



Buna!

Eu sunt Teo!

Studenta doctoranda la Oxford!

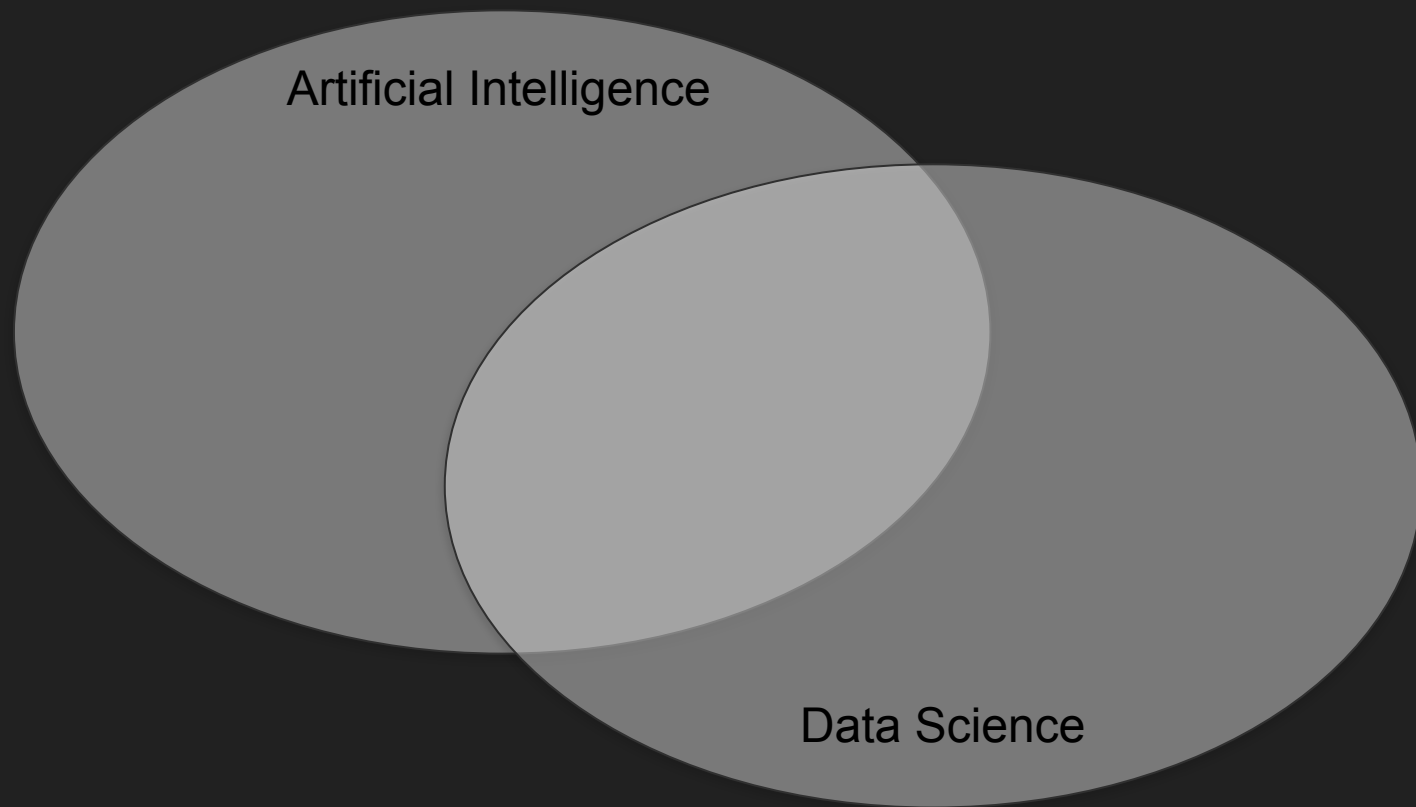
Ma pasioneaza Machine  
Learning si Modele Generative!

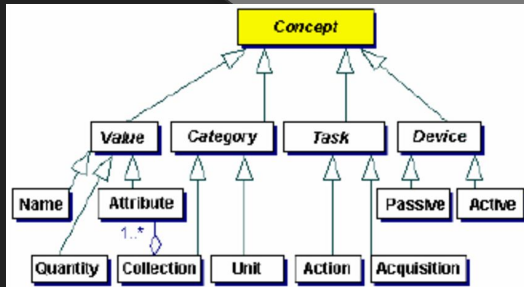
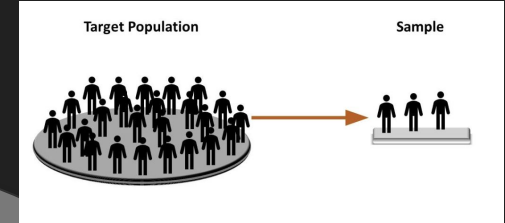
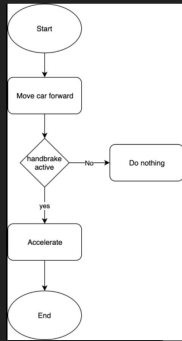
# Atelier în Reinforcement Learning

VitoriiOlimpici.ro

# Ce vom face astăzi?

1. **Prezentare (30-45 minute)**
2. **Discuscutii, Intrebari (10 minute)**
3. **Prezentare Laborator (5 minute)**
4. **Munca individuala (50 minute)**
5. **Trimitere solutii (10 minute)**
6. **Premii pentru cele mai creative lucrări**





Artificial Intelligence

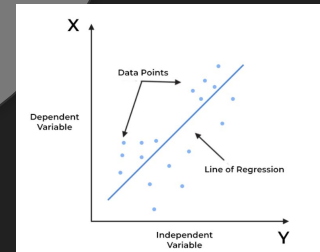
Symbolic Reasoning

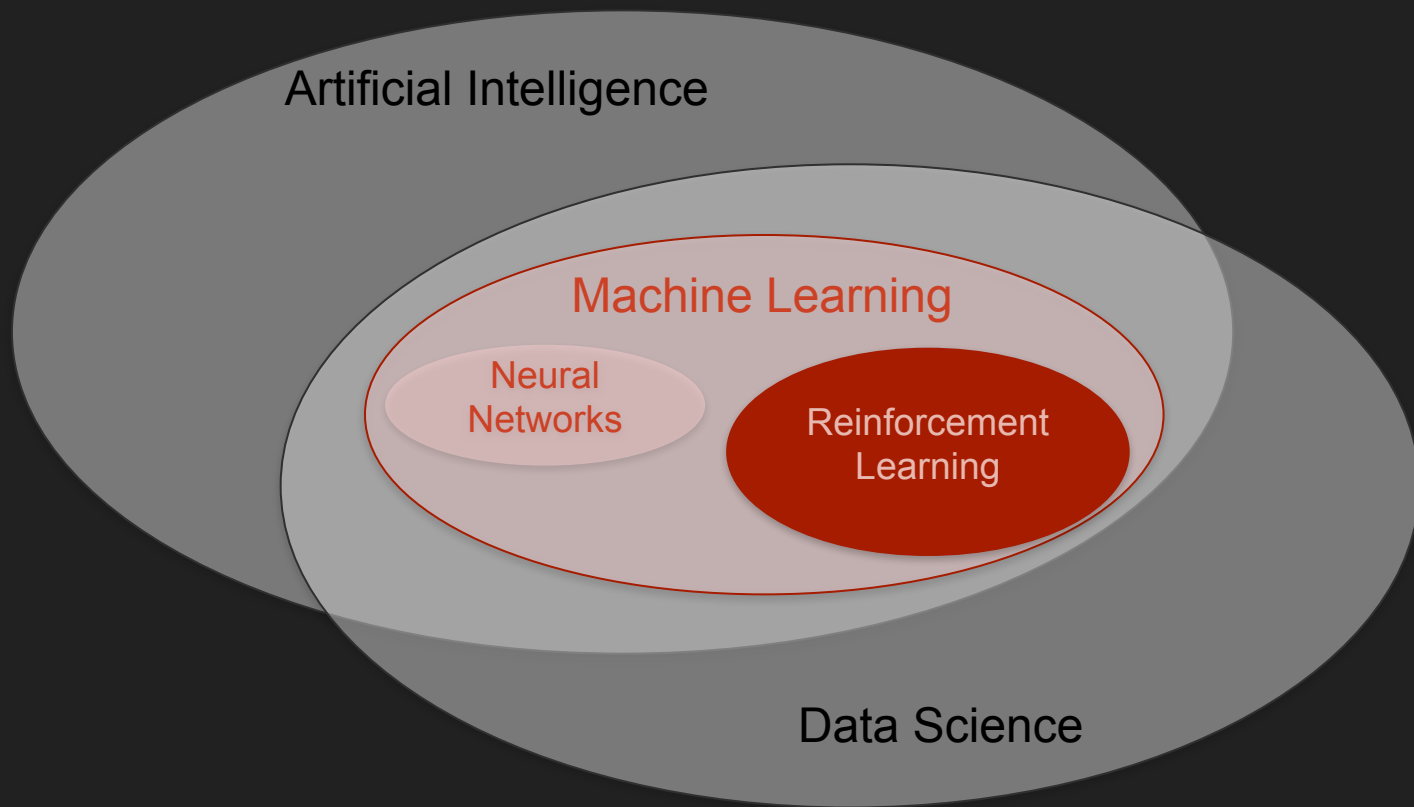
Knowledge Modelling

Statistics

Linear Regression

Data Science





# Robotii Elmer and Elsie (1948)

- 2 roboti electronici construiți materiale extra din război și mecanisme de ceas **programați sa gandeasca**
- **Un senzor de lumina**
- Programați sa evite obstacole din exterior utilizand senzorii de care dispuneau

În unul dintre experimente o lumina a fost plasata pe carapacea lui Elmer, moment în care robotul a început sa se miste sacadat, oamenii de stiinta au văzut asta ca pe un soi de entuziasm.

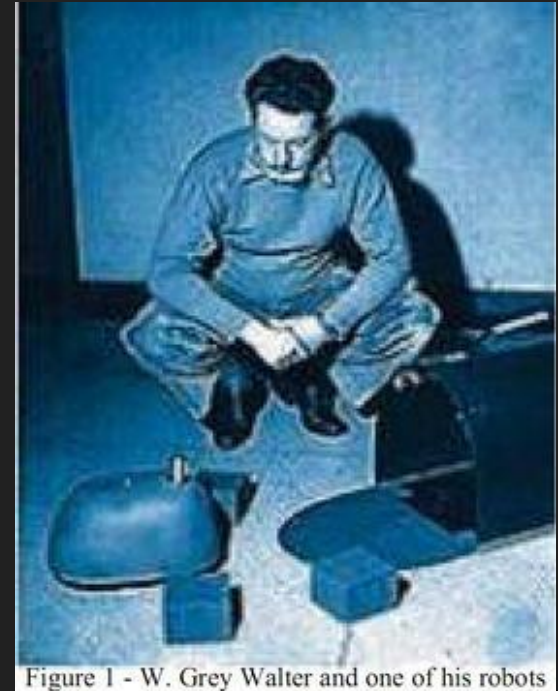
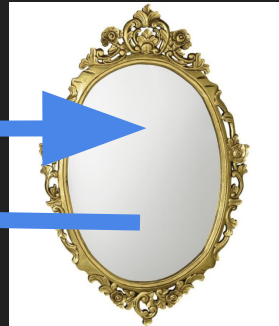
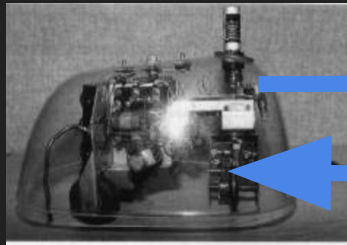


Figure 1 - W. Grey Walter and one of his robots

# Deep Blue vs. Kasparov (1996- 1997)

Deep Blue a fost un super-computer construit de IBM să joace șah.

The 1996 match

Game #	White	Black	Result	Method of conclusion
1	Deep Blue	Kasparov	1-0	Resignation
2	Kasparov	Deep Blue	1-0	Resignation
3	Deep Blue	Kasparov	½-½	Draw by mutual agreement
4	Kasparov	Deep Blue	½-½	Draw by mutual agreement
5	Deep Blue	Kasparov	0-1	Resignation
6	Kasparov	Deep Blue	1-0	Resignation

*Result: Kasparov-Deep Blue: 4-2*

The 1997 rematch

Game #	White	Black	Result	Method of conclusion
1	Kasparov	Deep Blue	1-0	Resignation
2	Deep Blue	Kasparov	1-0	Resignation
3	Kasparov	Deep Blue	½-½	Draw by mutual agreement
4	Deep Blue	Kasparov	½-½	Draw by mutual agreement
5	Kasparov	Deep Blue	½-½	Draw by mutual agreement
6	Deep Blue	Kasparov	1-0	Resignation

*Result: Deep Blue-Kasparov: 3½-2½*



Deep Blue  
IBM chess computer

Garry Kasparov  
World Chess Champion



# Deep Blue vs. Kasparov (1996- 1997)

Deep Blue a fost un super-computer construit de IBM să joace șah.

The 1996 match				
Game #	White	Black	Result	Method of conclusion
1	Deep Blue	Kasparov	1-0	Resignation
2	Kasparov	Deep Blue	1-0	Resignation
3	Deep Blue	Kasparov	½-½	Draw by mutual agreement
4	Kasparov	Deep Blue	½-½	Draw by mutual agreement
5	Deep Blue	Kasparov	0-1	Resignation
6	Kasparov	Deep Blue	1-0	Resignation
Result: Kasparov-Deep Blue: 4-2				

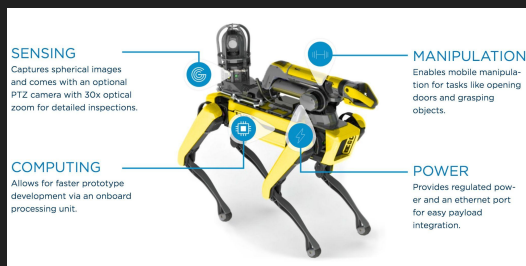
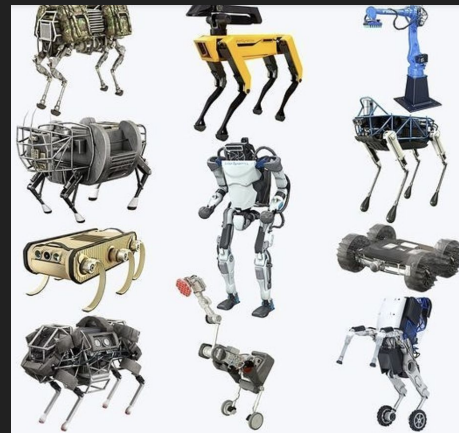
The 1997 rematch				
Game #	White	Black	Result	Method of conclusion
1	Kasparov	Deep Blue	1-0	Resignation
2	Deep Blue	Kasparov	1-0	Resignation
3	Kasparov	Deep Blue	½-½	Draw by mutual agreement
4	Deep Blue	Kasparov	½-½	Draw by mutual agreement
5	Kasparov	Deep Blue	½-½	Draw by mutual agreement
6	Deep Blue	Kasparov	1-0	Resignation
Result: Deep Blue-Kasparov: 3½-2½				



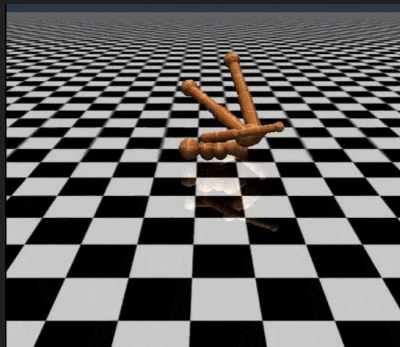
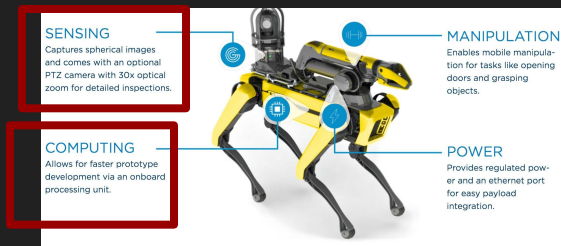
În al doilea și ultimul joc într-o pozitie de final, Deep Blue a ales o mișcări “atipice” unui calculator după spusele campionului, care nu urmăreau un **castig concret**. Codul care guverna mișcărilor lui Deep Blue a fost sters iar computerul ars.

# Boston Dynamics Robots (1992- 2024)

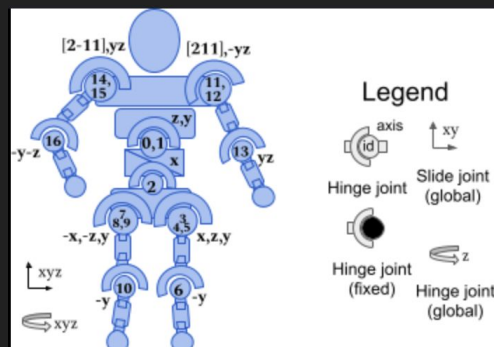
- Extrem de agili, navighează pe terenuri variate și se adaptează la obstacole.
- Utilizați în industrie pentru inspecții, supraveghere și transport, demonstrând aplicații reale ale AI și învățării prin întărire.
- Deschid uși și urcă scări, executând sarcini complexe și transformând interacțiunea cu mediul.



# MuJoCo - DeepMind (2012)



# Antrenare



Ce am observat?

# Agent



# Mediu



Agentul observa Mediul și poziția sa în mediu

Agent



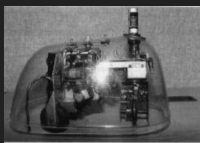
Mediu



Agent

Agentul observa Mediul și poziția sa în mediu

Mediu



Muta piciorul înainte

Ce7

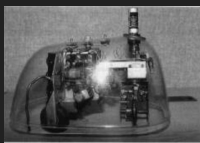
Porneste motorul A cu viteza X



Agent

Agentul observa Mediul și poziția sa în mediu

Mediu



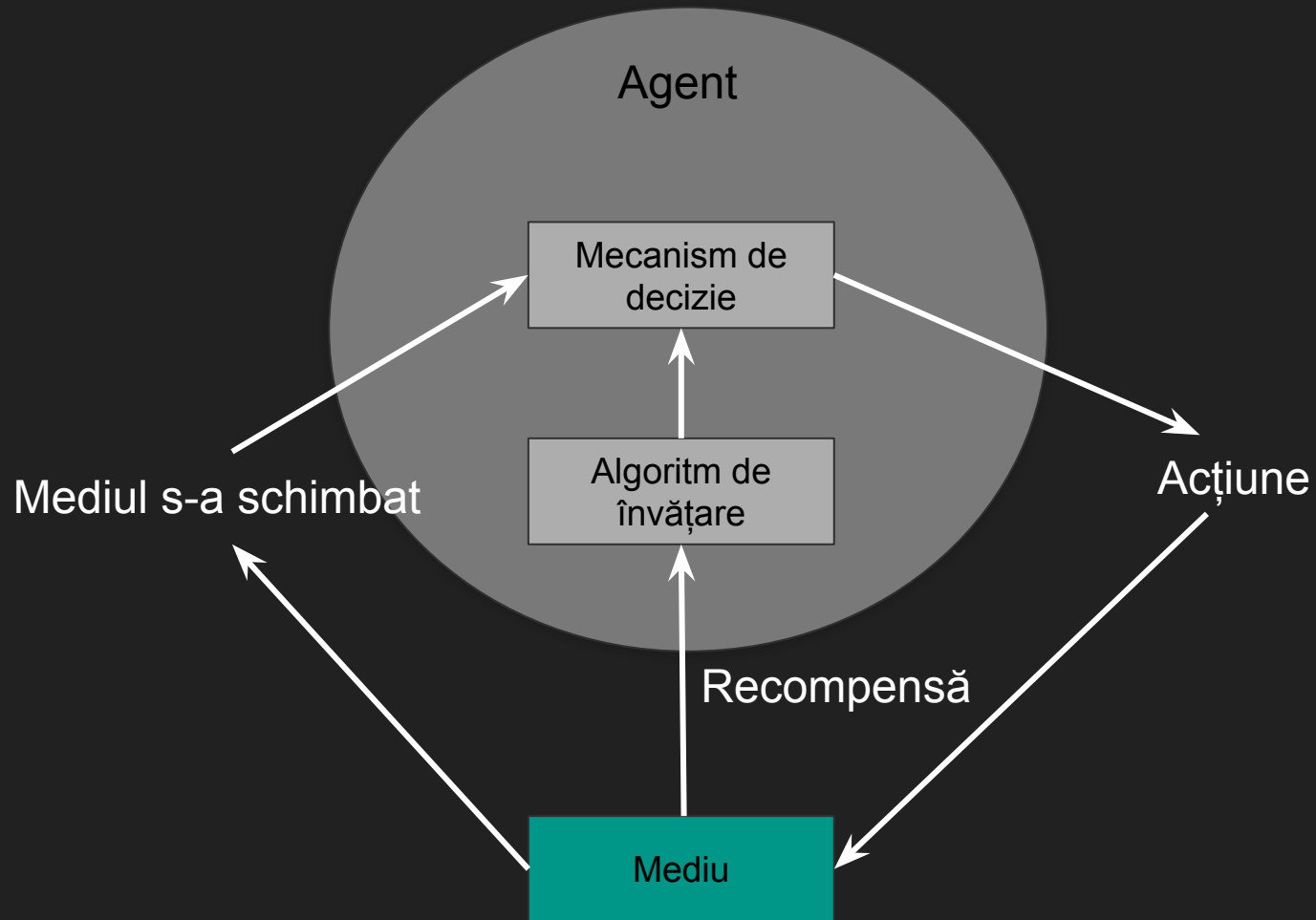
Omulețul nu mai este în picioare

Antrenează  
mecanismul de  
luat decizii

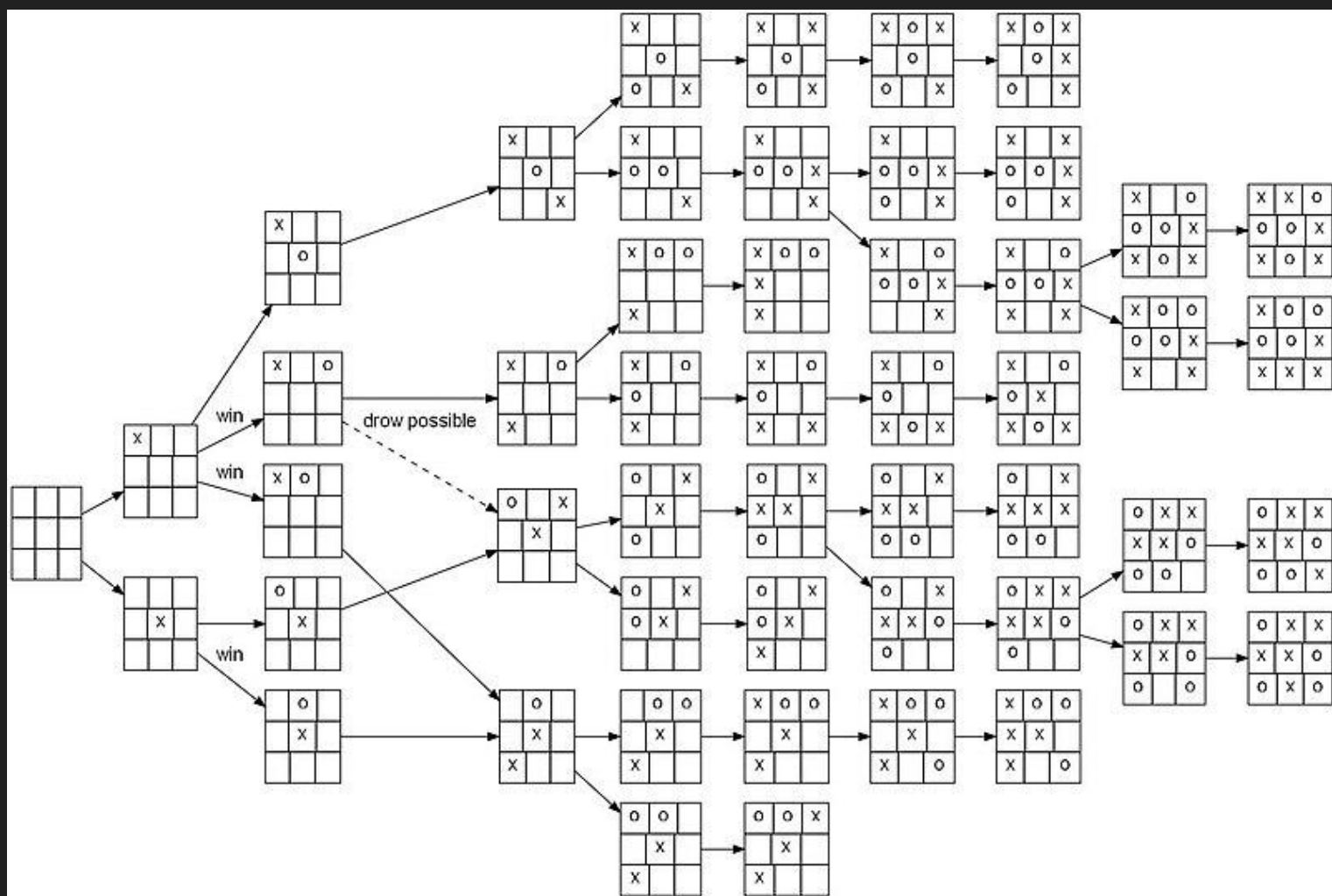
Mecanism  
de luat  
decizii

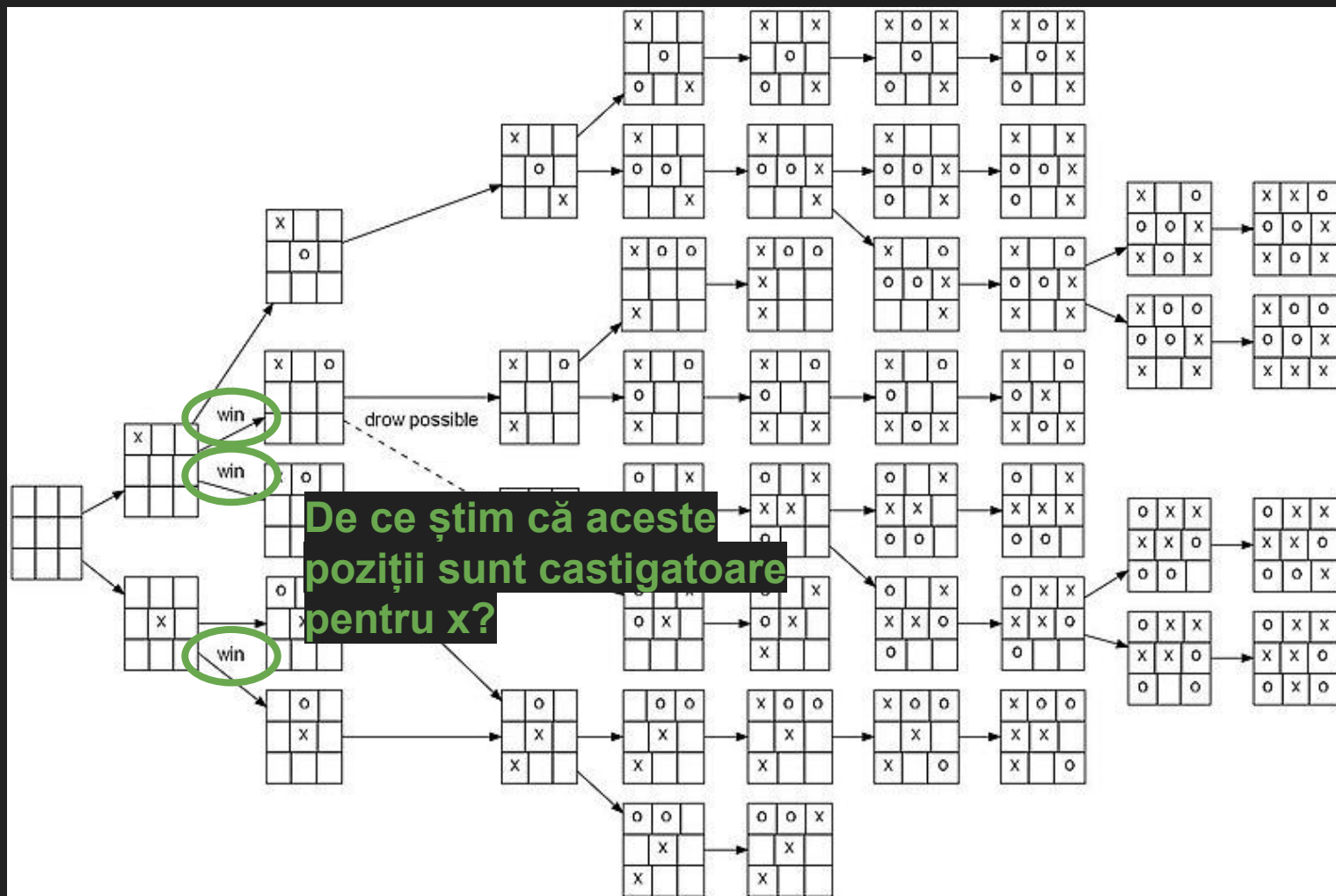


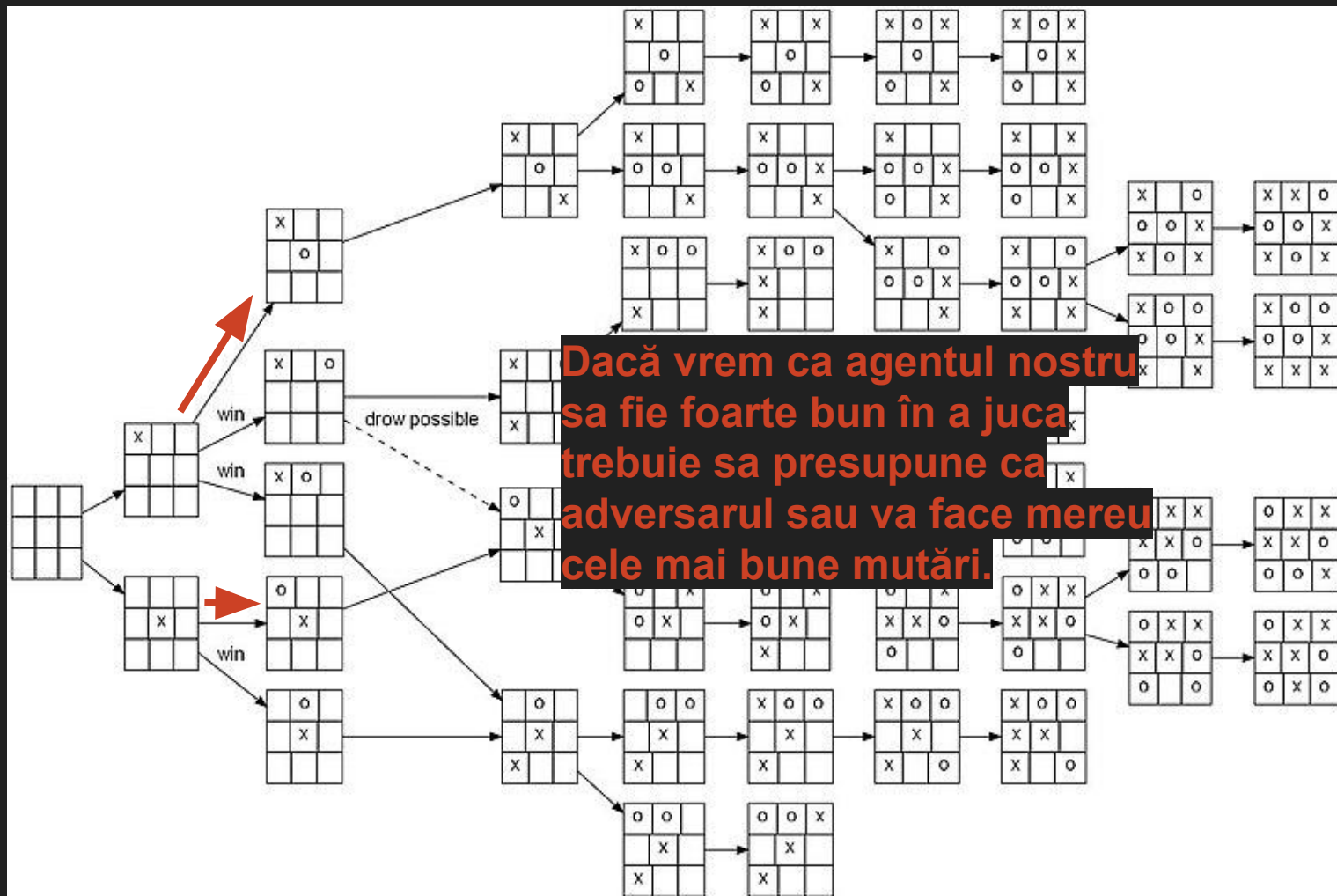




Cum construim sisteme  
inteligente?








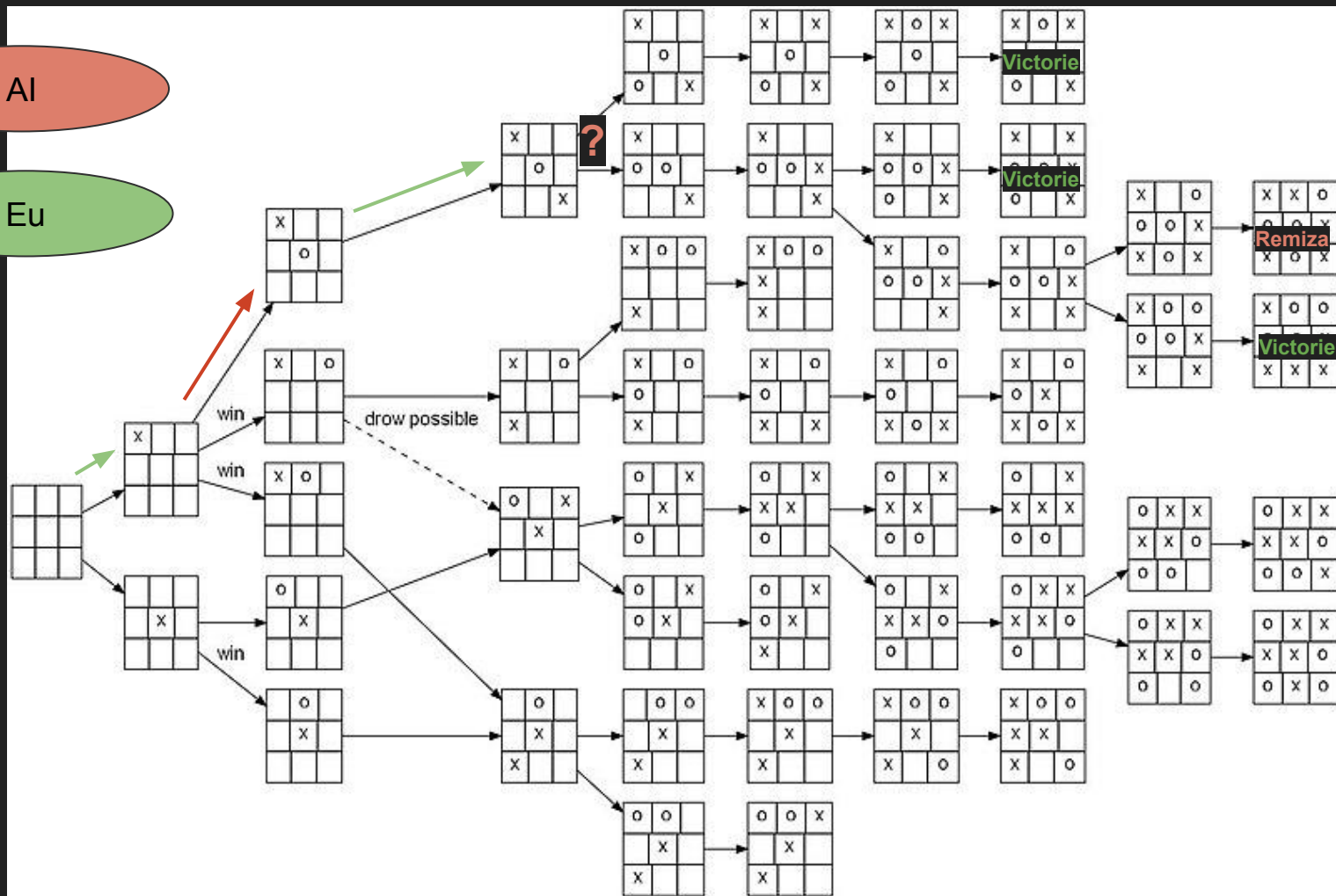
Hai sa simulăm un joc!




AI




Eu

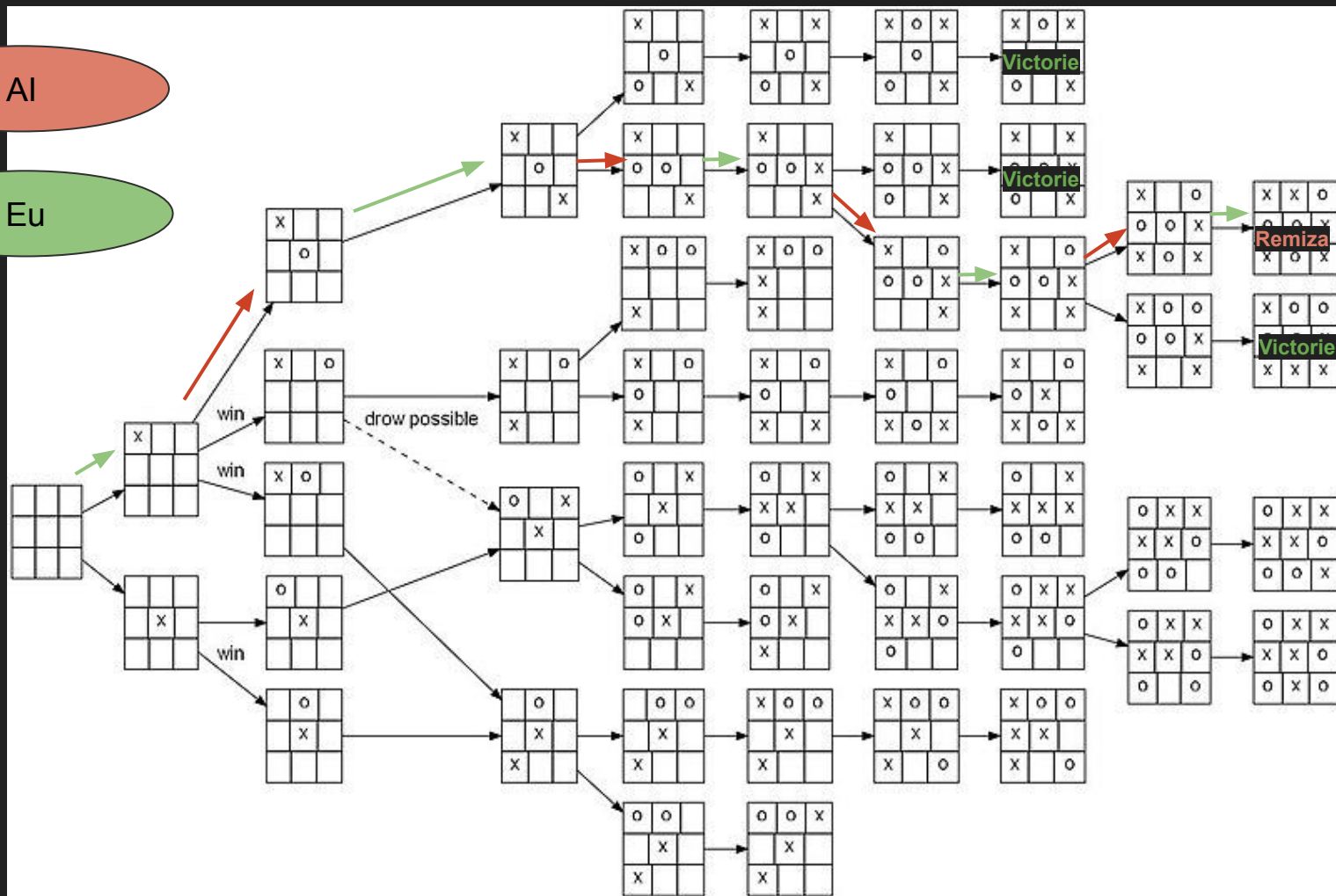




AI



Eu





1.

Ce se întâmplă în  
momentul în care nu ne  
putem uita suficient de  
adânc în viitor?

Observații:

2.

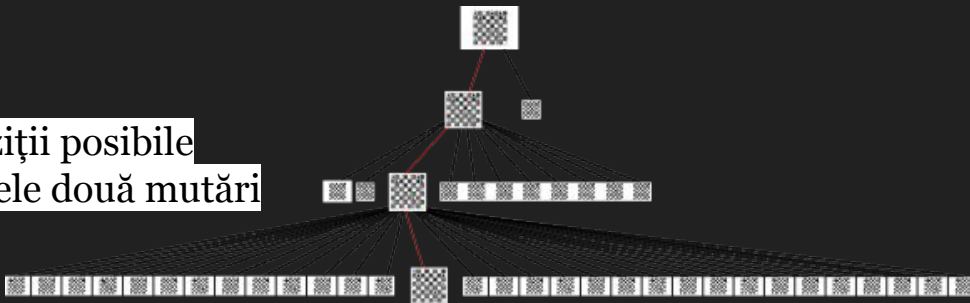
AI

Eu

Algoritmul  
Mini-Max

# Poziții într-un joc de șah.

400 de poziții posibile  
după primele două mutări



Cum putem evalua ajută AI-ul să ia decizii mai bune dacă nu putem vedea finalul jocului?

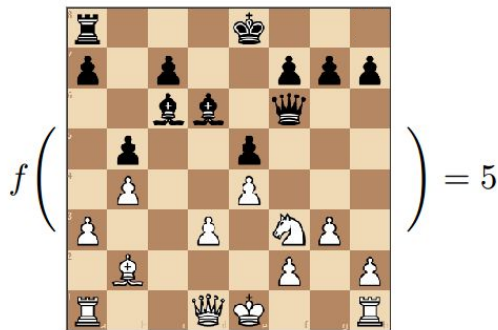
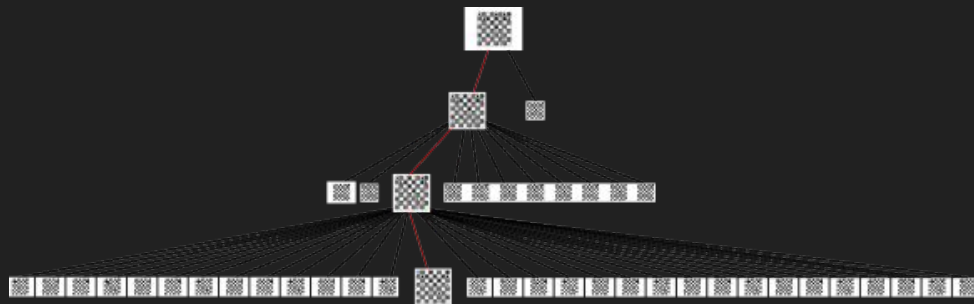
...

...

10<sup>111</sup> pozitív

# Poziții într-un joc de șah.

Cum putem evalua ajută AI-ul  
să ia decizii mai bune dacă nu  
putem vedea finalul jocului?



$$f\left(\begin{array}{c} \text{Chessboard Position} \end{array}\right) = 5$$

Pieces and Point Value		
Pawn		1
Knight		3
Bishop		3
Rook		5
Queen		9
King		priceless

...

...

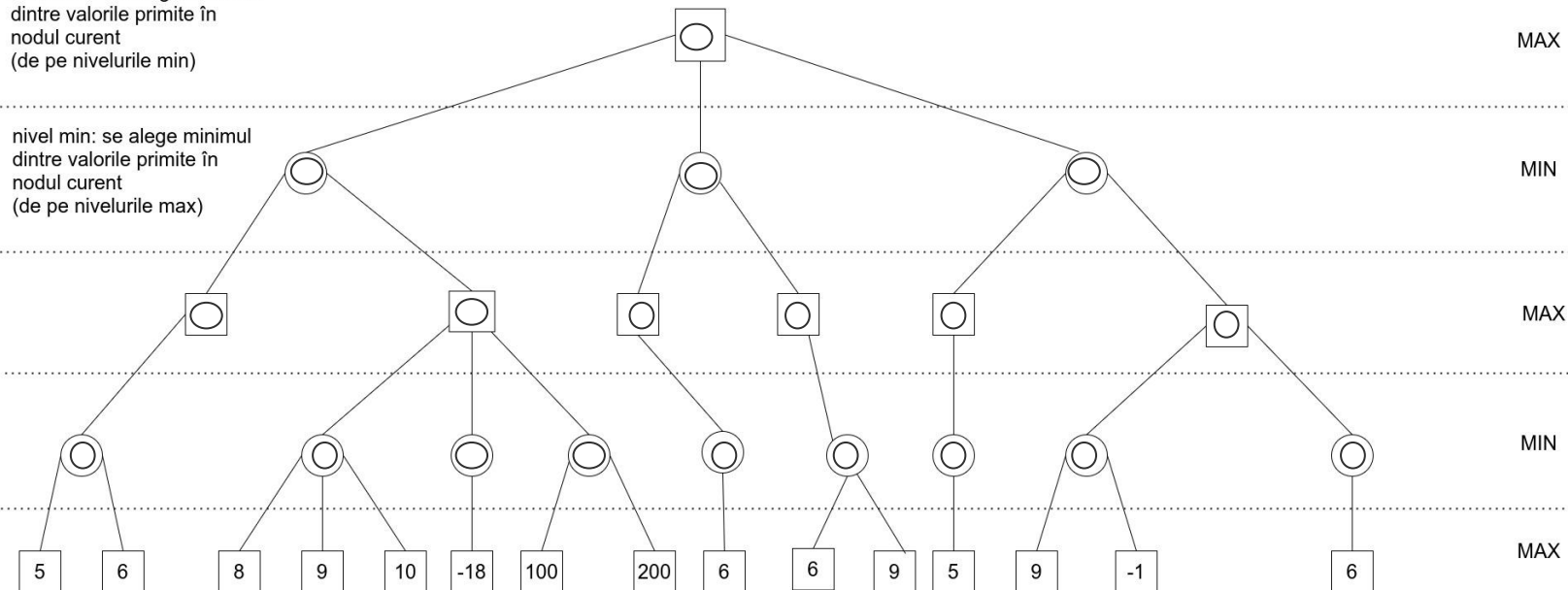
$10^{11}$  pozitii

# Algoritmul MiniMax

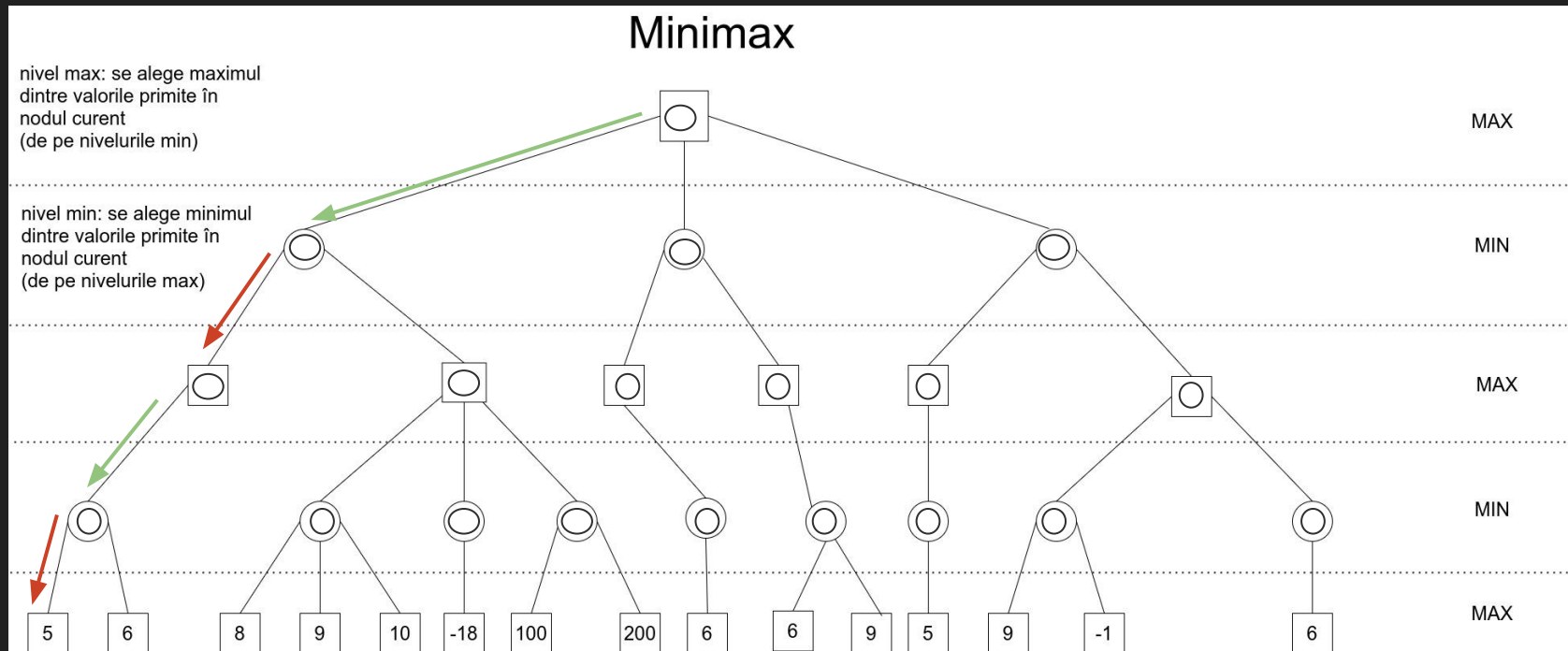
## Minimax

nivel max: se alege maximul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile min)

nivel min: se alege minimul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile max)



# Algoritmul MiniMax

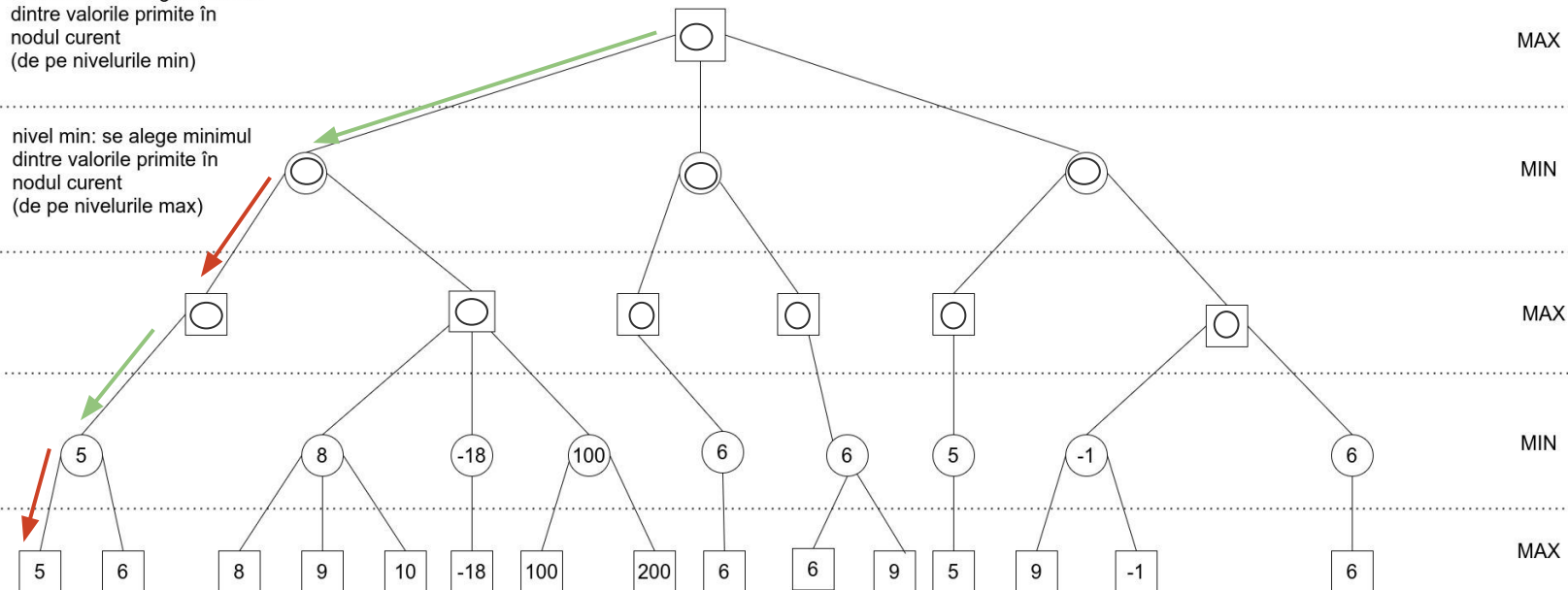


# Algoritmul MiniMax

## Minimax

nivel max: se alege maximul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile min)

nivel min: se alege minimul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile max)

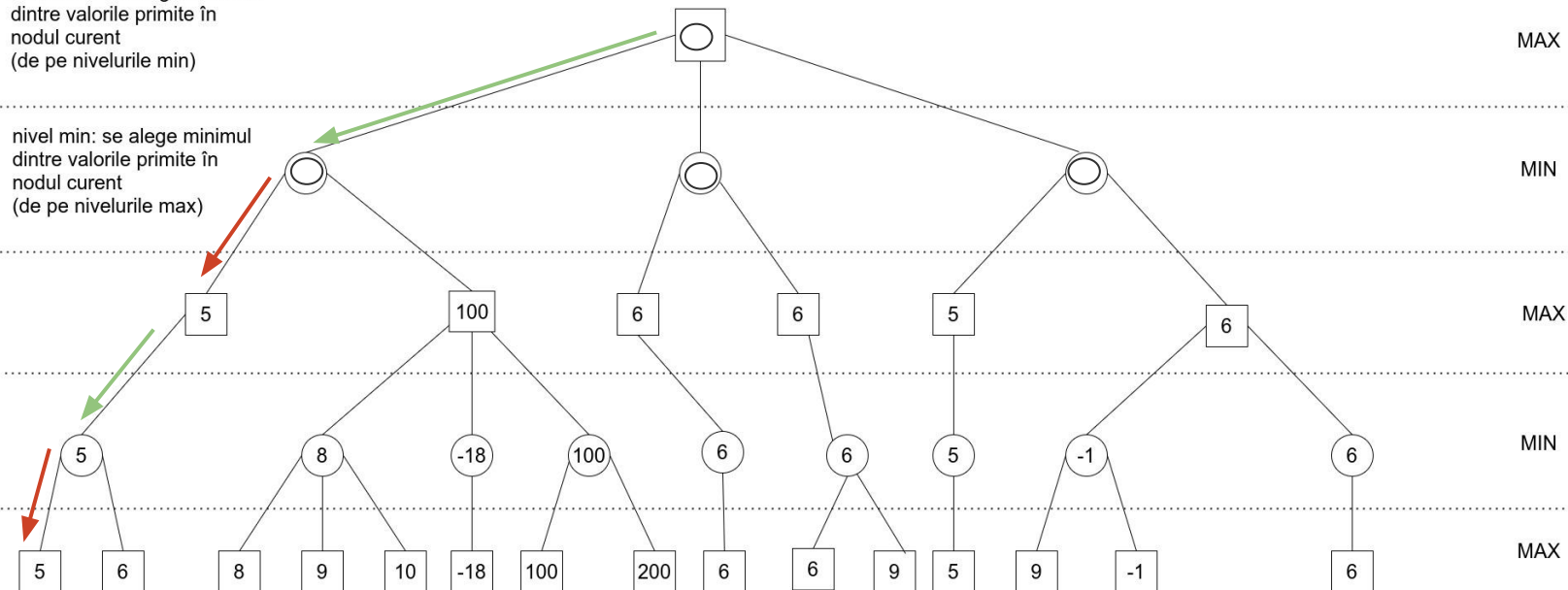


# Algoritmul MiniMax

## Minimax

nivel max: se alege maximul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile min)

nivel min: se alege minimul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile max)

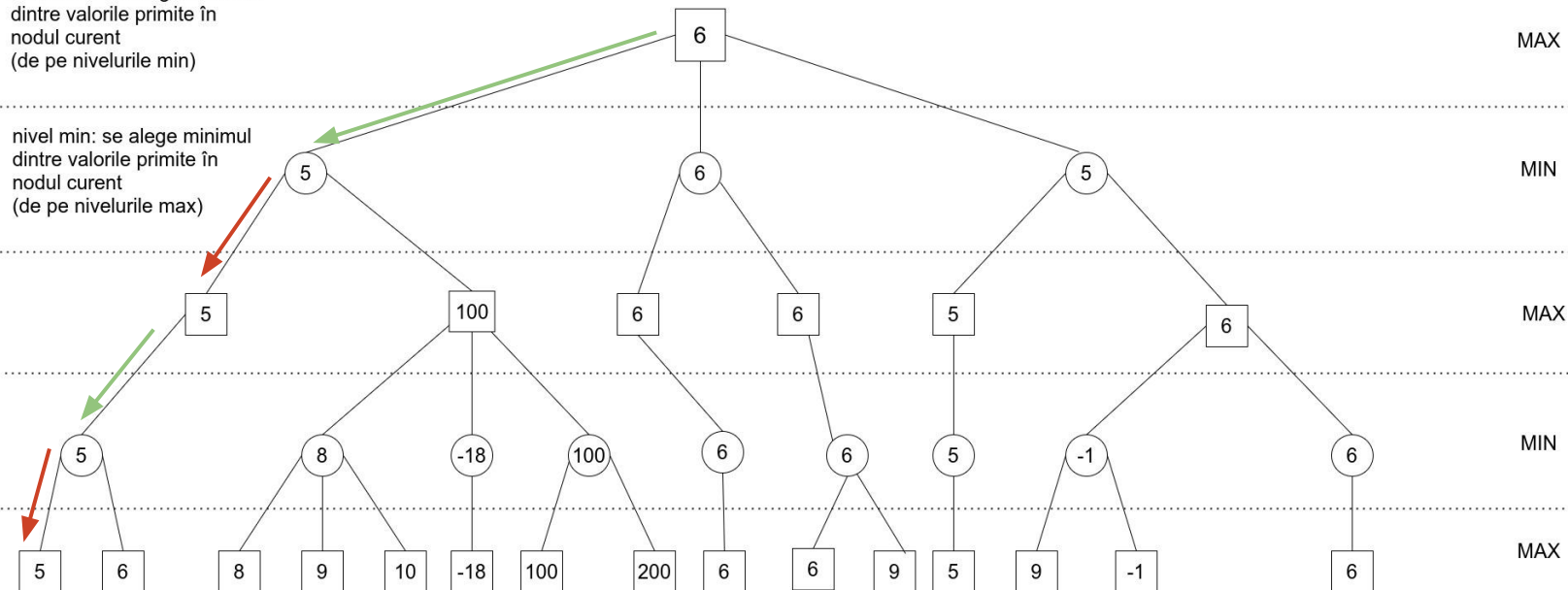


# Algoritmul MiniMax

## Minimax

nivel max: se alege maximul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile min)

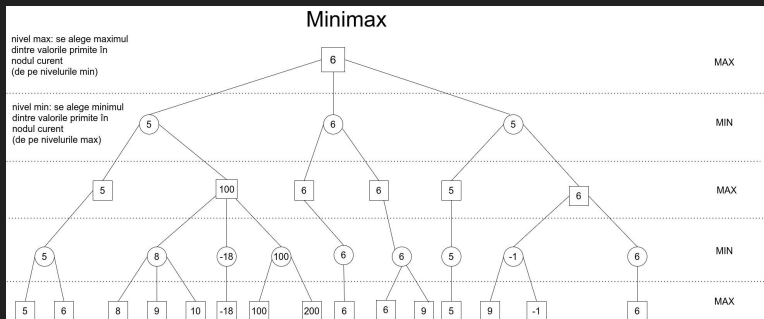
nivel min: se alege minimul  
dintre valorile primite în  
nodul curent  
(de pe nivelurile max)





# Algoritmul

## MiniMax



Minimax(node, depth, is\_maximizing\_player)

IF depth > 3 OR node is terminal

RETURN the heuristic value of node

IF is\_maximizing\_player

best\_value =  $-\infty$

FOR each child of node

value = Minimax(child, depth + 1, FALSE)

best\_value = max(best\_value, value)

RETURN best\_value

ELSE

best\_value =  $\infty$

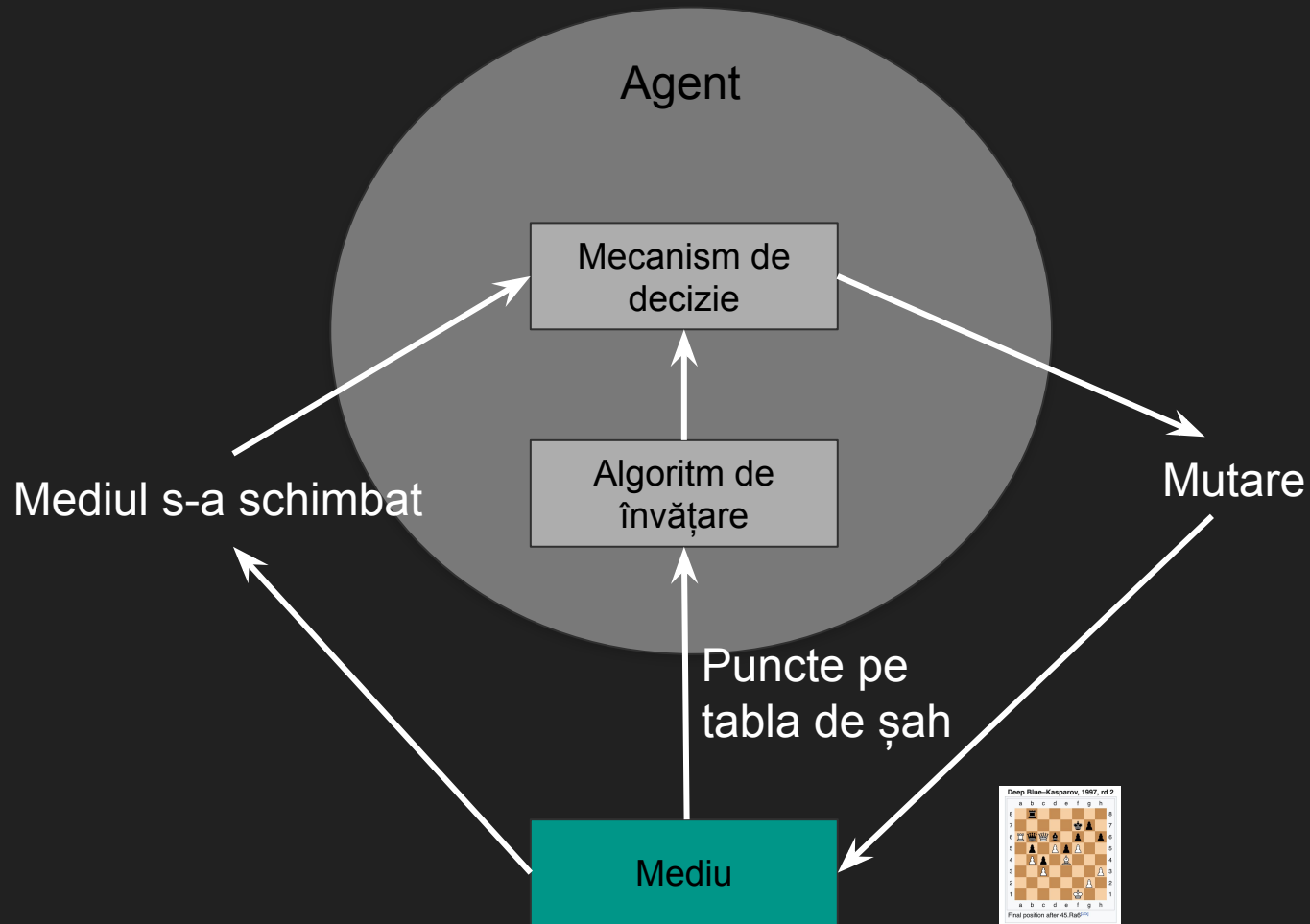
FOR each child of node

value = Minimax(child, depth + 1, TRUE)

best\_value = min(best\_value, value)

RETURN best\_value

Ce am învățat astăzi?



Link: [https://github.com/TeoReu/VO\\_RL](https://github.com/TeoReu/VO_RL)