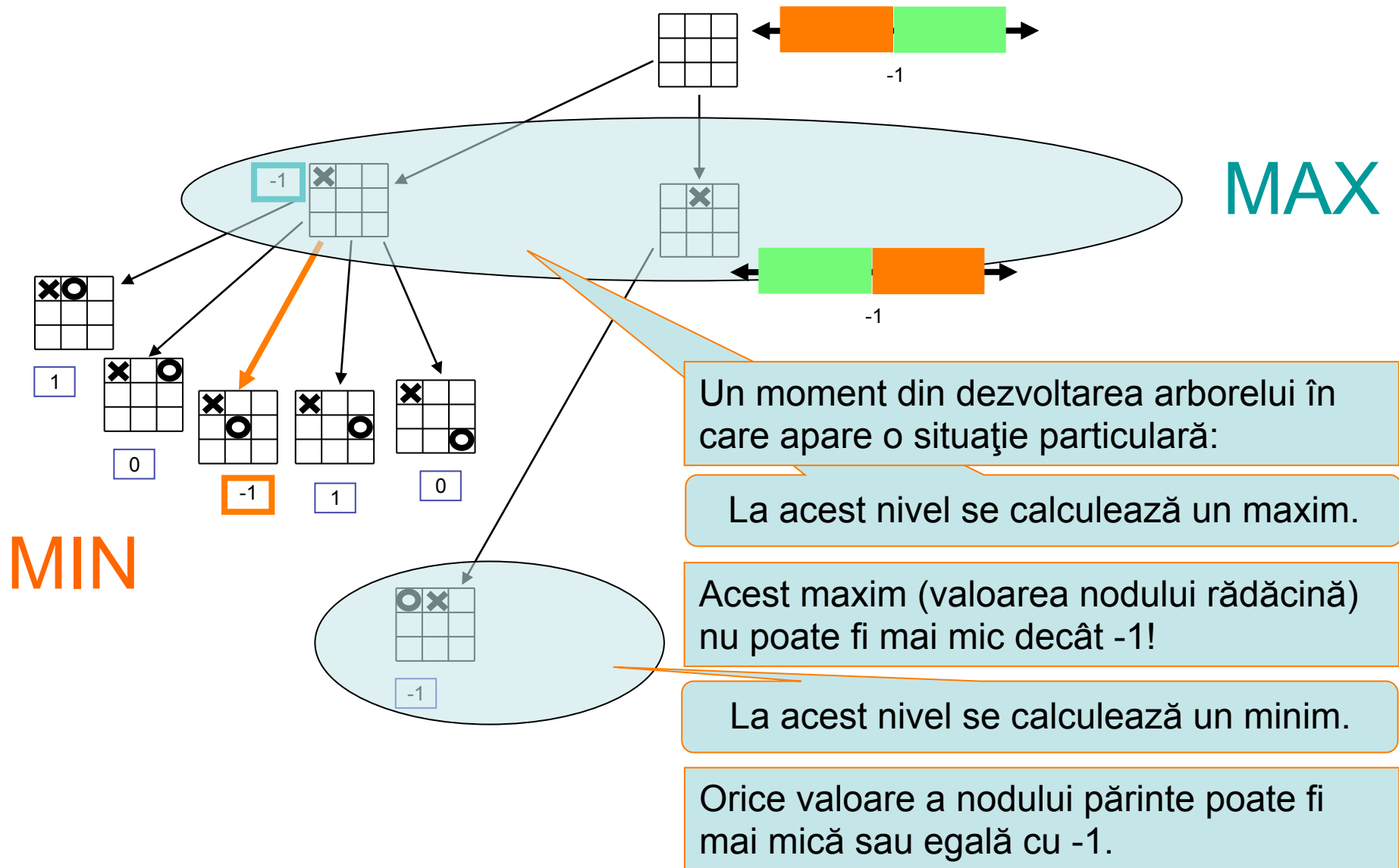
val <- back-up-compare(val, -1, player)
if (val = -worst(player)) return(val);
end
    
```

~~-1~~ -2

# Metoda alpha-beta

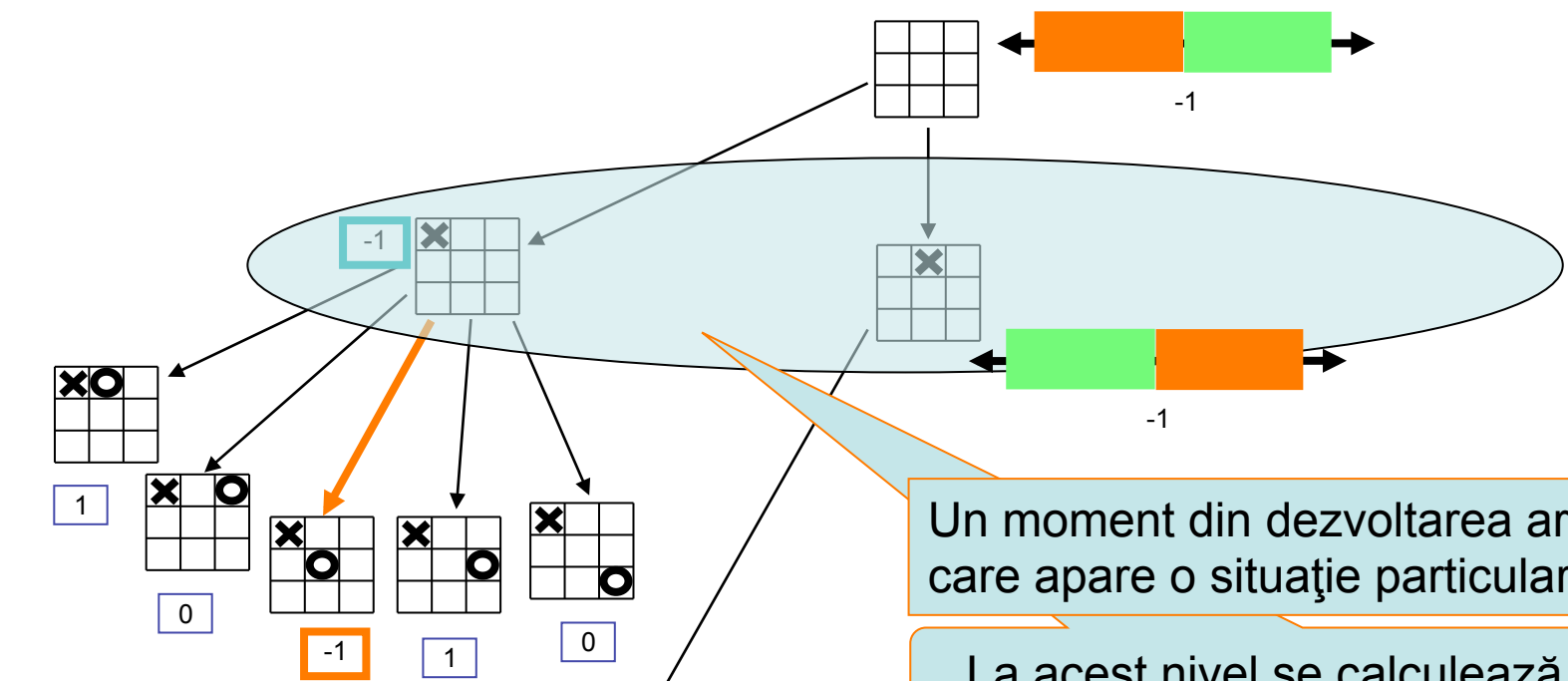


# Metoda alpha-beta



# Metoda alpha-beta

MAX



MIN

Generarea poate fi oprită!

Ea nu mai poate influența valoarea nodului rădăcină!

Un moment din dezvoltarea arborelui în care apare o situație particulară:

La acest nivel se calculează un maxim.

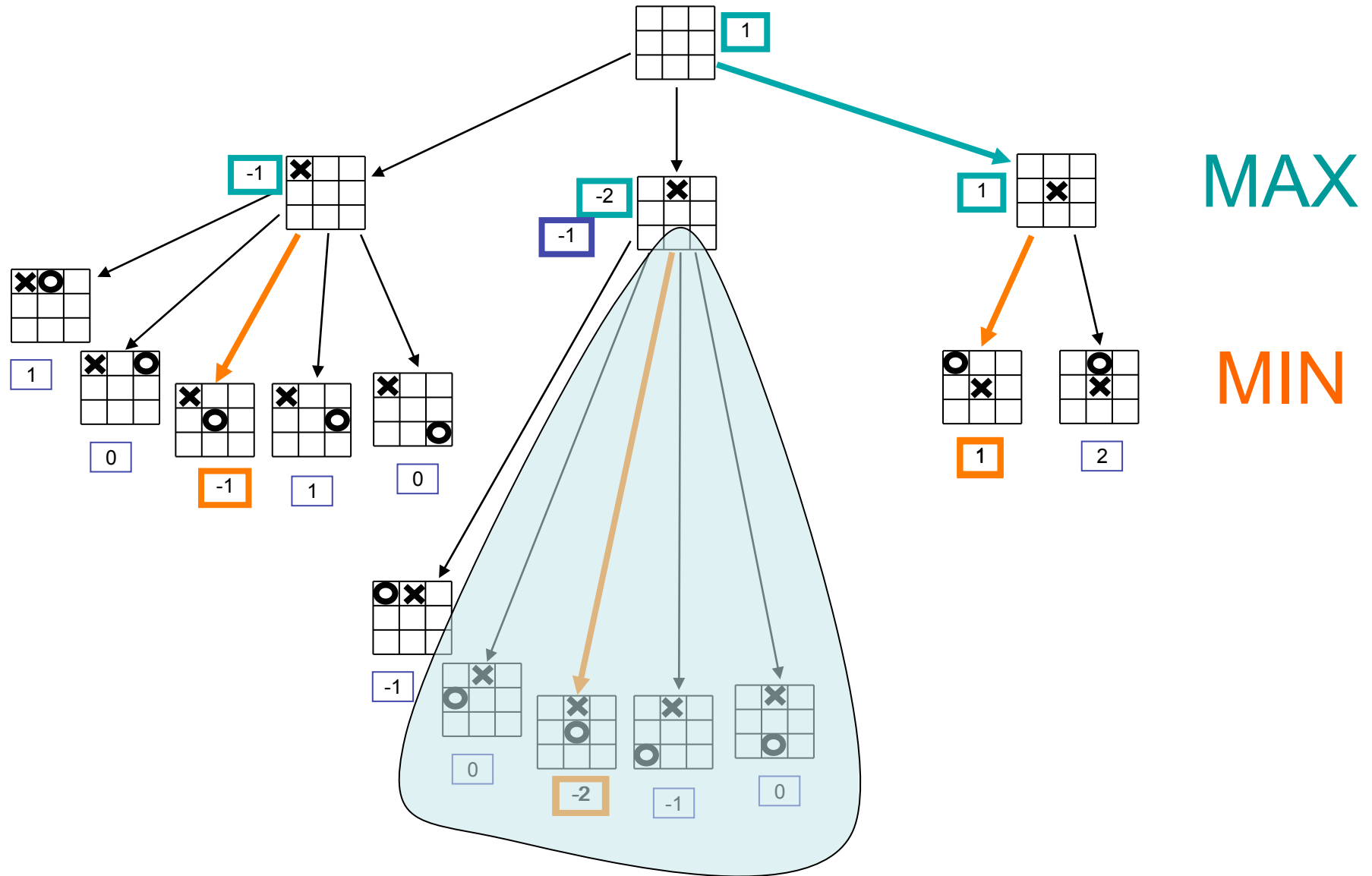
Acest maxim (valoarea nodului rădăcină) nu poate fi mai mic decât -1!

La acest nivel se calculează un minim.

Orice valoare a nodului părinte poate fi mai mică sau egală cu -1.



# Metoda alpha-beta






# Metoda alpha-beta

```
function alpha-beta(state, player, depth)
begin
  if (depth = 0) then return score(state);
  val = worst(player);
  while (mai sunt stări de generat) begin
    generez o stare -> s;
    newval <- alpha-beta(s, not(player), depth-1);
    if player=MAX & newval ≤ val then return(newval);
    else if player=MIN & newval ≥ val then return(newval);
    else val ← back-up-compare(val, min-max(s, not(player), depth-1), player);
      // următoarea mișcare micșorează spațiul de căutare în cazul în care se obține poziția de câștig
      // într-una din stările generate:
    if (val = -worst(player)) return(val);
  end
  return(val);
end
```

```
function worst(player)
begin
  if player = MAX then return  $-\infty$ ;
  else return  $+\infty$ ;
end
```

```
function back-up-compare(val1, val2, player)
begin
  if player = MAX then return max(val1, val2);
  else return min(val1, val2);
end
```

Apelul:

alpha-beta (, MAX, 2)