



IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille



MinMax

Monte Carlo Tree Search

Alpha Zero

17/09/2021



IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Un peu d'histoire

#IMTomorrow

#IMTNordEurope



IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Un peu d'histoire

1997 : Deep Blue développé par IBM bat le champion du monde Garry Kasparov aux échecs.



2016: AlphaGo développé par DeepMind bat Lee Sedol (joueur de 9^e dam) au Go



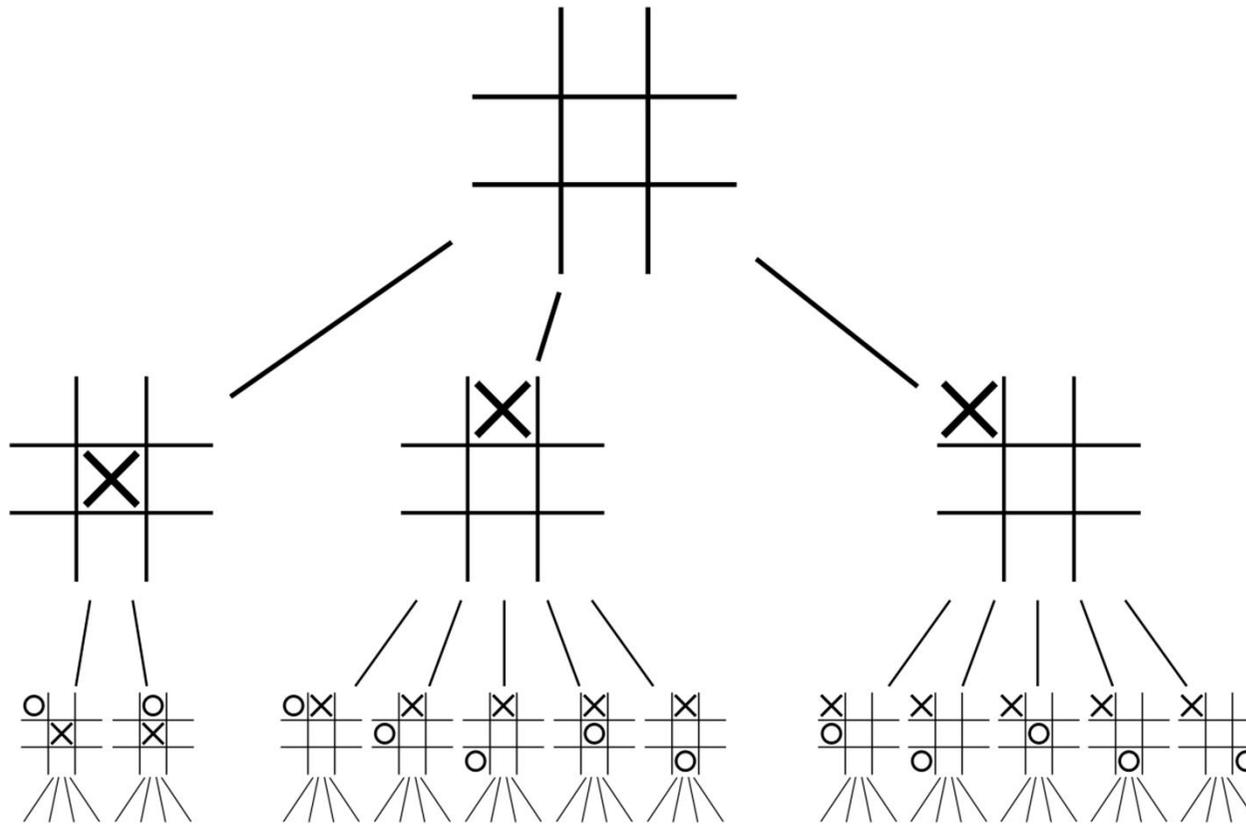


IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

La problématique

#IMTomorrow

#IMTNordEurope





➤ Aux échecs

- il y a en moyenne 30 coups possibles
- Il y a donc 30^n positions à analyser à la profondeur n
- Une recherche à la profondeur 16 (chaque joueur joue 8 coups) nécessiterait d'analyser $30^{16} = 430\ 467\ 210\ 000\ 000\ 000\ 000\ 000$ positions
- L'ordinateur le plus rapide effectuant 120×10^{15} opérations par secondes aurait besoin de plus de 40 jours pour jouer un coup.

➤ Au Go

- La taille de l'arbre est estimée à 10^{600} .

➤ Certains jeux sont résolus

- Le jeu de dame anglaises
- Le puissance 4.



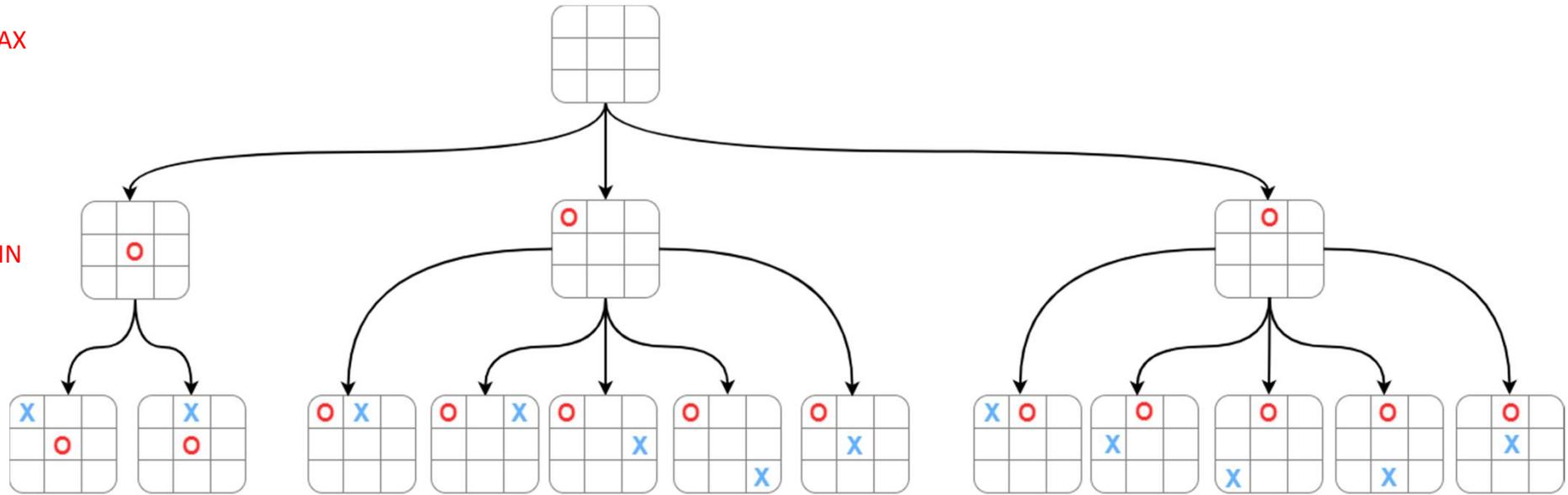
IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

MinMax

Joueur 1 : MAX



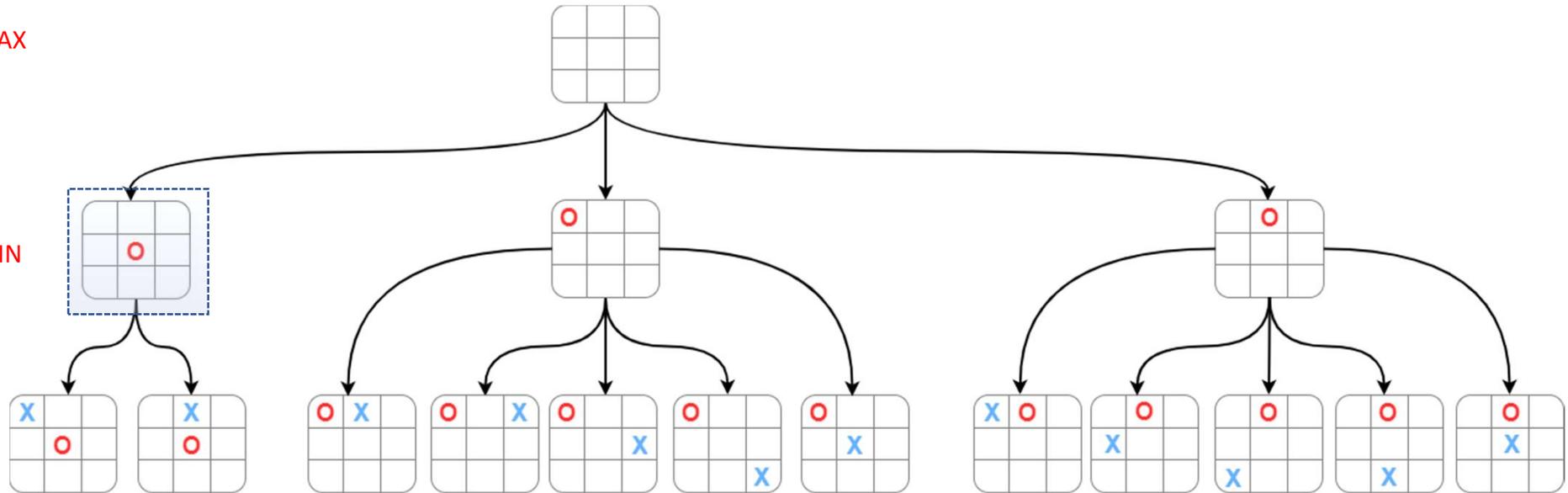
Joueur 2 : MIN



Joueur 1 : MAX



Joueur 2 : MIN

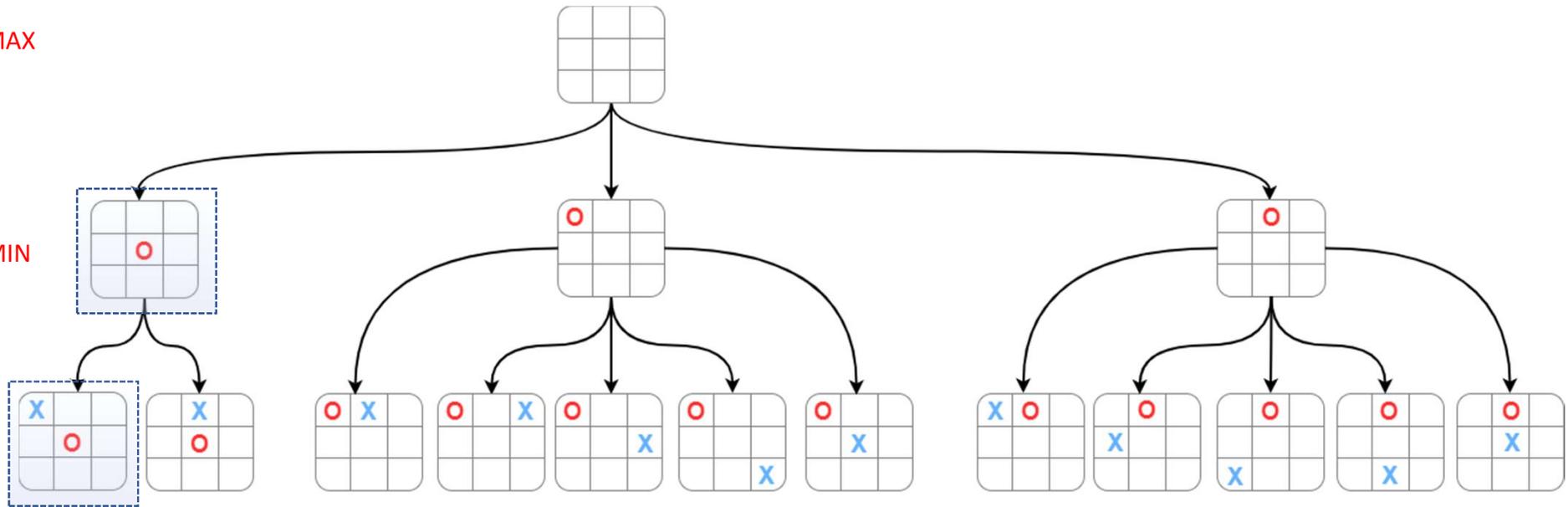


L'IA joue le premier coup pour le joueur 1.

Joueur 1 : MAX



Joueur 2 : MIN

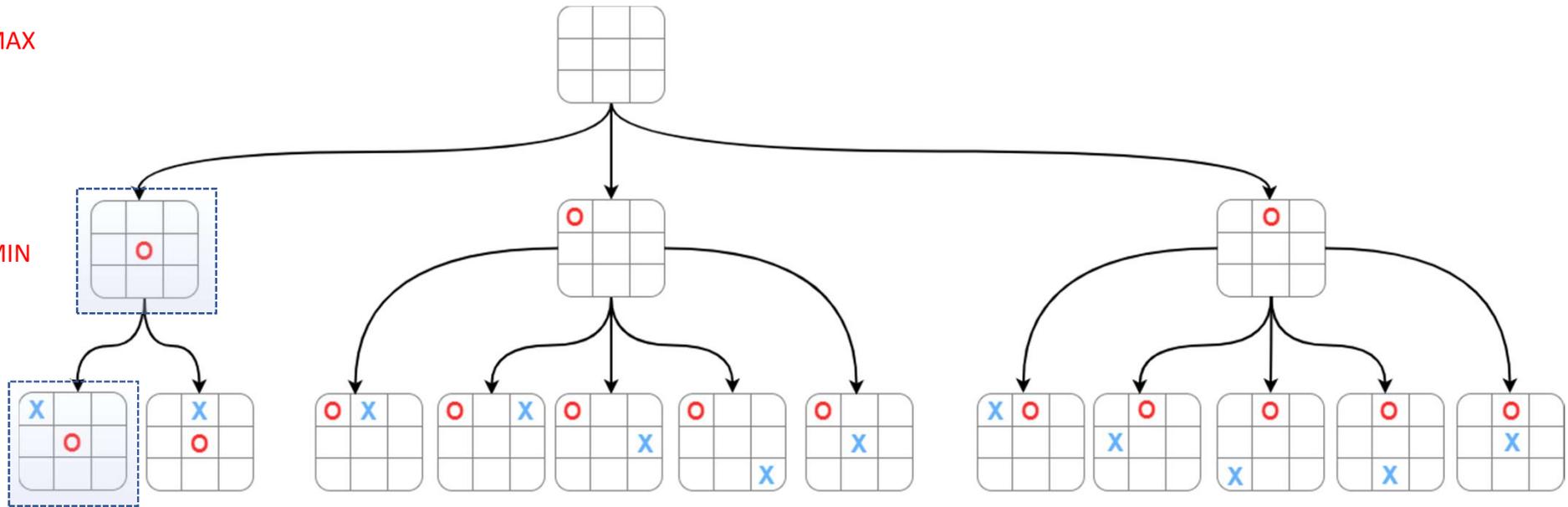


L'IA joue un coup possible pour le joueur 2.

Joueur 1 : MAX



Joueur 2 : MIN



On fixe une profondeur maximale de recherche. Dans notre exemple elle est de 2.
A ce stade on utilise une fonction d'évaluation afin d'obtenir une note de la position.

Comment évaluer cette position au jeu d'échecs ?



Fonction d'évaluation simple pour le jeu d'échecs

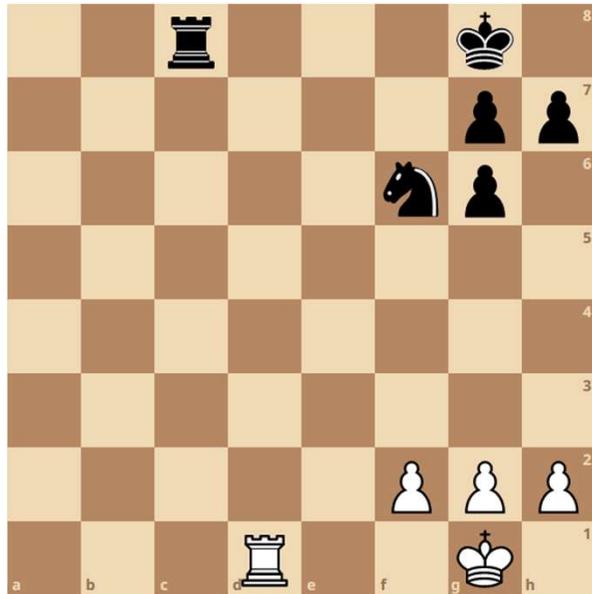
Pièce	Valeur
Dame	10
Tour	5
Fou	3
Cavalier	3
Pion	1

Associer un score à chaque pièce

Evaluation (Position) = Score matériel (Blanc) – Score matériel (Noir)

Les blancs vont maximiser cette évaluation alors que les noirs la minimiserons.

Fonction d'évaluation simple pour le jeu d'échecs



Evaluation : -3

Pièces	Blancs	Noirs
Tour	5 points	5 points
Pion	3 points	3 points
Cavalier		3 points
Total	8	11
Evaluation de la position d'un point de vue des blancs	-3	

Fonction d'évaluation un peu plus élaborée pour le jeu d'échecs



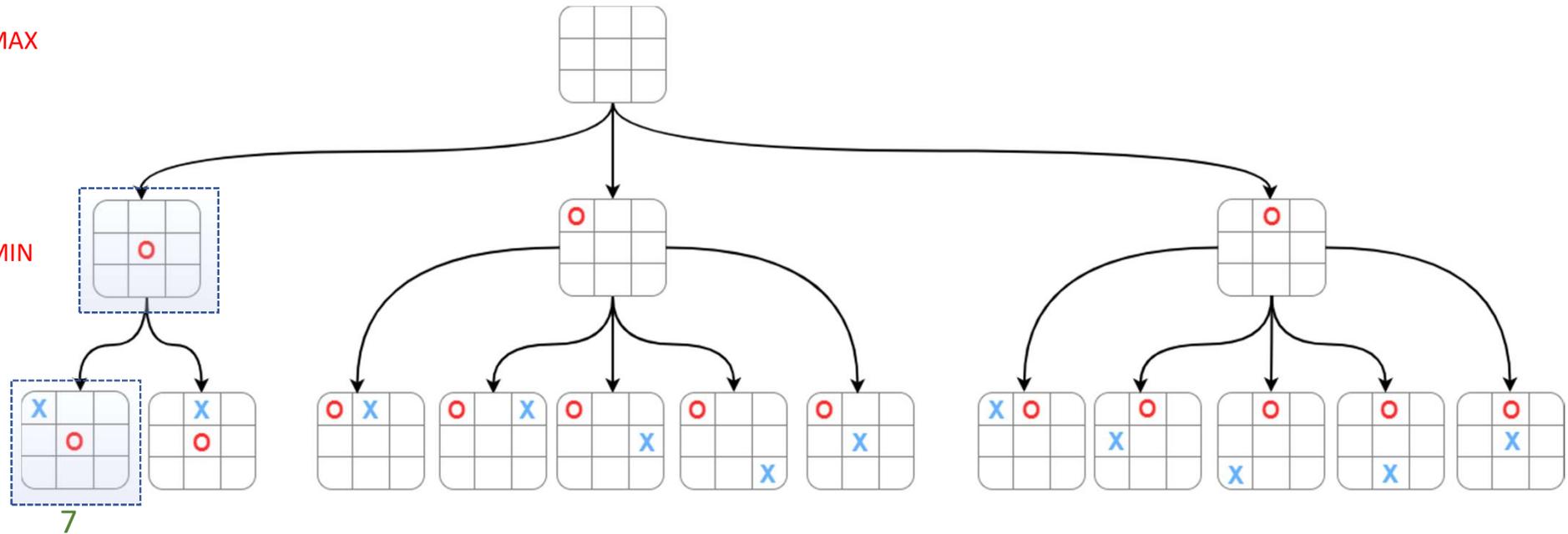
Evaluation : -2,5

Critères d'évaluation	Blancs	Noirs	Score Blancs – Noirs
Matériel	8	11	-3
Structure des pions. Un malus de 0,2 pour chaque pion situé sur la même colonne.	0	-0,2	0,2
Sécurité du roi. Un malus de 0,3 pour chaque pion absent devant le roque	0	-0,3	0,3
		Total :	-2,5

Joueur 1 : MAX



Joueur 2 : MIN



La fonction d'évaluation donne un score d'un point de vue du joueur 1. La valeur retournée par cette fonction d'évaluation sera d'autant plus grande que la position est bonne pour le Joueur 1. Le joueur 1 tente donc de maximiser cette valeur alors que le joueur 2 tente de la minimiser.

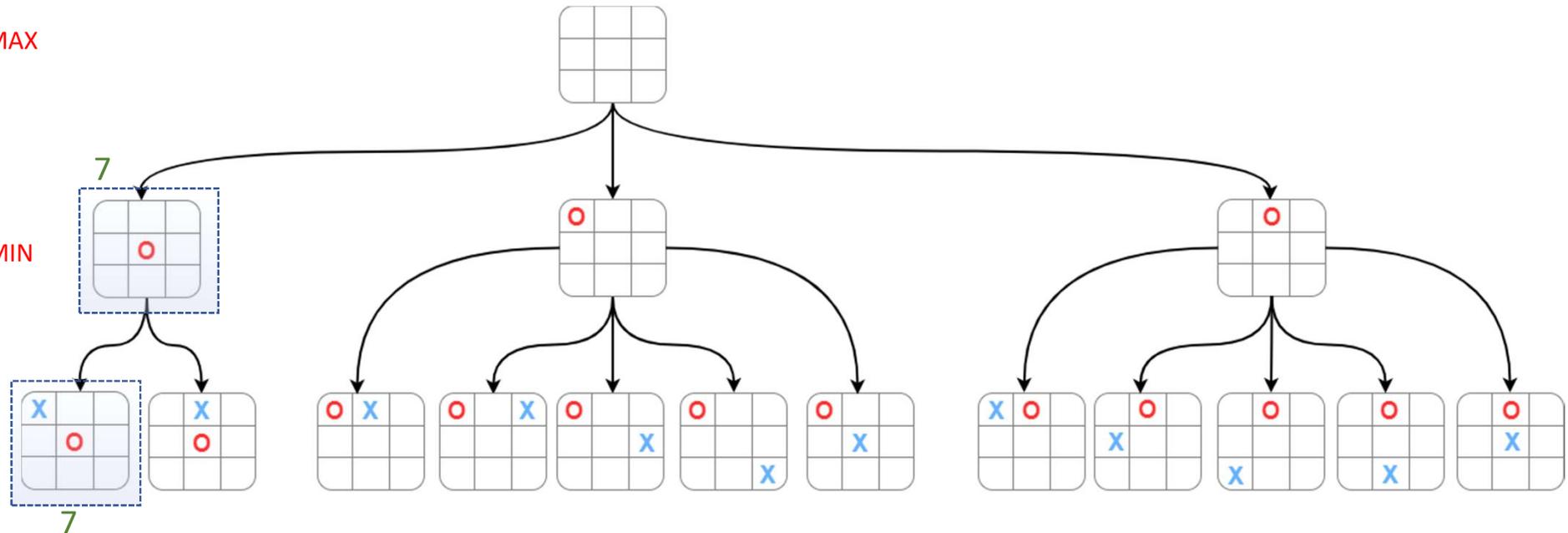
Joueur 1 : MAX



Joueur 2 : MIN



Evaluation

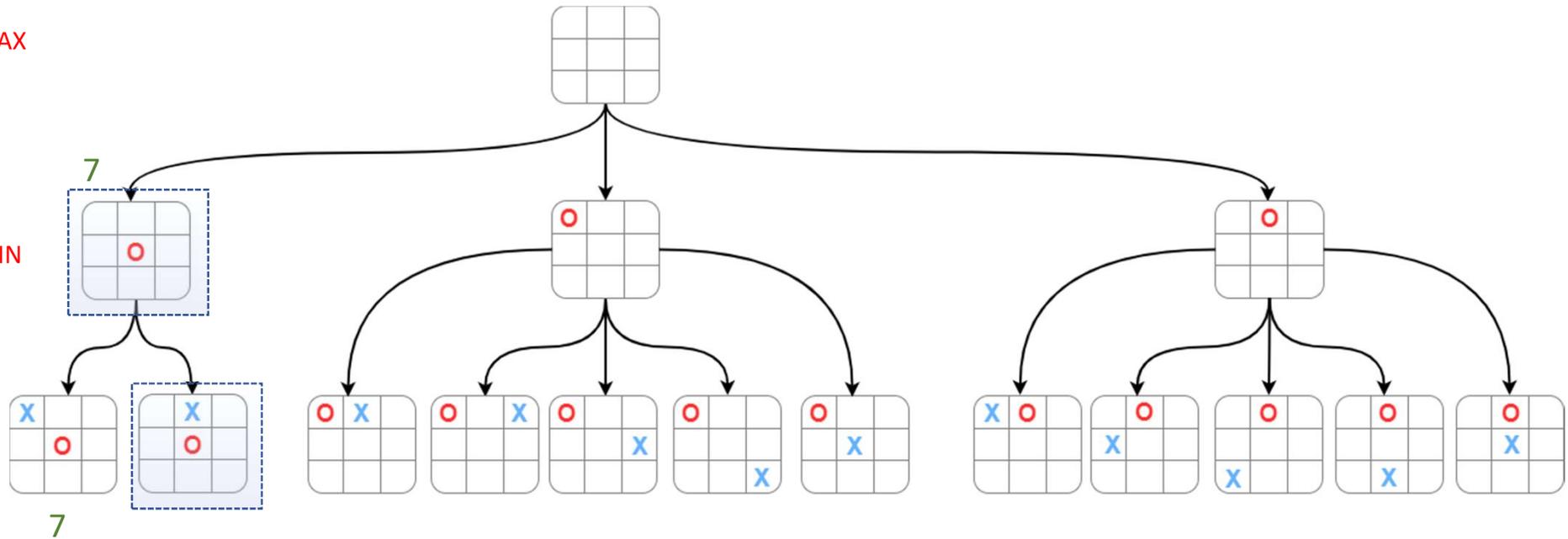


Le Joueur 2 minimise cette valeur. Son score est donc pour l'instant de 7.

Joueur 1 : **MAX**



Joueur 2 : **MIN**



Evaluation

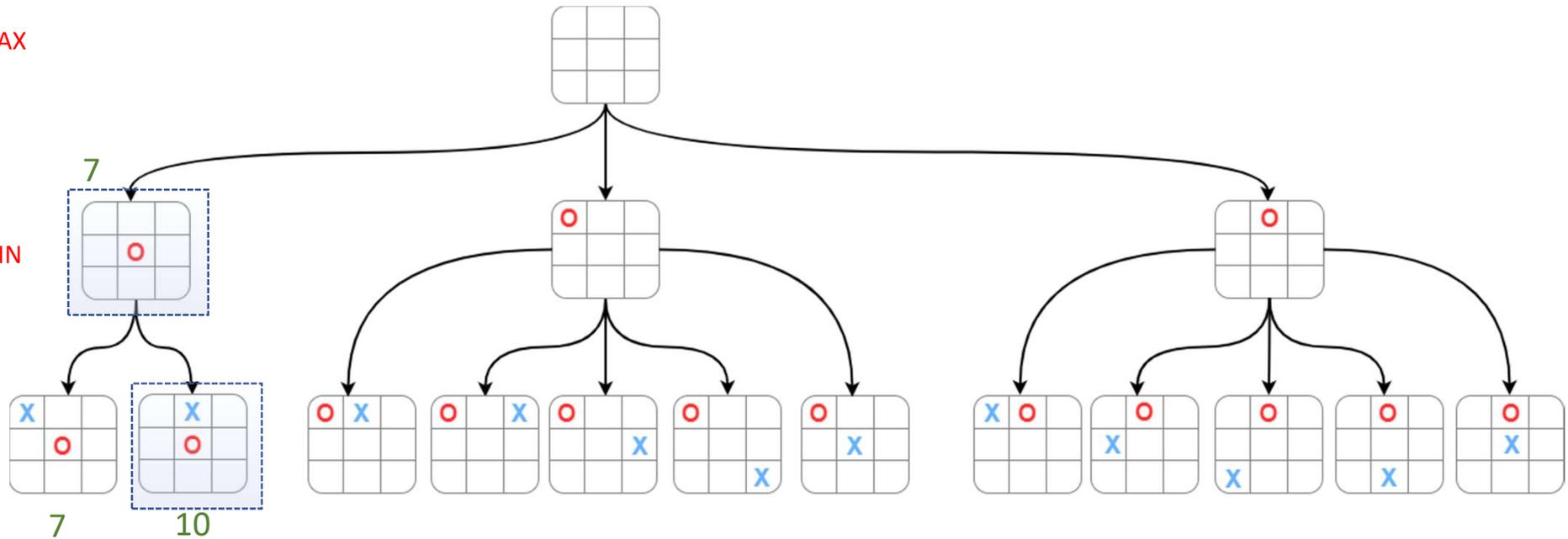
7

L'IA joue ensuite le second coup possible.

Joueur 1 : MAX



Joueur 2 : MIN

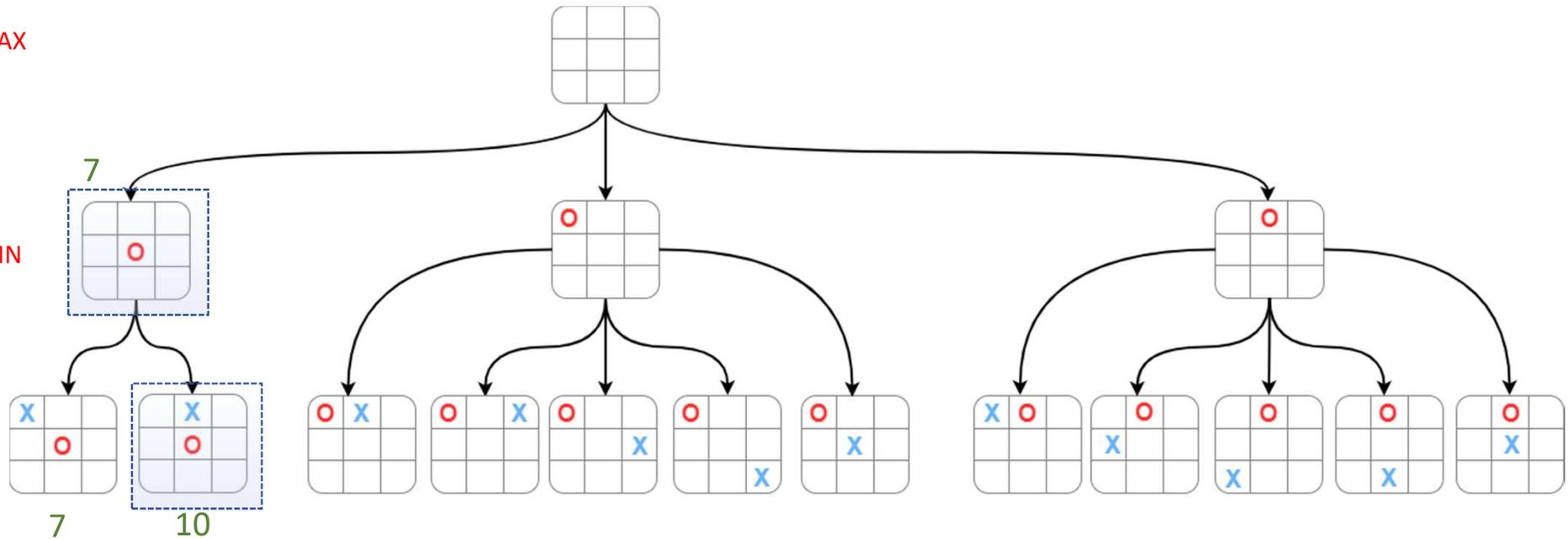


On évalue la position. Le score est ici de 10.

Joueur 1 : MAX



Joueur 2 : MIN



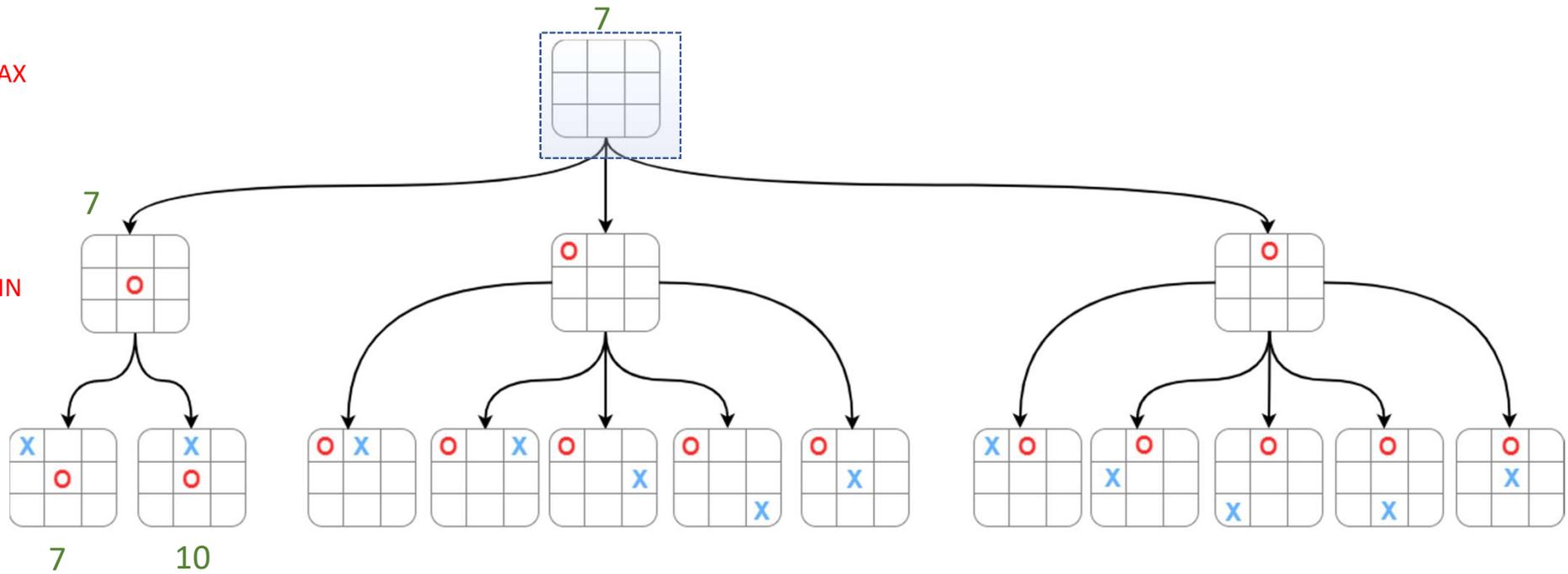
Evaluation

Le Joueur 2 compare son score actuel de 7 avec le score de 10. Comme il minimise le score, son score ne change pas.

Joueur 1 : MAX



Joueur 2 : MIN

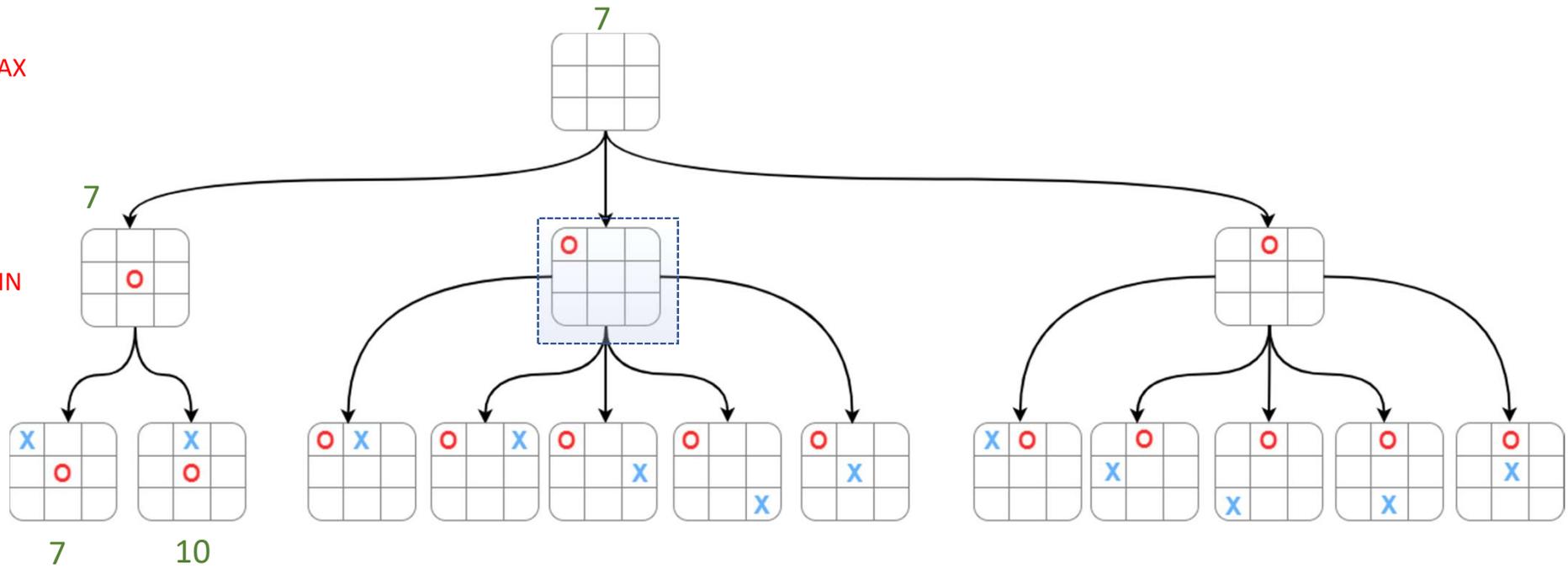


Le score du joueur 1 passe à 7.

Joueur 1 : MAX



Joueur 2 : MIN

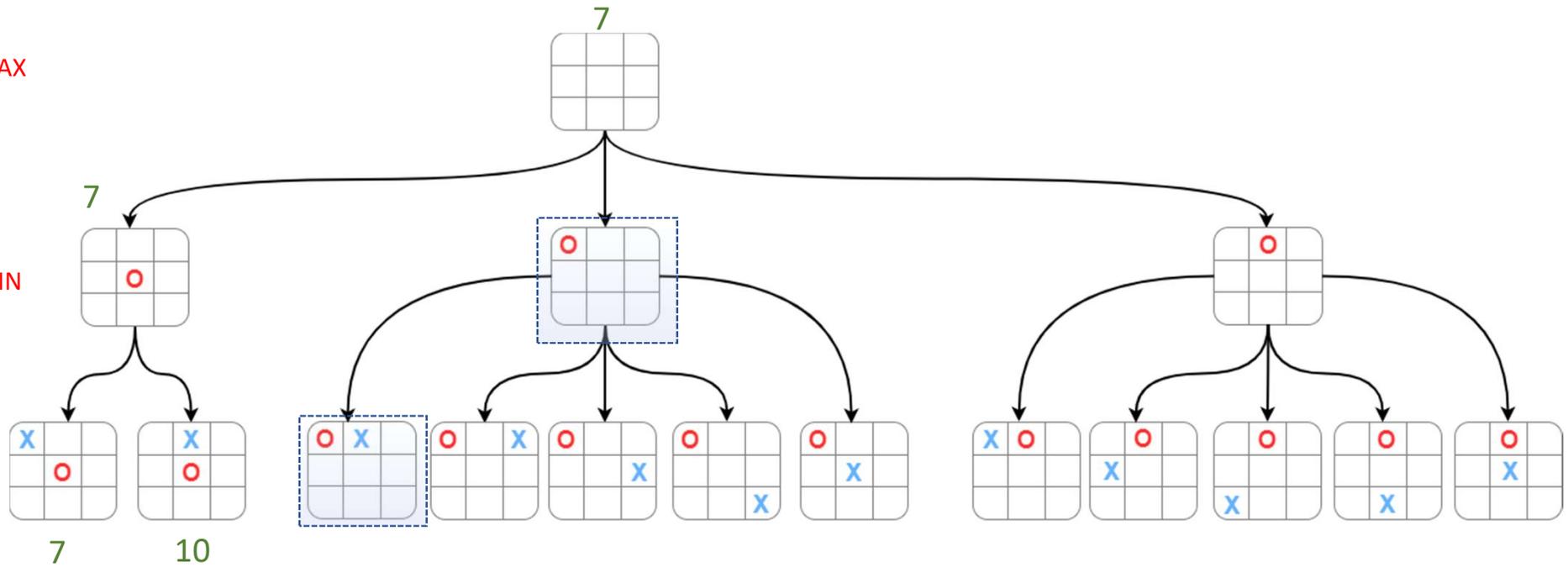


L'IA joue ensuite les autres coups possibles pour le joueur 1.

Joueur 1 : MAX



Joueur 2 : MIN

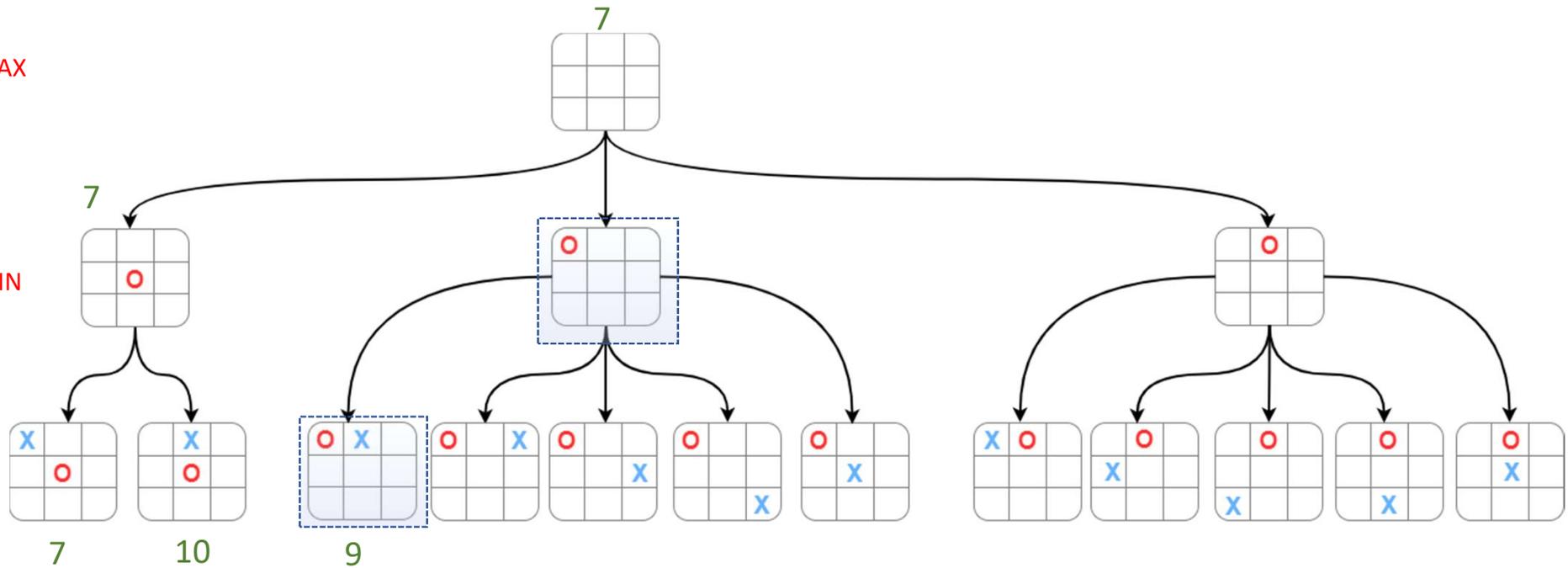


L'IA joue un premier coup possible pour le joueur 2.

Joueur 1 : **MAX**



Joueur 2 : **MIN**

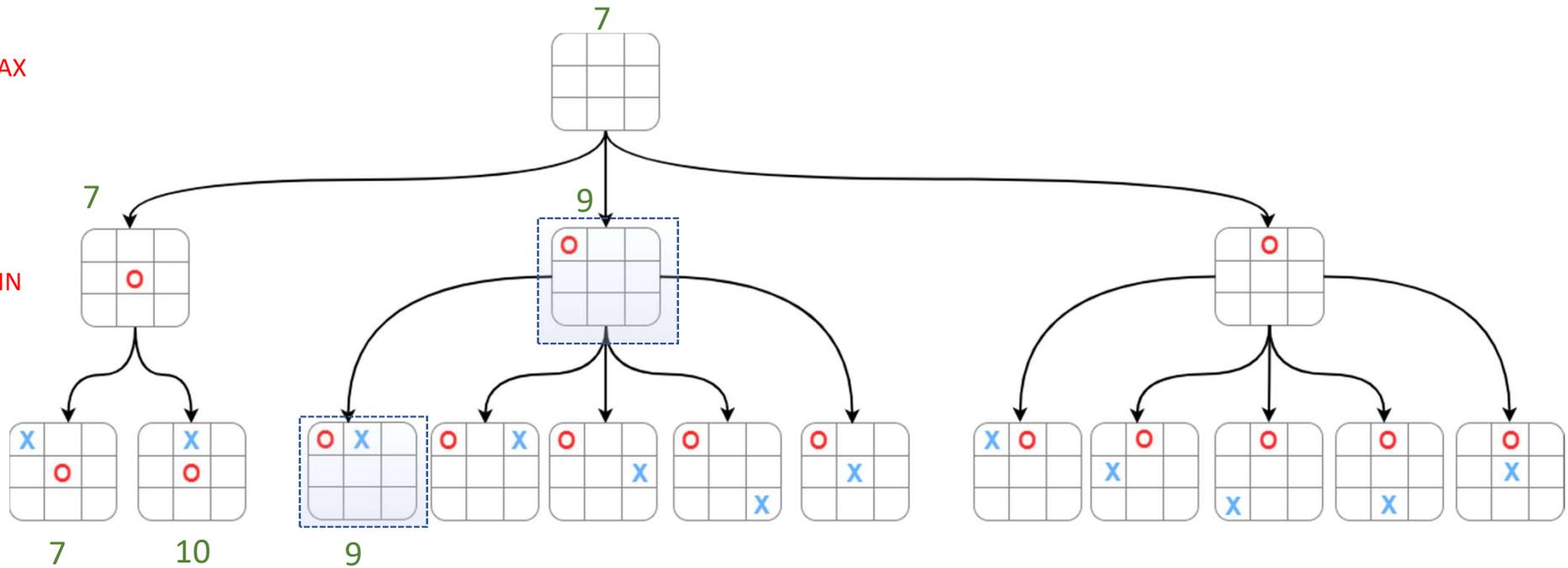


La fonction d'évaluation donne un score de 9.

Joueur 1 : MAX



Joueur 2 : MIN

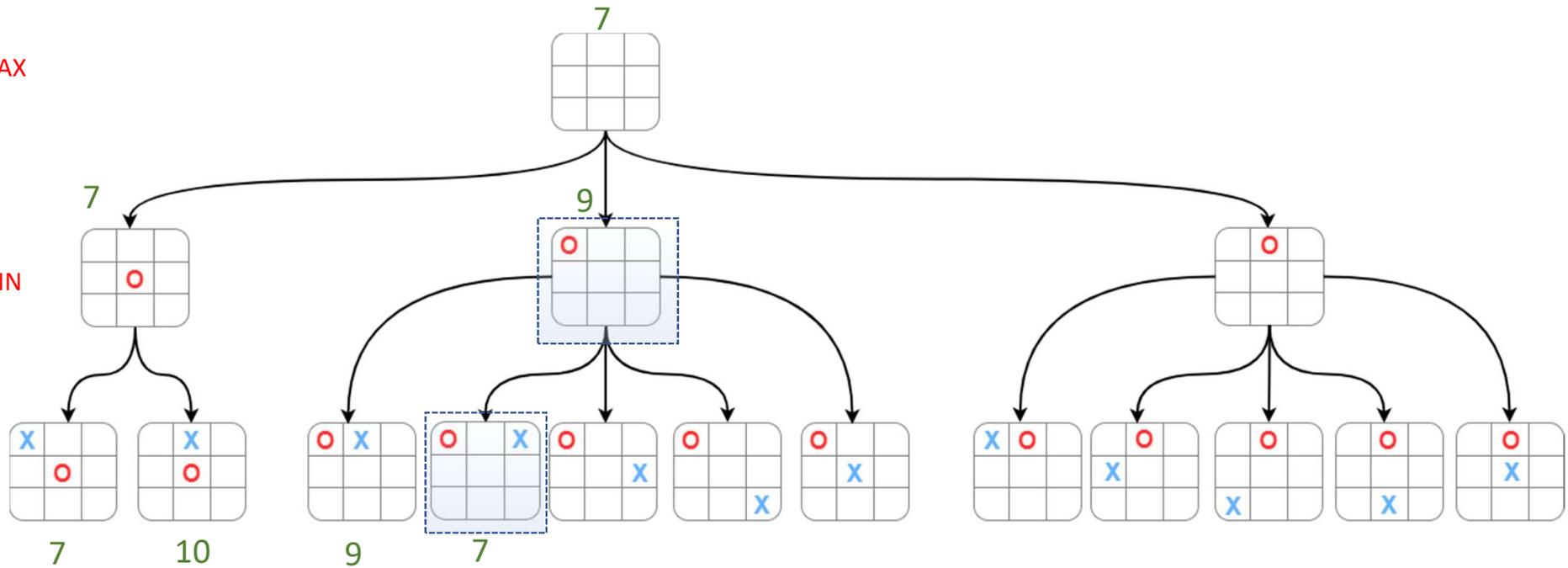


Le meilleur score dans cette position pour le joueur 2 est pour le moment de 9.

Joueur 1 : MAX



Joueur 2 : MIN

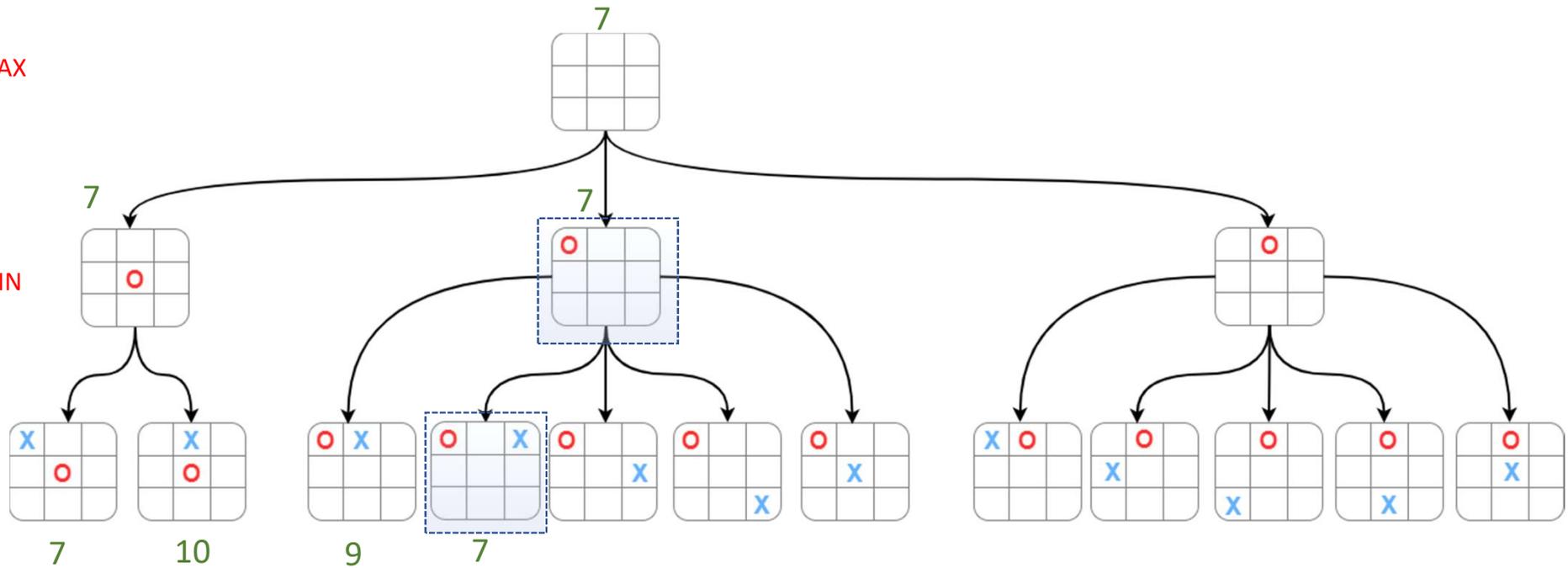


L'IA joue ensuite le second coup possible du joueur 2 et évalue la position. Son score est de 7.

Joueur 1 : **MAX**



Joueur 2 : **MIN**

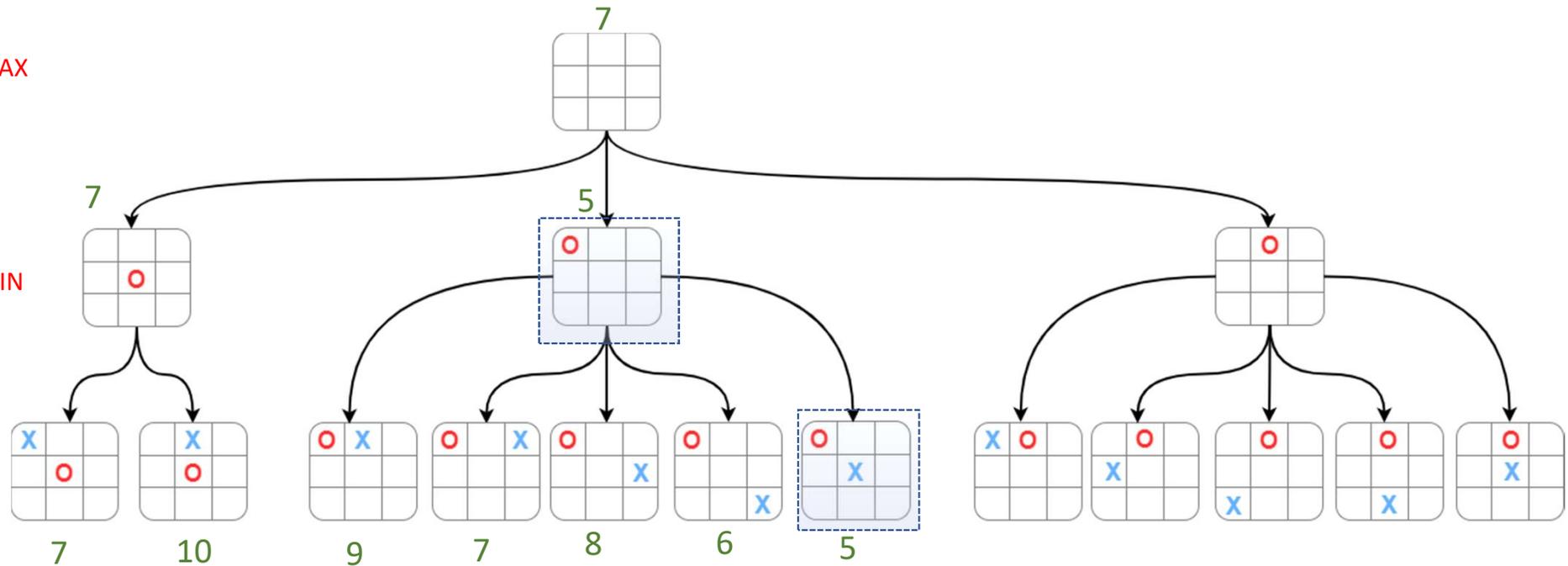


Comme 7 est inférieur à 9, le meilleur score du joueur 2 passe de 9 à 7.

Joueur 1 : MAX



Joueur 2 : MIN

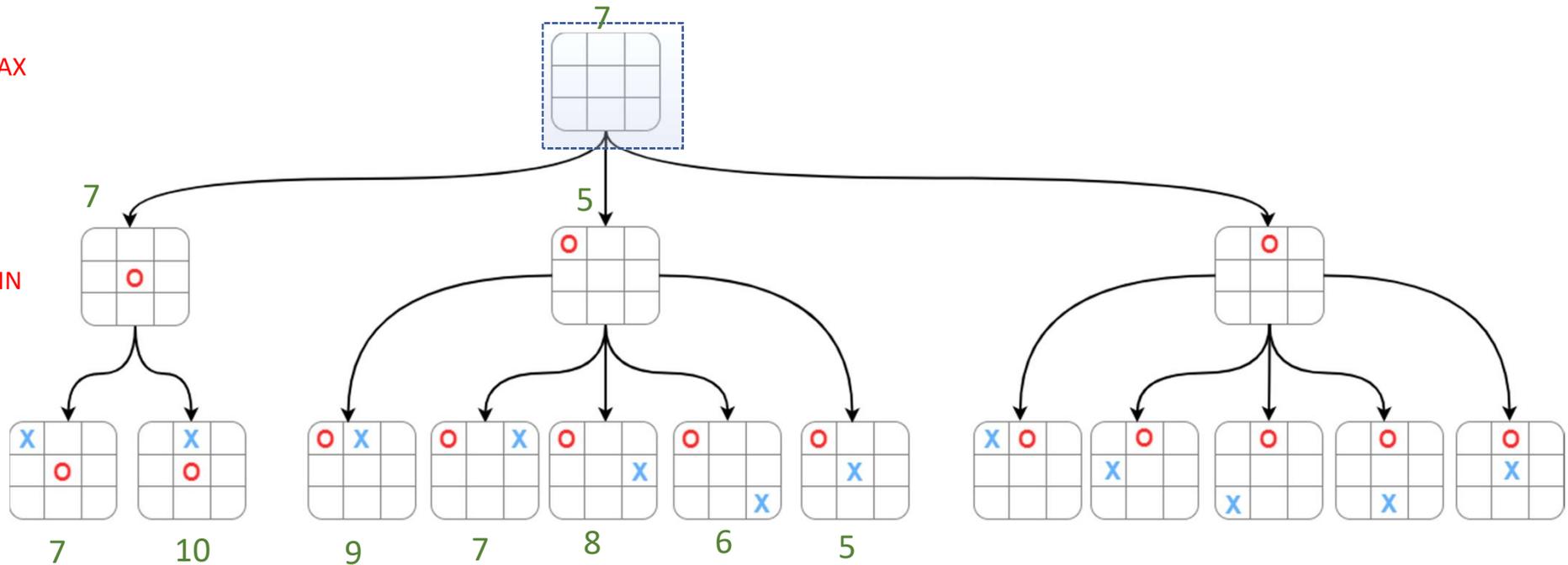


L'IA joue tous les coups possibles du joueur 2 et évalue chaque position obtenue. Elle détermine le score du joueur 2 en minimisant les valeurs obtenues par la fonction d'évaluation. Dans cette position le meilleur score possible pour le joueur 2 est donc de 5.

Joueur 1 : MAX



Joueur 2 : MIN

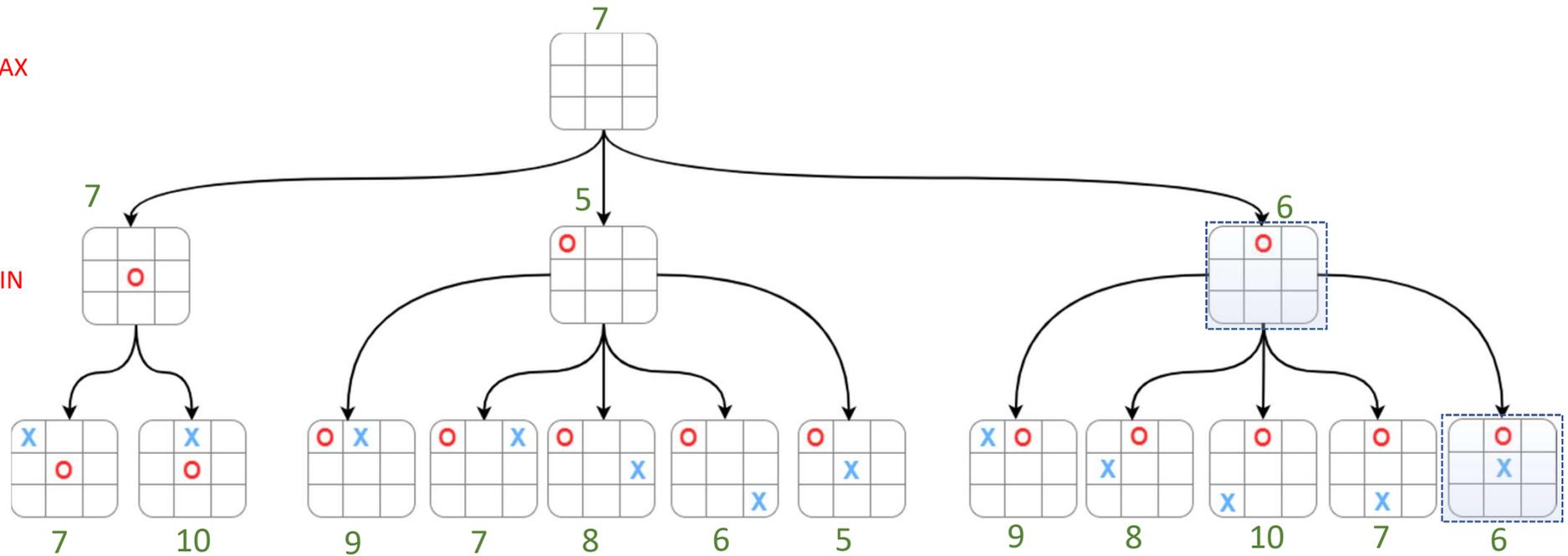


Le joueur 1 compare son score actuel de 7 avec le score de 5. Son score ne change pas puisqu'il maximise ce score.

Joueur 1 : MAX



Joueur 2 : MIN



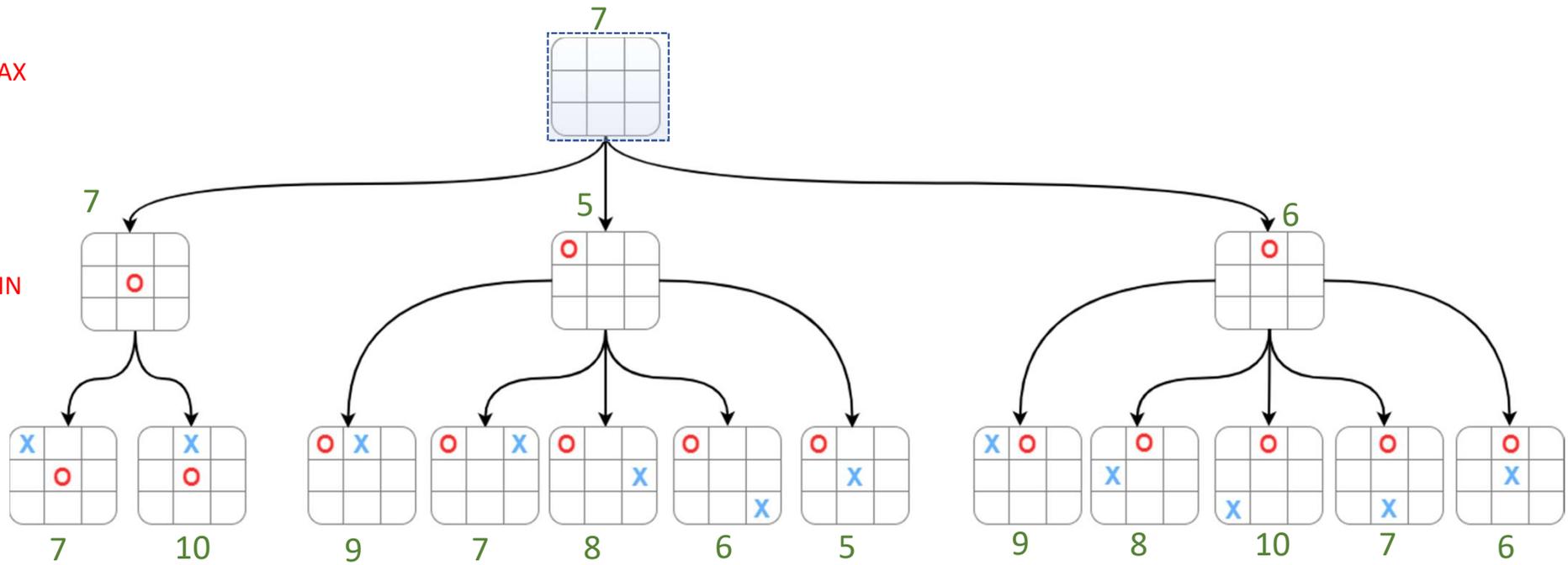
Evaluation

L'IA procède de la même manière pour le 3^{ème} coup possible du joueur 1.

Joueur 1 : **MAX**



Joueur 2 : **MIN**



Evaluation

Le joueur 1 maximise le score. Son score est donc de 7.

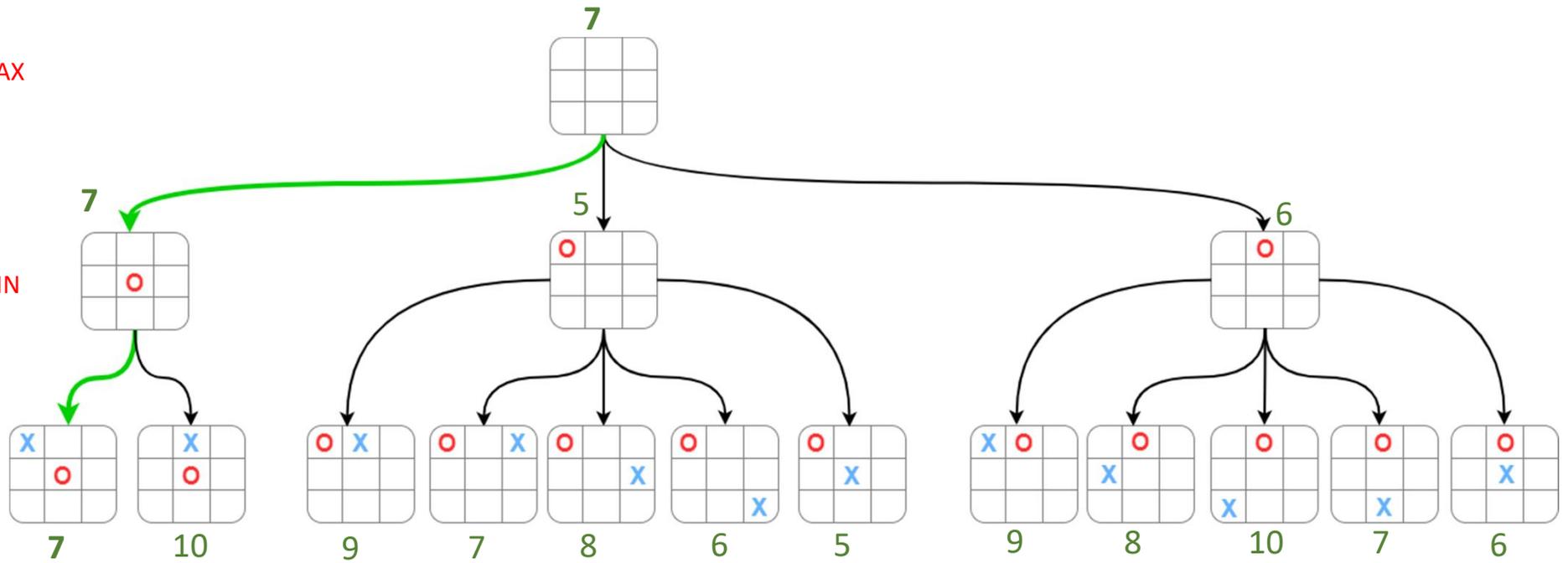
Joueur 1 : **MAX**



Joueur 2 : **MIN**



Evaluation



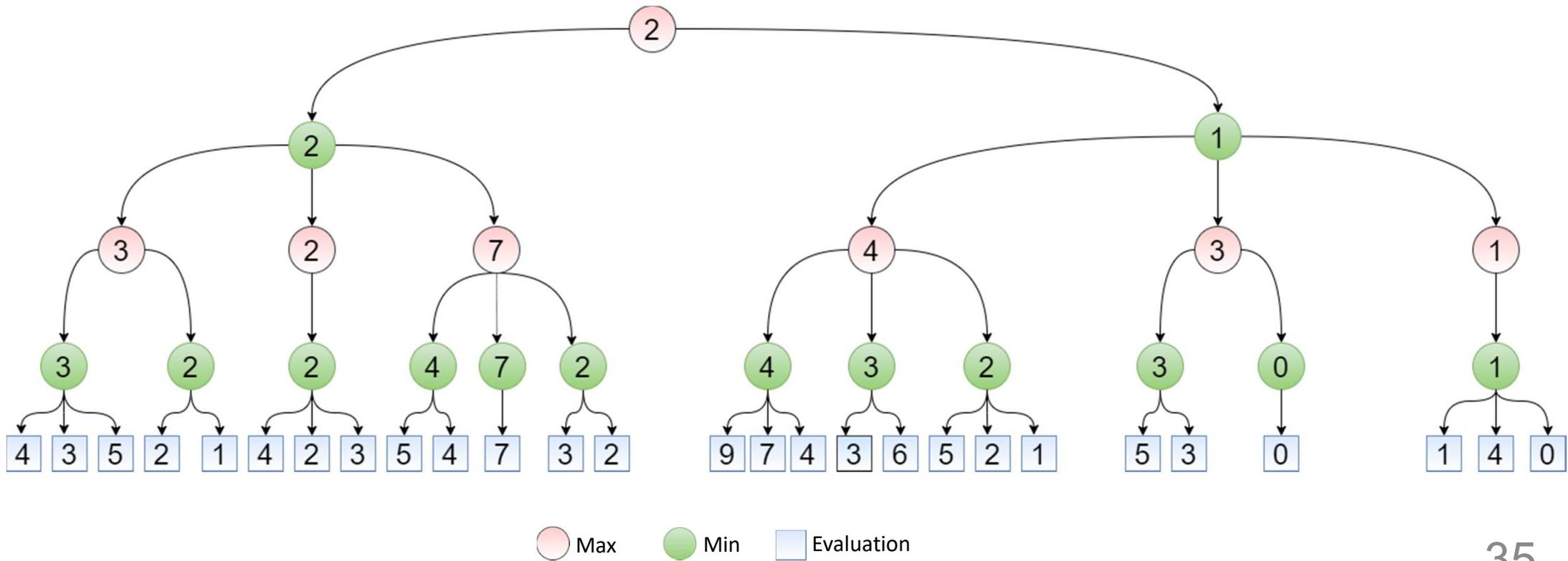
Le joueur 1 jouera donc le premier coup qu'il a analysé.

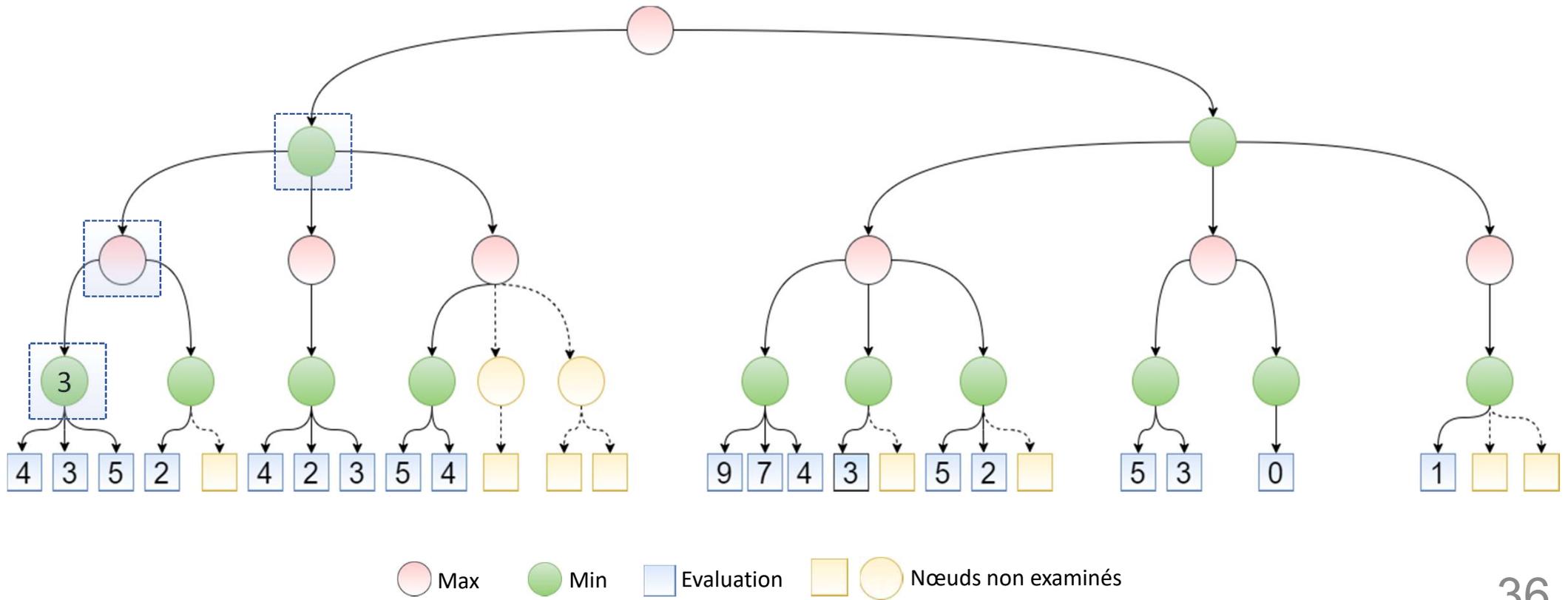


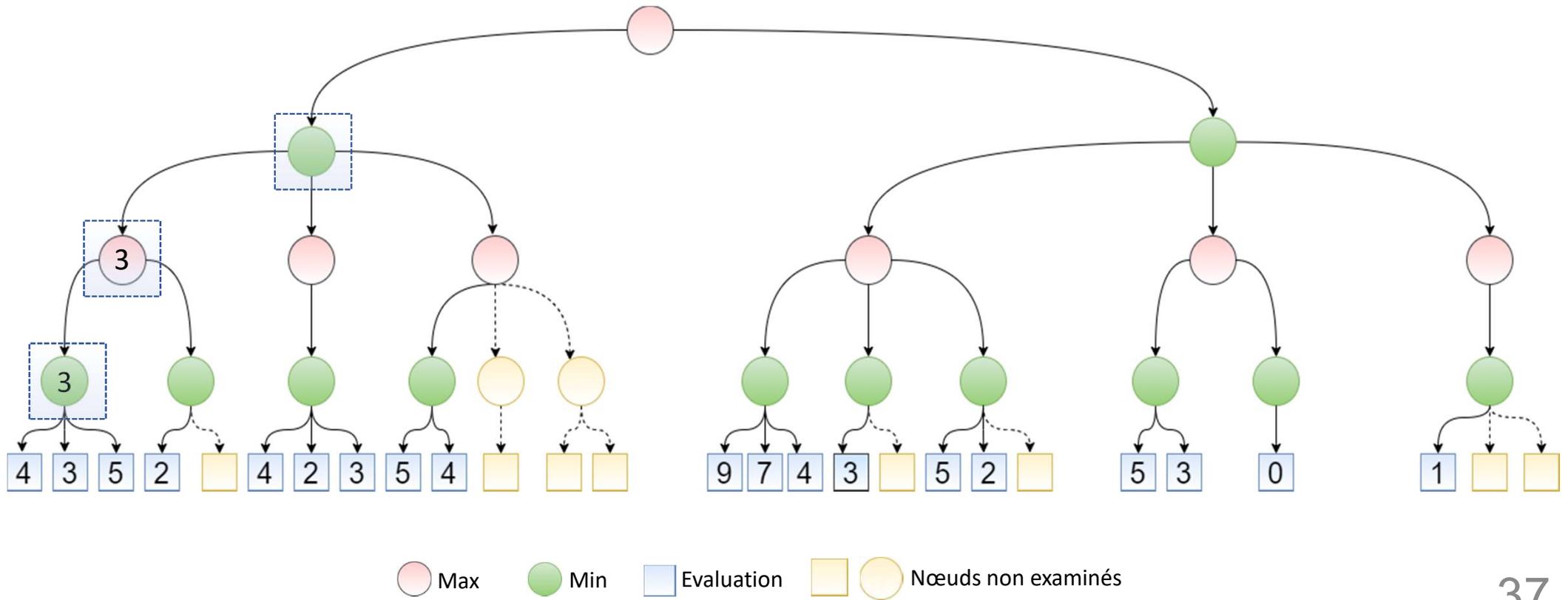
IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Alpha Beta

Amélioration
du Min-Max

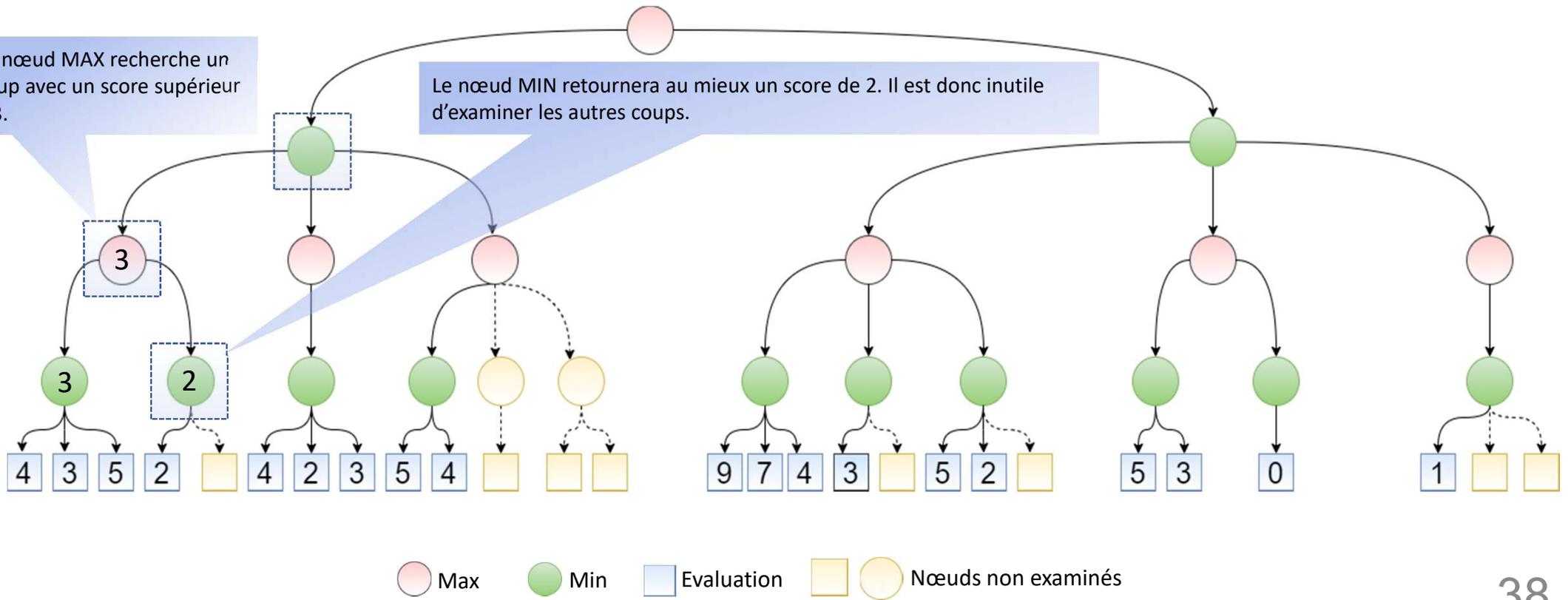


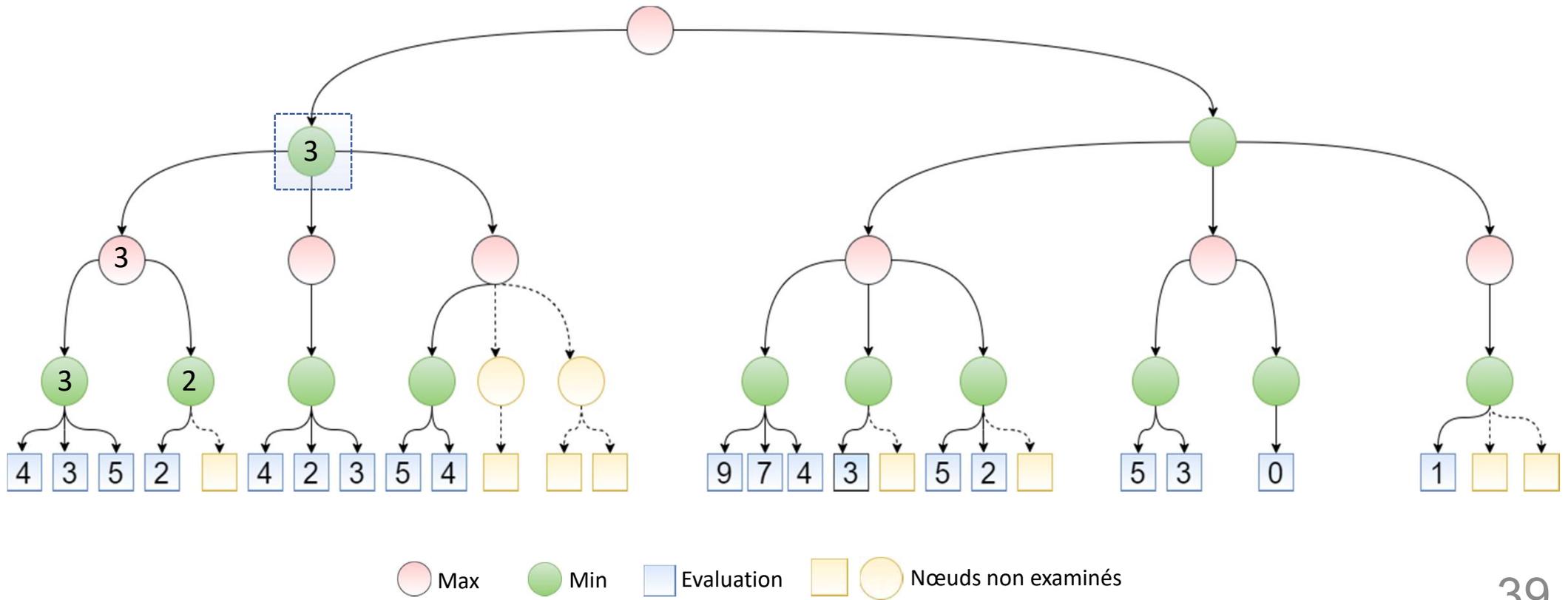


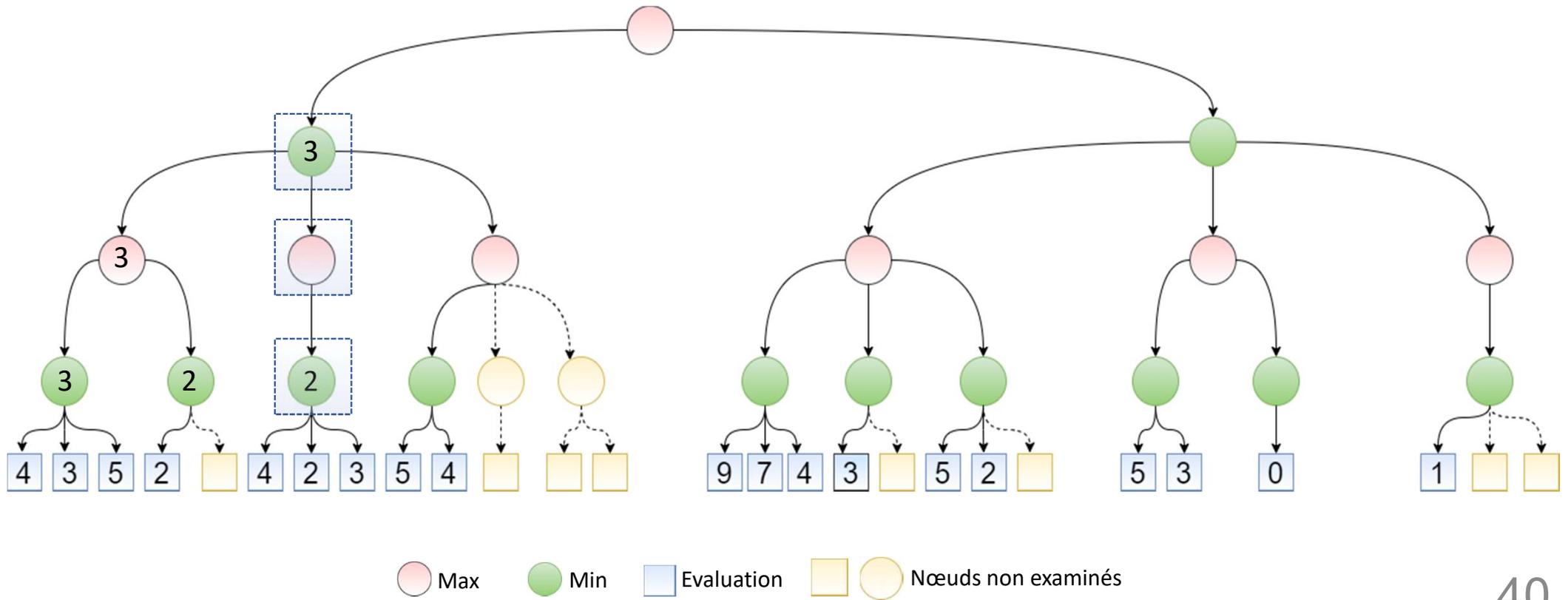


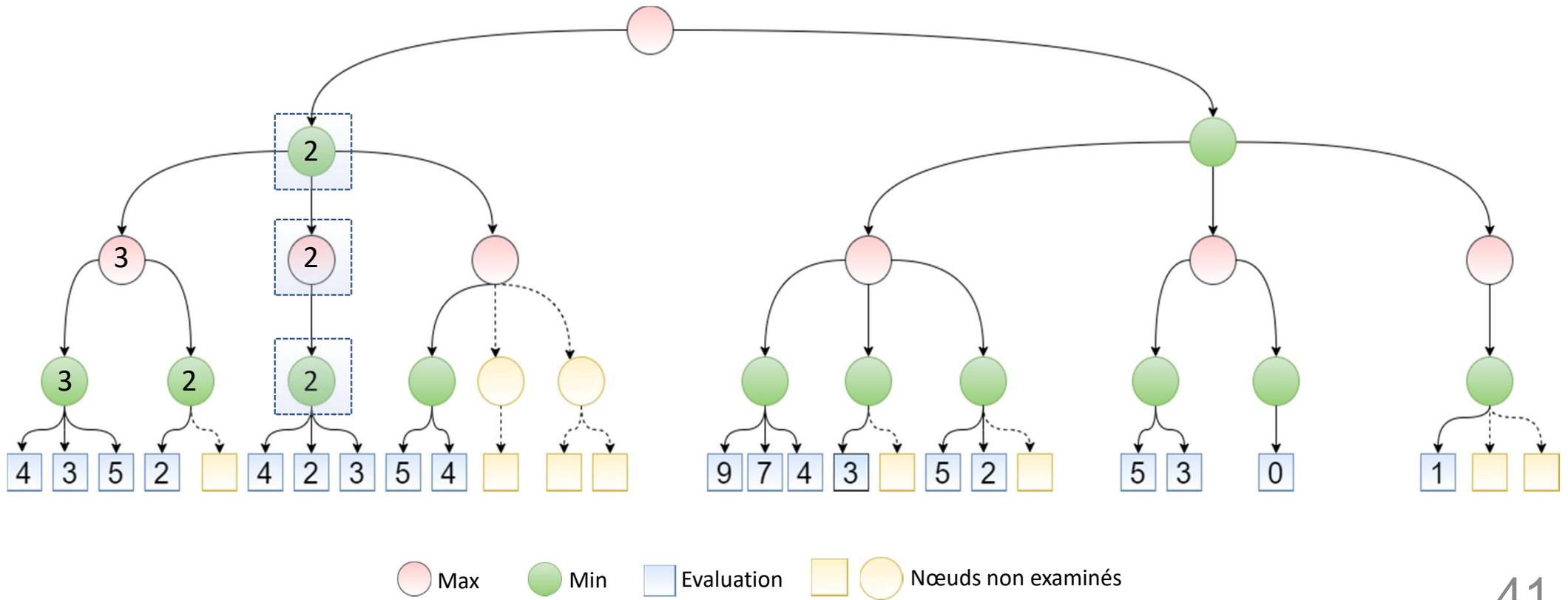
Le nœud MAX recherche un coup avec un score supérieur à 3.

Le nœud MIN retournera au mieux un score de 2. Il est donc inutile d'examiner les autres coups.



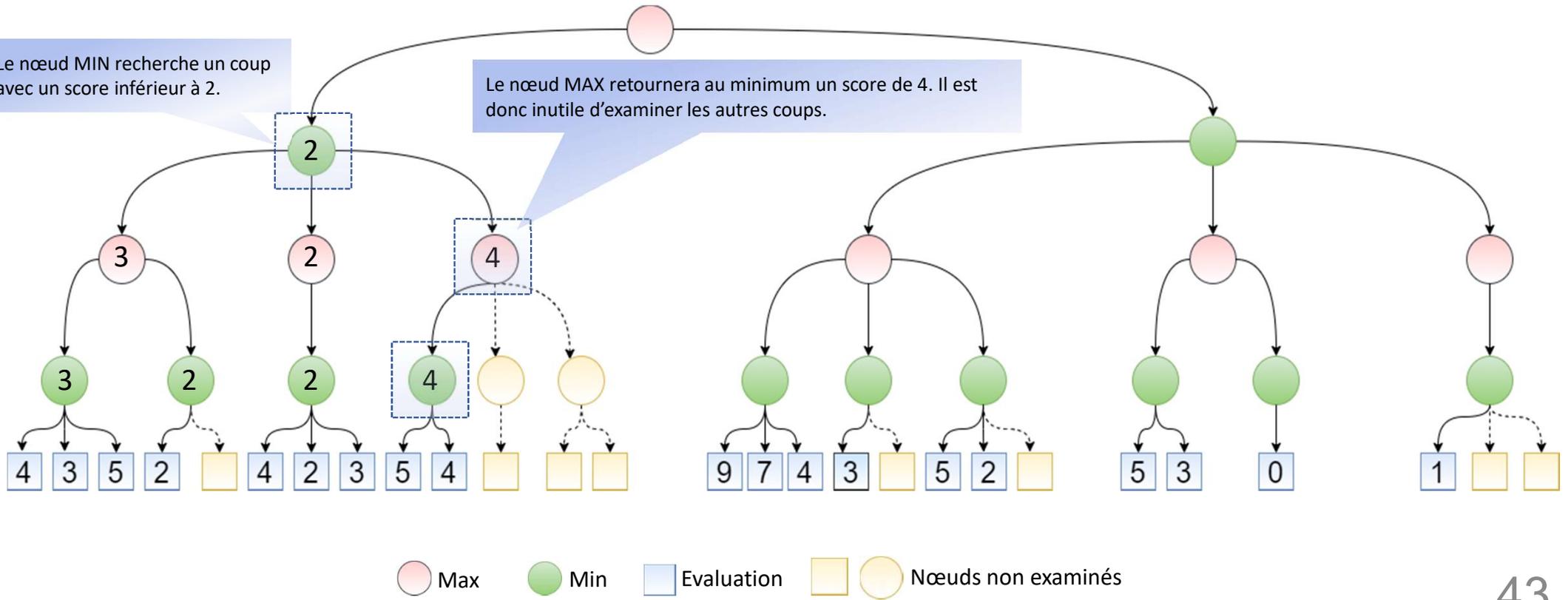


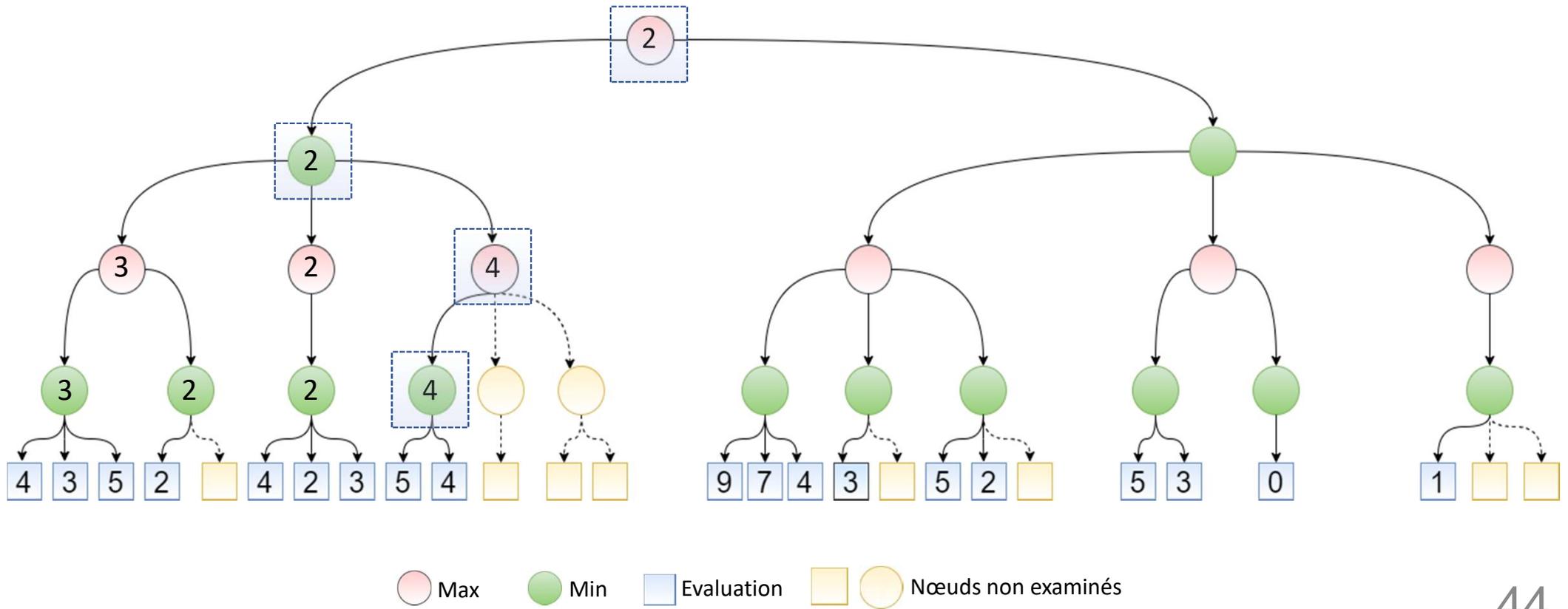


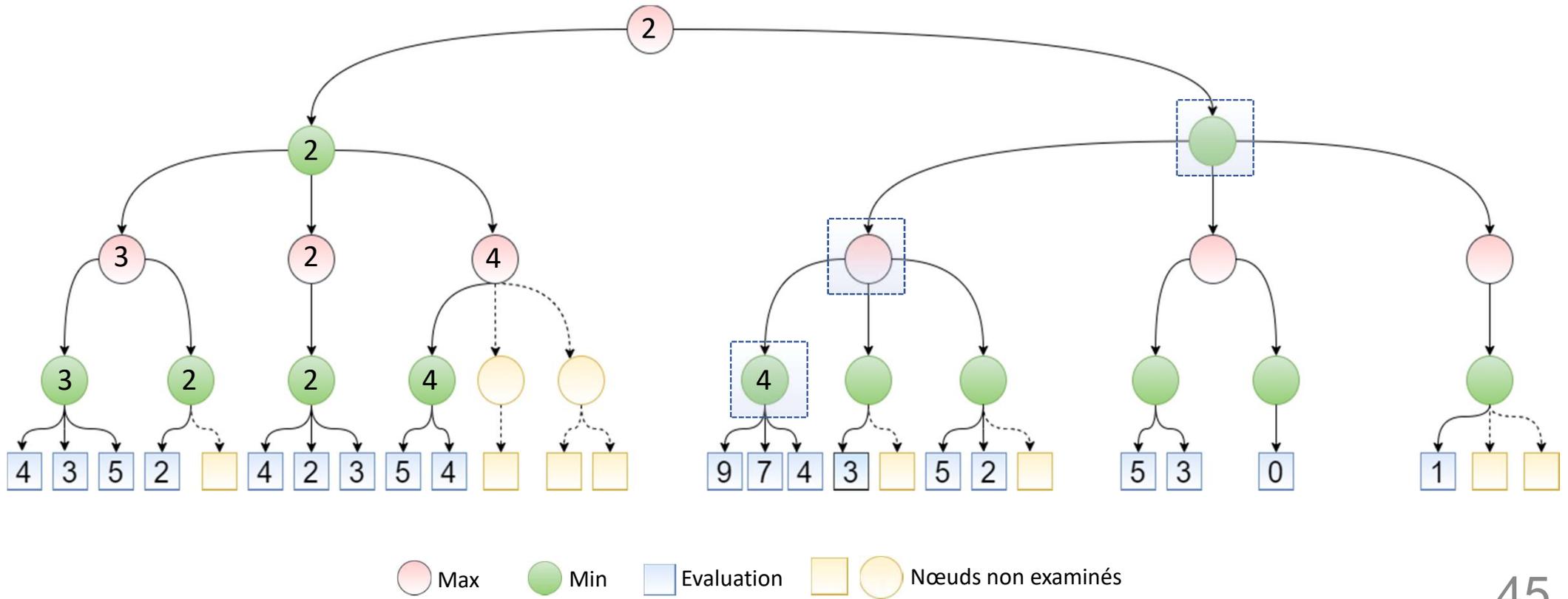


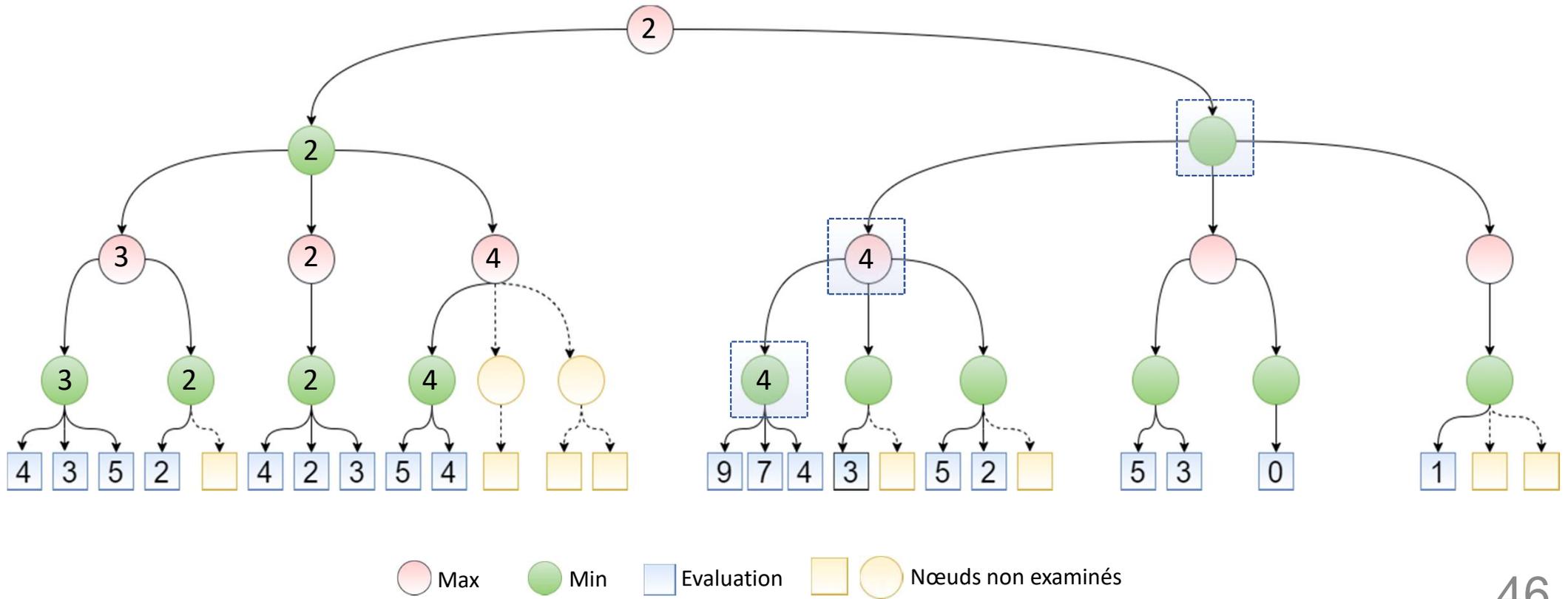
Le nœud MIN recherche un coup avec un score inférieur à 2.

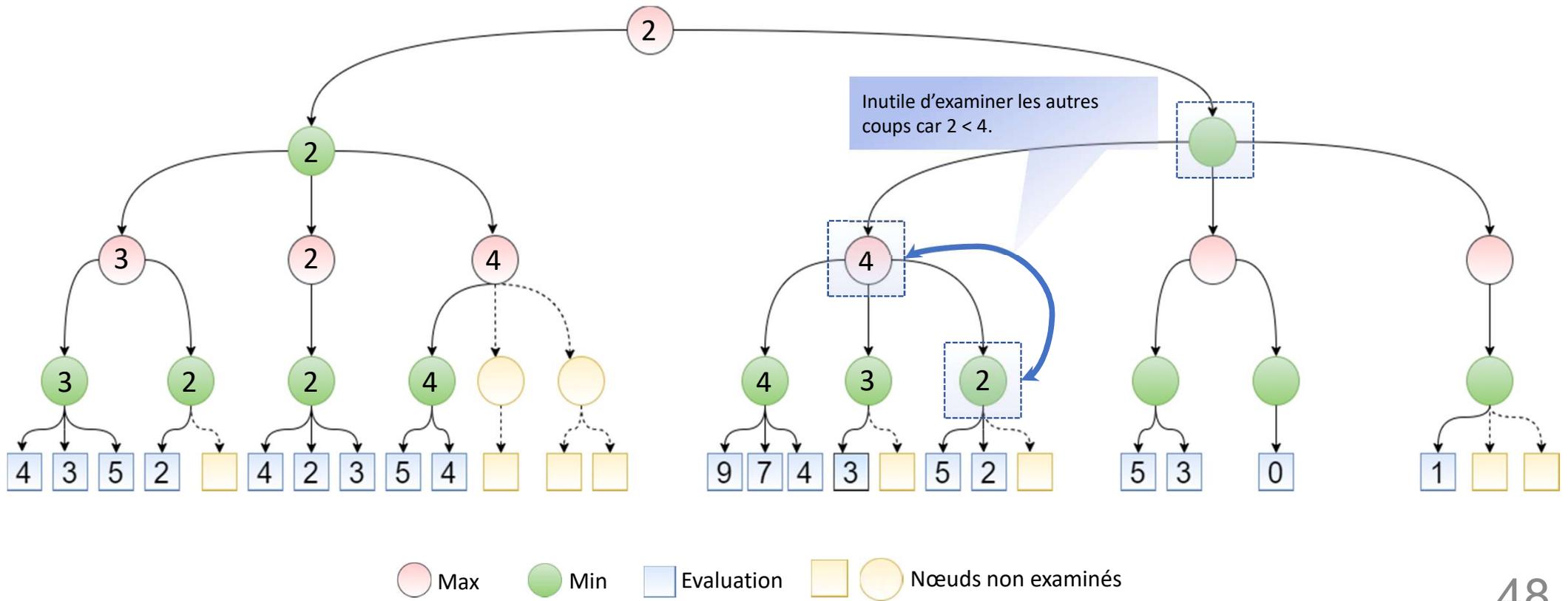
Le nœud MAX retournera au minimum un score de 4. Il est donc inutile d'examiner les autres coups.

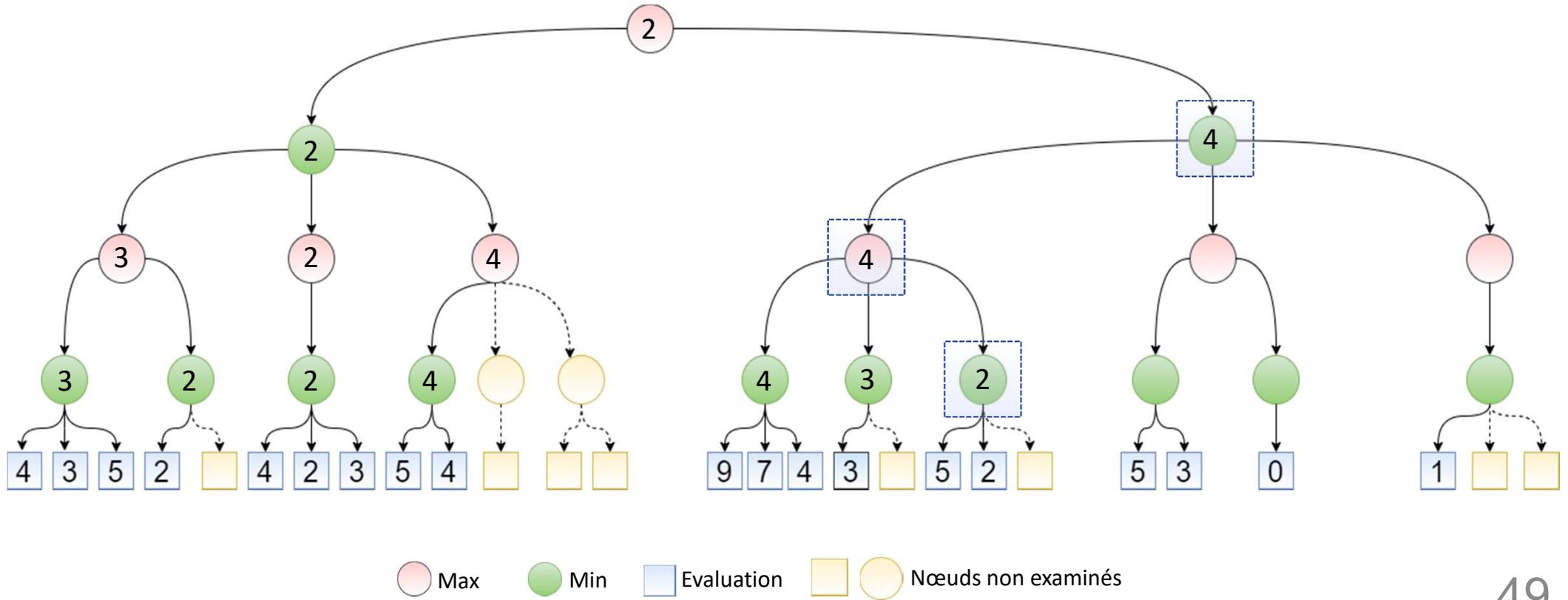


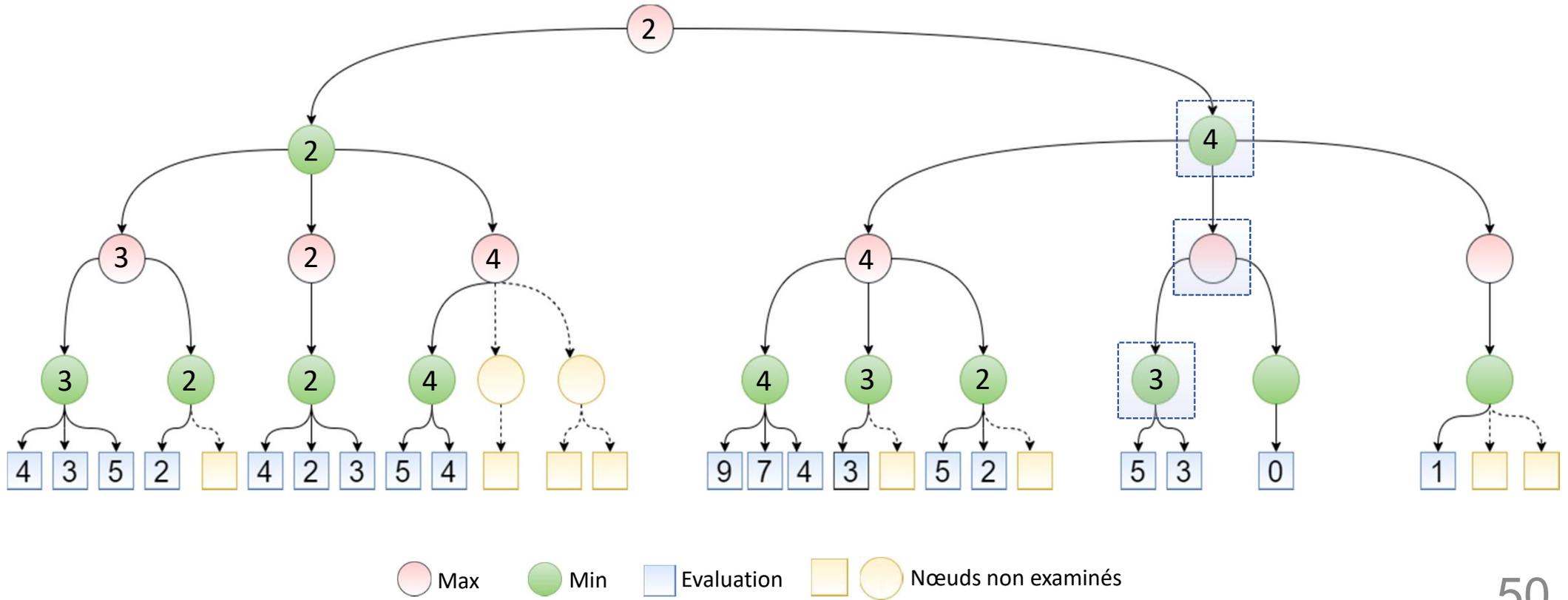


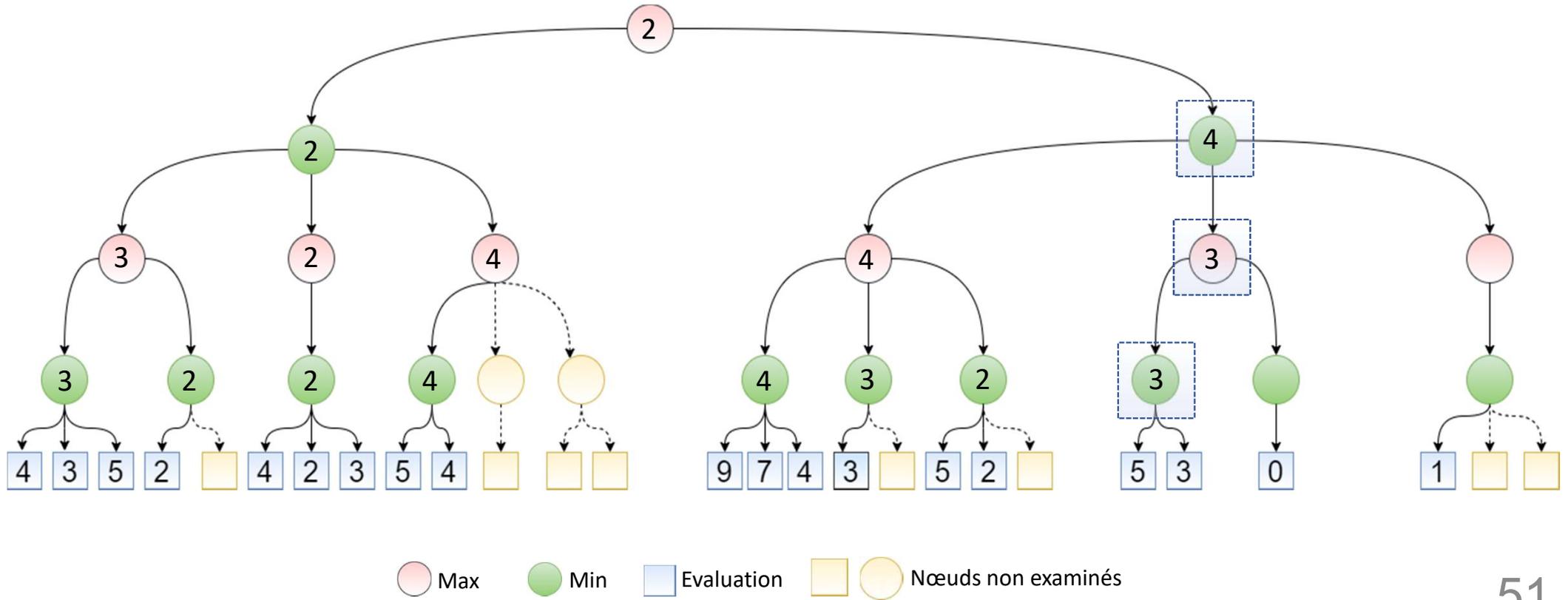


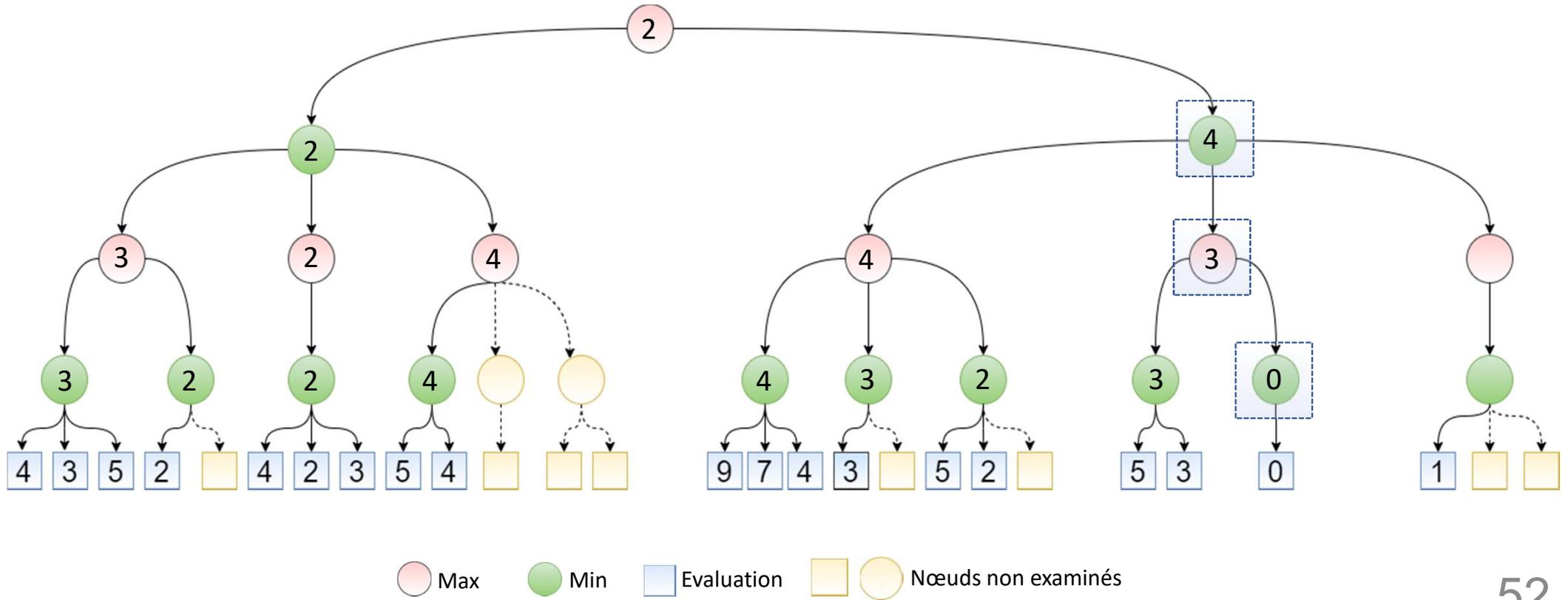


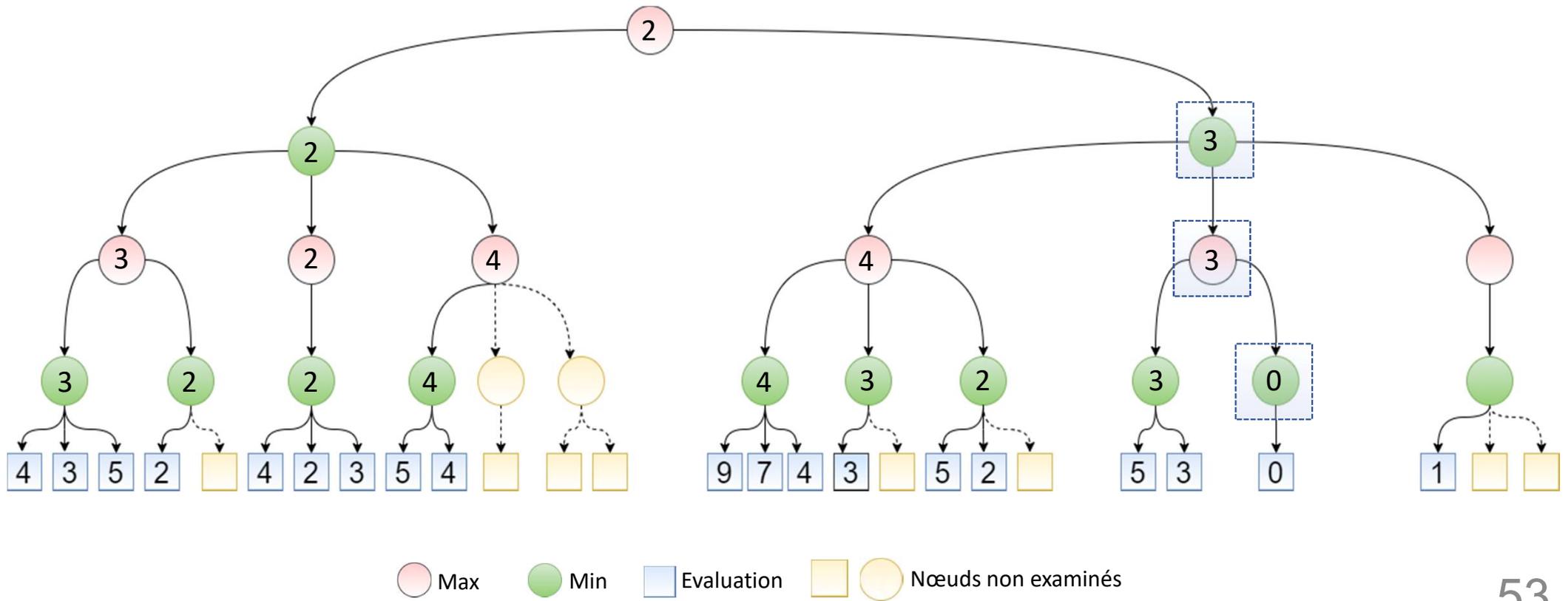


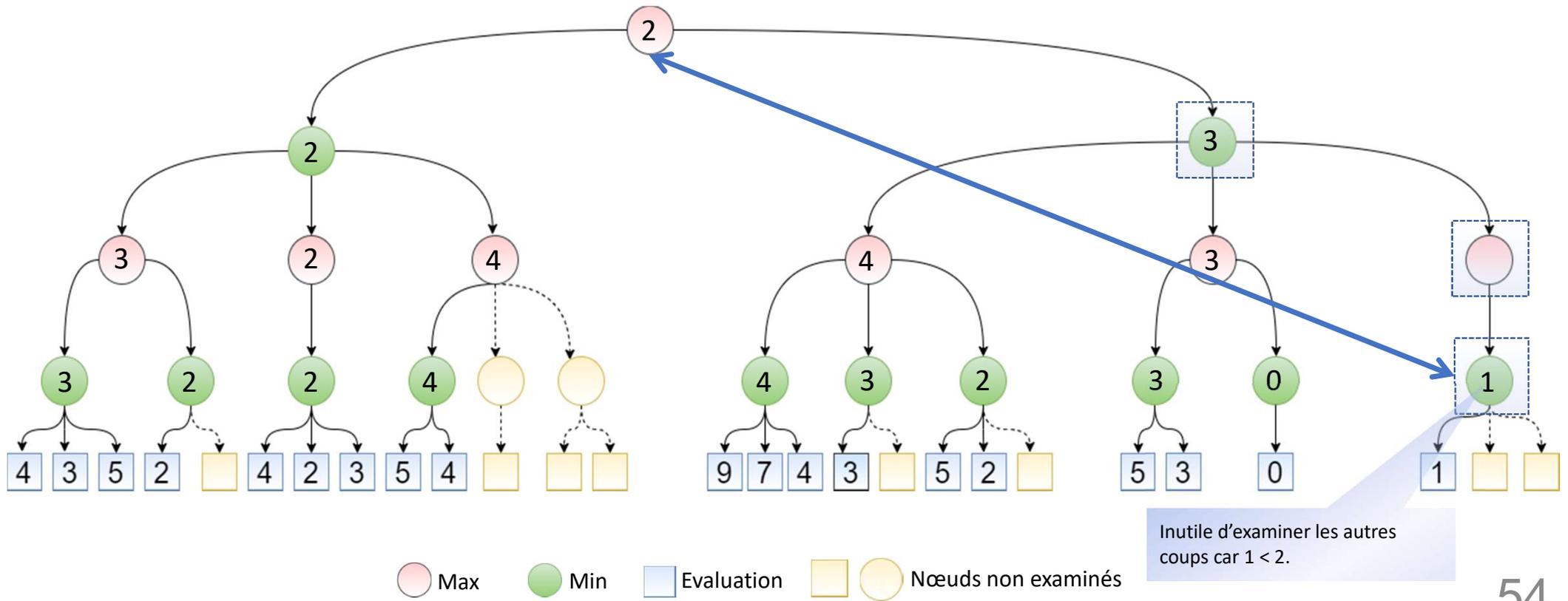


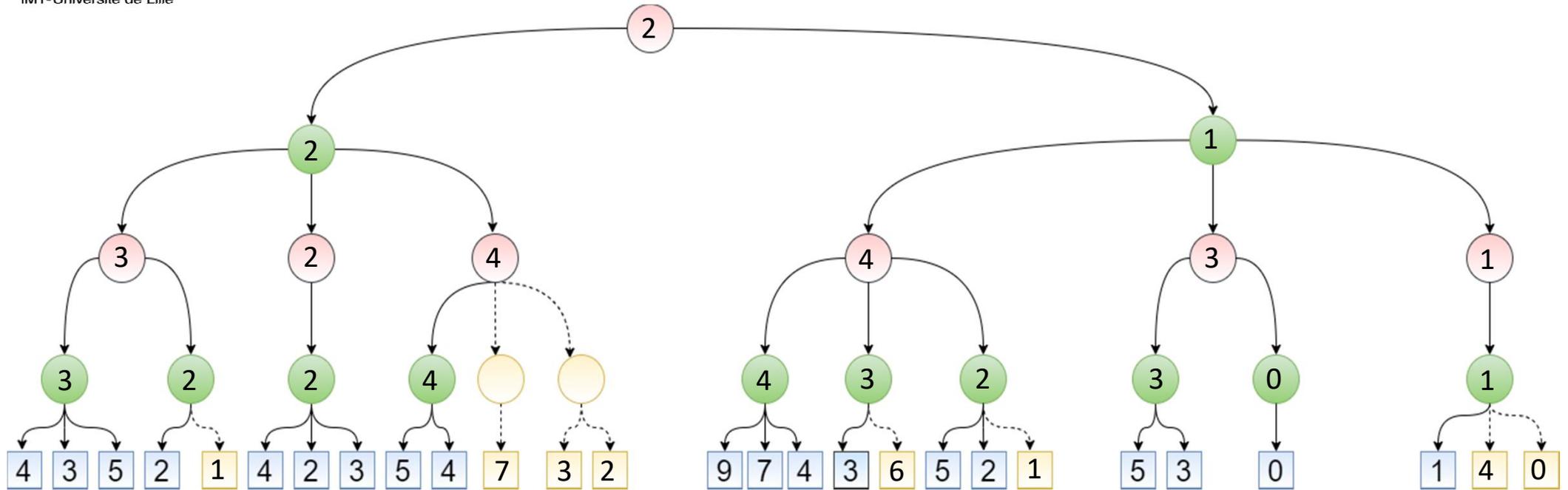






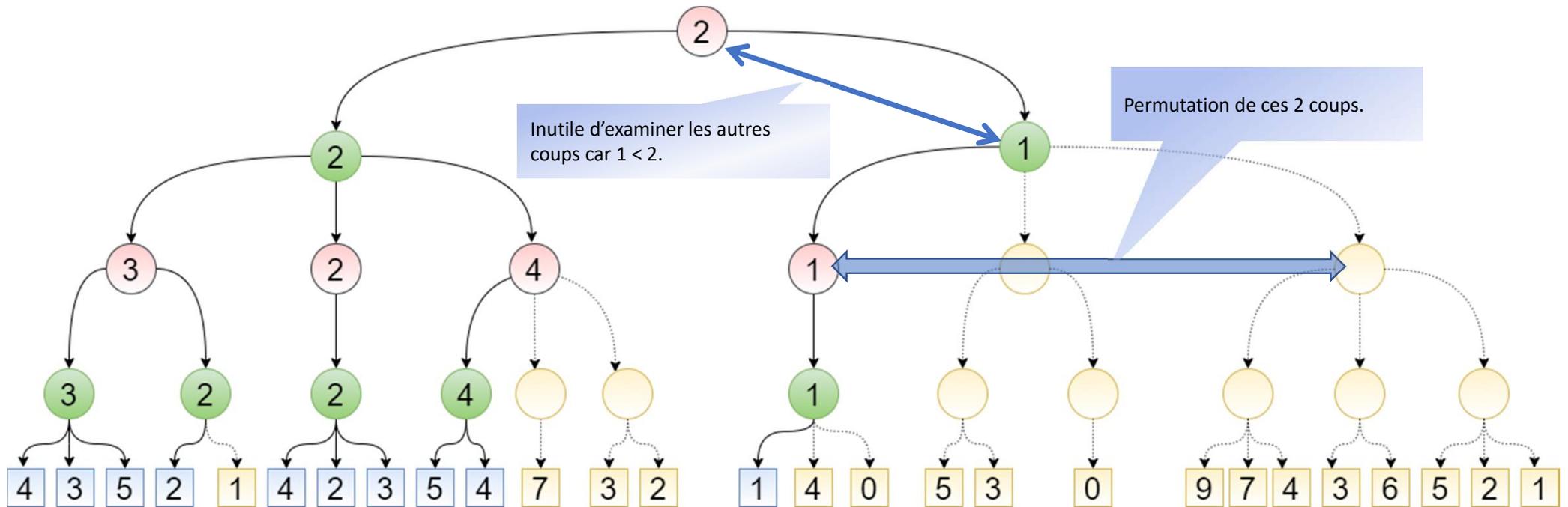






L'algorithme Alpha-Beta a permis d'éviter l'analyse de 8 feuilles sur les 27.

Peut-on mieux faire ?



Mise en place d'une heuristique de tri des coups, afin de jouer dans la mesure du possible les meilleurs coups en premiers. Ici l'algorithme analyse 10 feuilles sur les 27.



IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

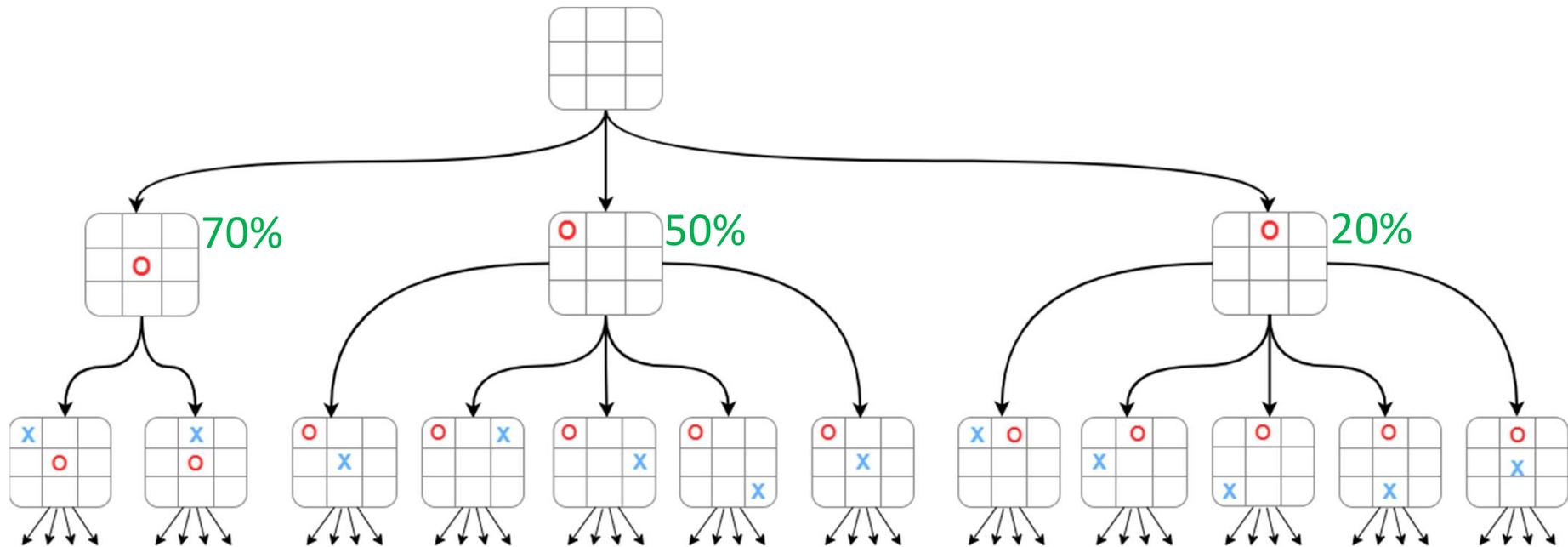
Monte Carlo Tree Search

MCTS

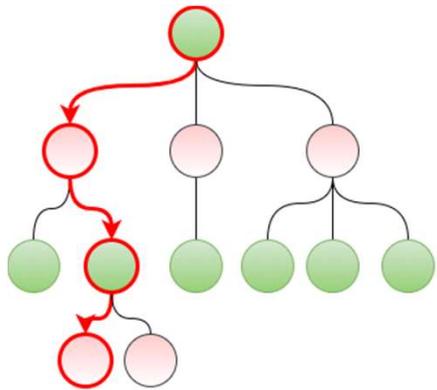
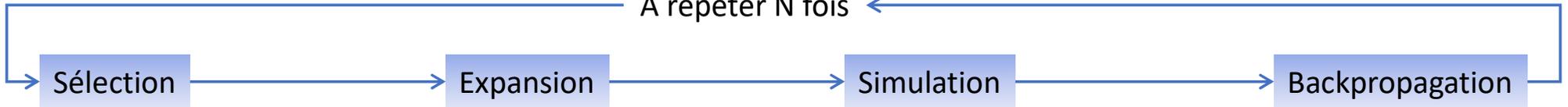
L'algorithme Alpha-Beta fonctionne très bien sur un jeu comme le jeu d'échecs où l'arbre de jeu reste "raisonnable".

En revanche, il donne des résultats très médiocres sur un jeu comme le Go.

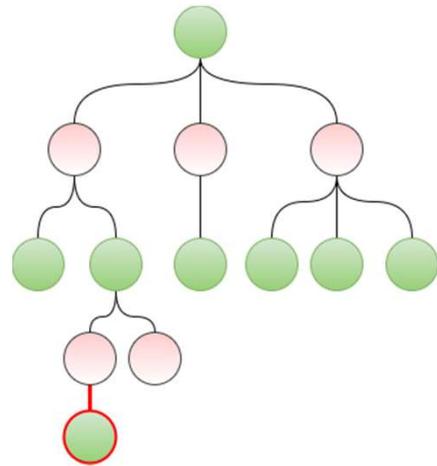
L'idée de cet algorithme est de simuler énormément de parties au hasard et de jouer le coup ayant le taux de gain le plus élevé.



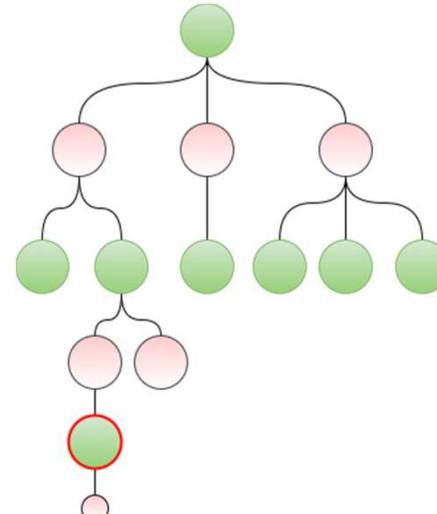
A répéter N fois ←



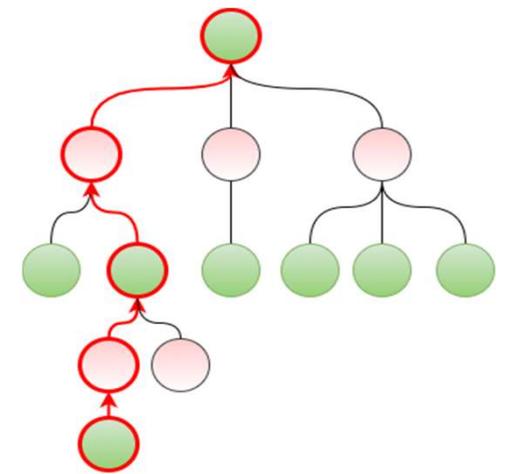
On parcourt l'arbre en sélectionnant à chaque niveau un nœud jusqu'à atteindre une feuille de l'arbre.



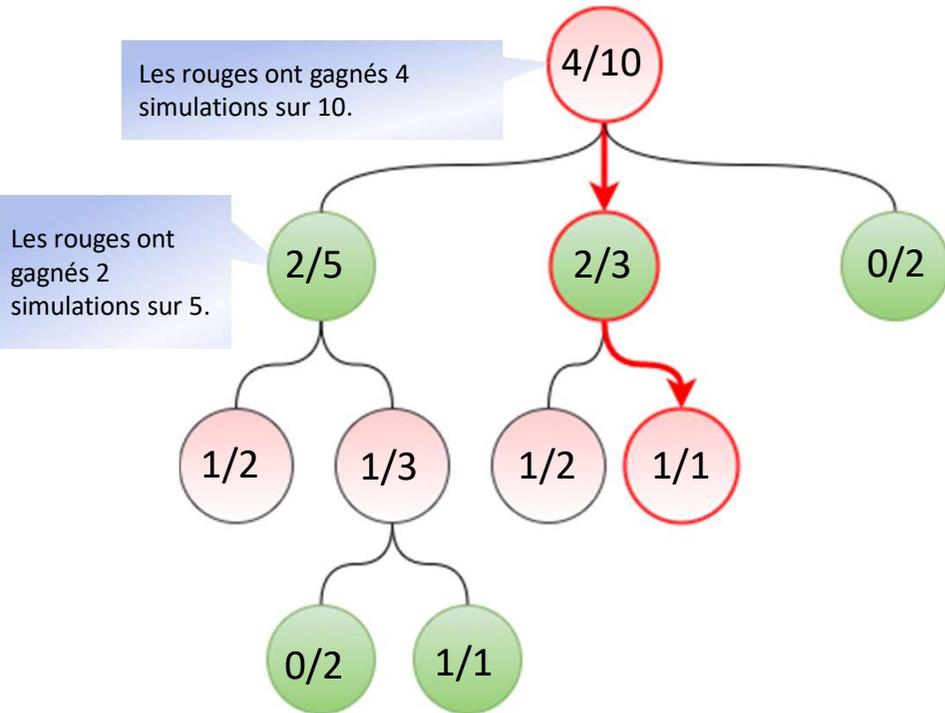
A partir du nœud sélectionné précédemment on ajoute un nouveau nœud.



On effectue une simulation d'une partie à partir de cette position.



On met à jour le score de la branche à partir du résultat de la simulation.



On sélectionne le nœud fils qui maximise la valeur suivante :

$$\underbrace{\frac{V_i}{S_i}}_{\text{Exploitation}} + C \underbrace{\sqrt{\frac{\ln S_p}{S_i}}}_{\text{Exploration}}$$

V_i : Nombre de simulations ayant abouti à une victoire

S_i : Nombre de simulations effectuées

S_p : Nombre de simulations effectuées au niveau du nœud père.

C : Une constante qui permet de faire varier le niveau d'exploration.

On sélectionne le nœud fils qui maximise la valeur suivante :

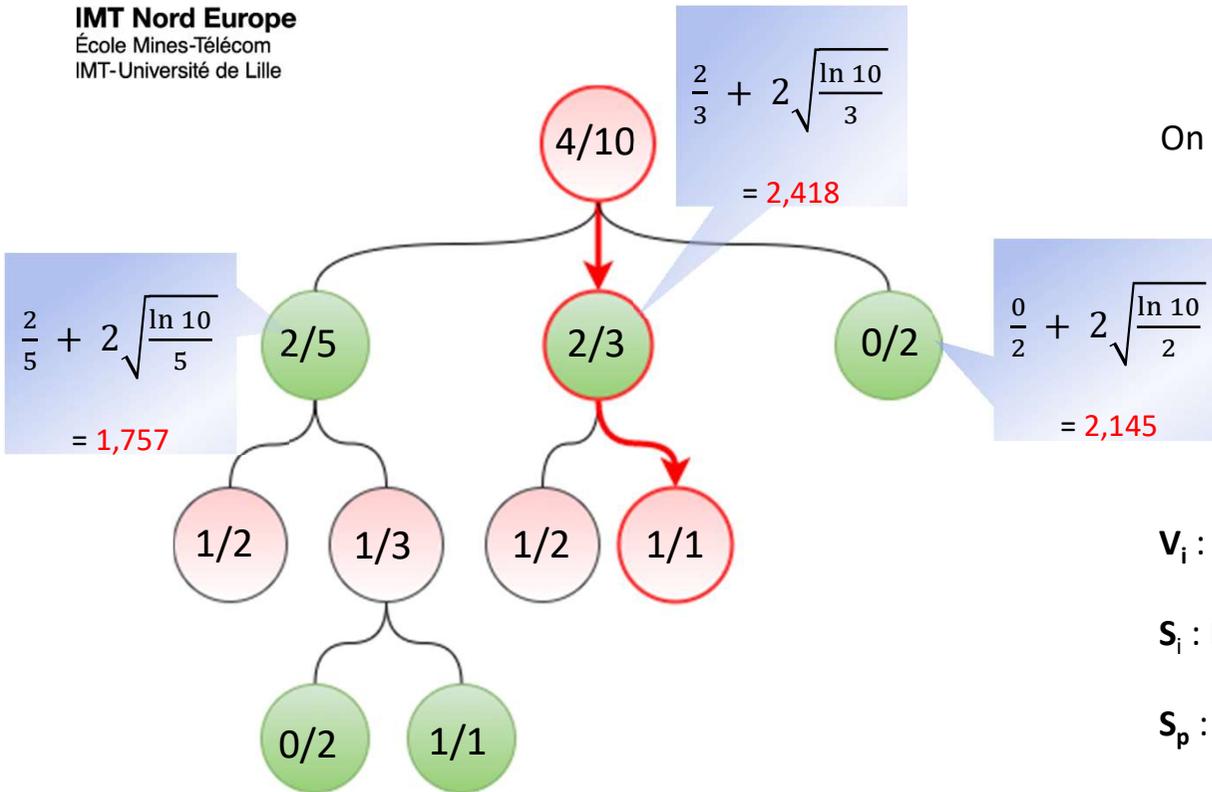
$$\underbrace{\frac{V_i}{S_i}}_{\text{Exploitation}} + C \underbrace{\sqrt{\frac{\ln S_p}{S_i}}}_{\text{Exploration}}$$

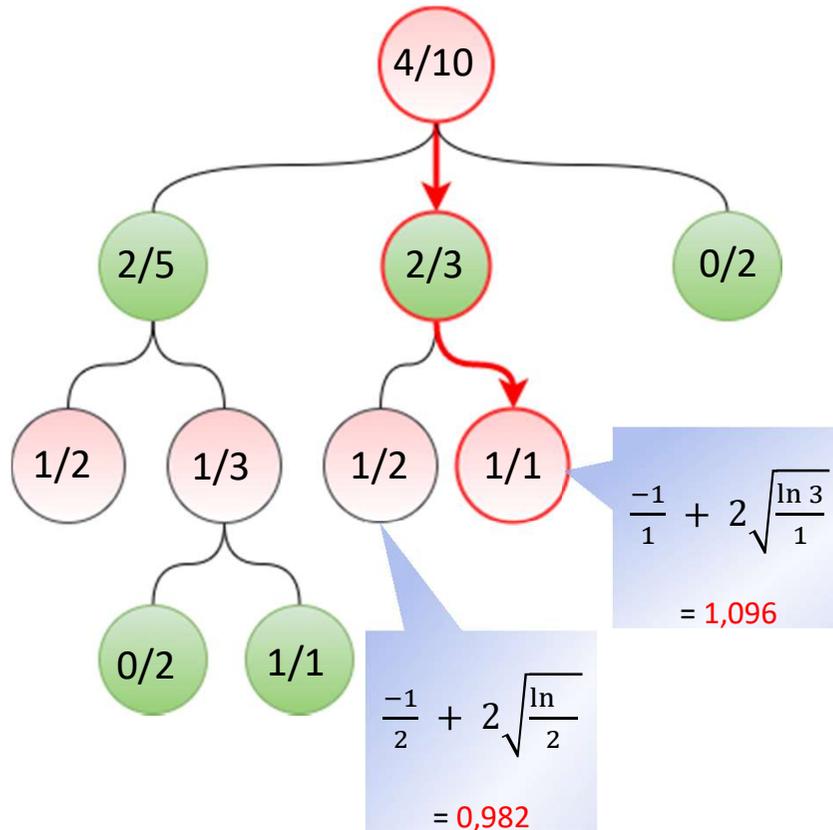
V_i : Nombre de simulations ayant abouti à une victoire

S_i : Nombre de simulations effectuées

S_p : Nombre de simulations effectuées au niveau du nœud père.

C : Une constante qui permet de faire varier le niveau d'exploration.





On sélectionne le nœud fils qui maximise la valeur suivante :

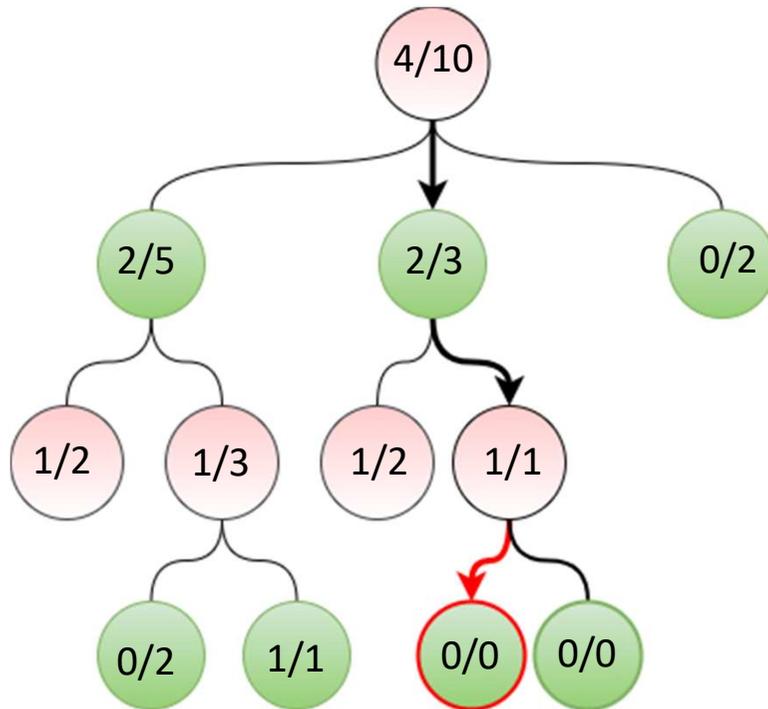
$$\underbrace{\frac{V_i}{S_i}}_{\text{Exploitation}} + C \underbrace{\sqrt{\frac{\ln S_p}{S_i}}}_{\text{Exploration}}$$

V_i : Nombre de simulations ayant abouti à une victoire

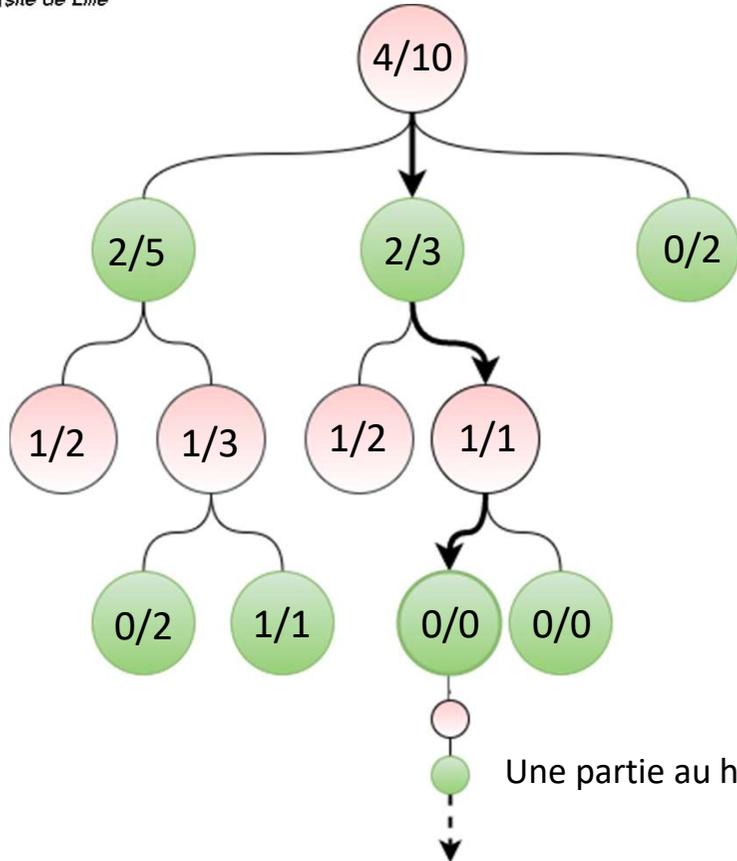
S_i : Nombre de simulations effectuées

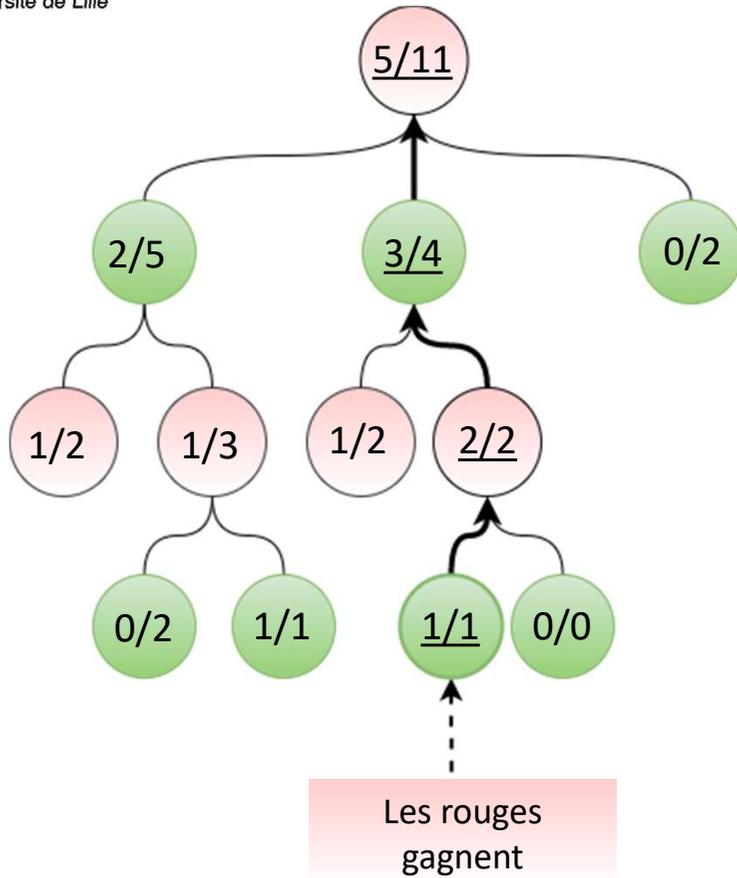
S_p : Nombre de simulations effectuées au niveau du nœud père.

C : Une constante qui permet de faire varier le niveau d'exploration.

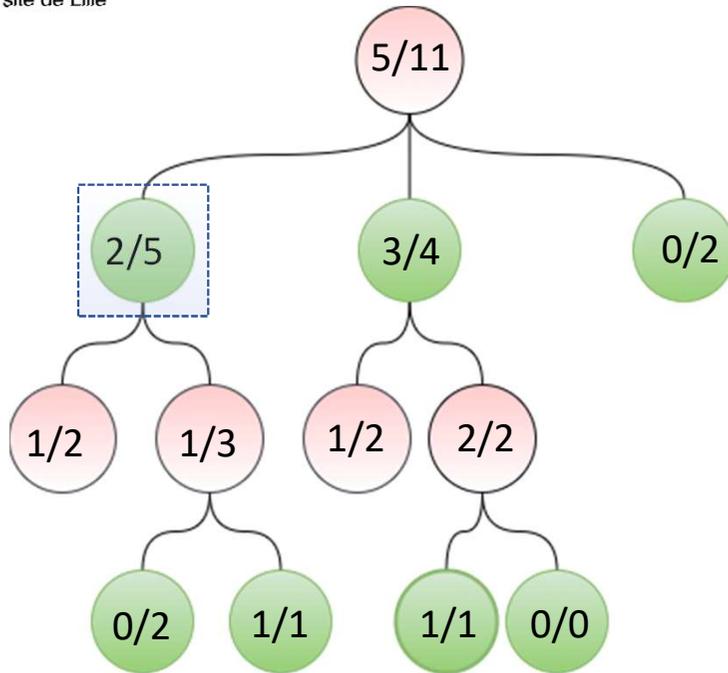


1. Ajouter dans l'arbre les nœuds fils du nœud sélectionné pendant la phase de sélection.
2. Initialiser le nombre de victoires et le nombre de simulations à 0 à chacun de ces nœuds.
3. Choisir au hasard un des nœuds fils créés.





A la fin de la simulation, le compteur de simulation et le score des différents nœuds parcourus dans l'arbre sont mis à jour.



Plusieurs possibilités pour déterminer le coup à jouer.

Jouer le coup ayant :

- Le meilleur ratio $\frac{V}{S}$
- Le plus de grand nombre de simulations



IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

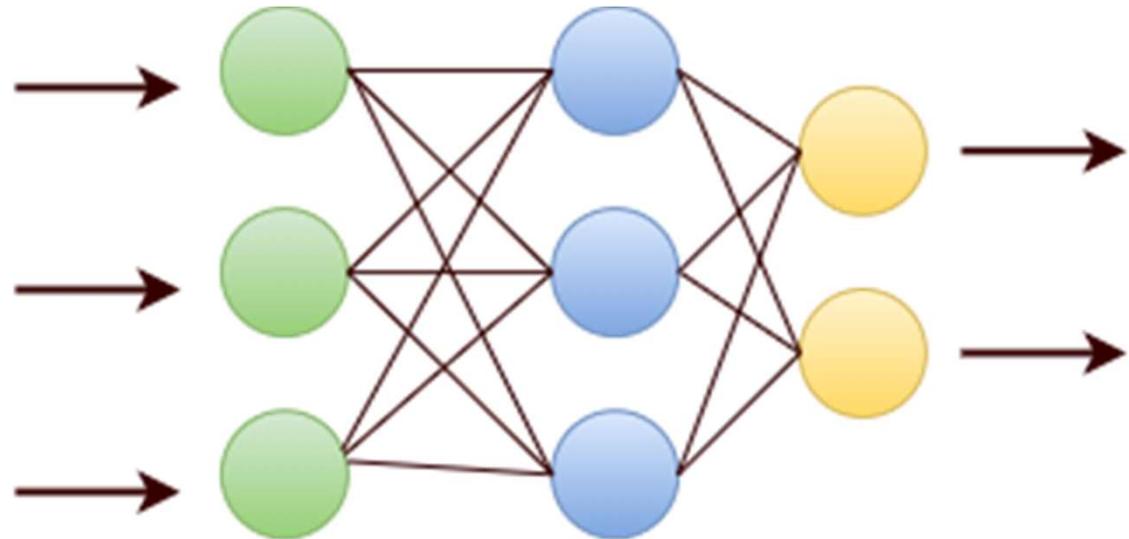
Alpha Zéro

#IMTomorrow

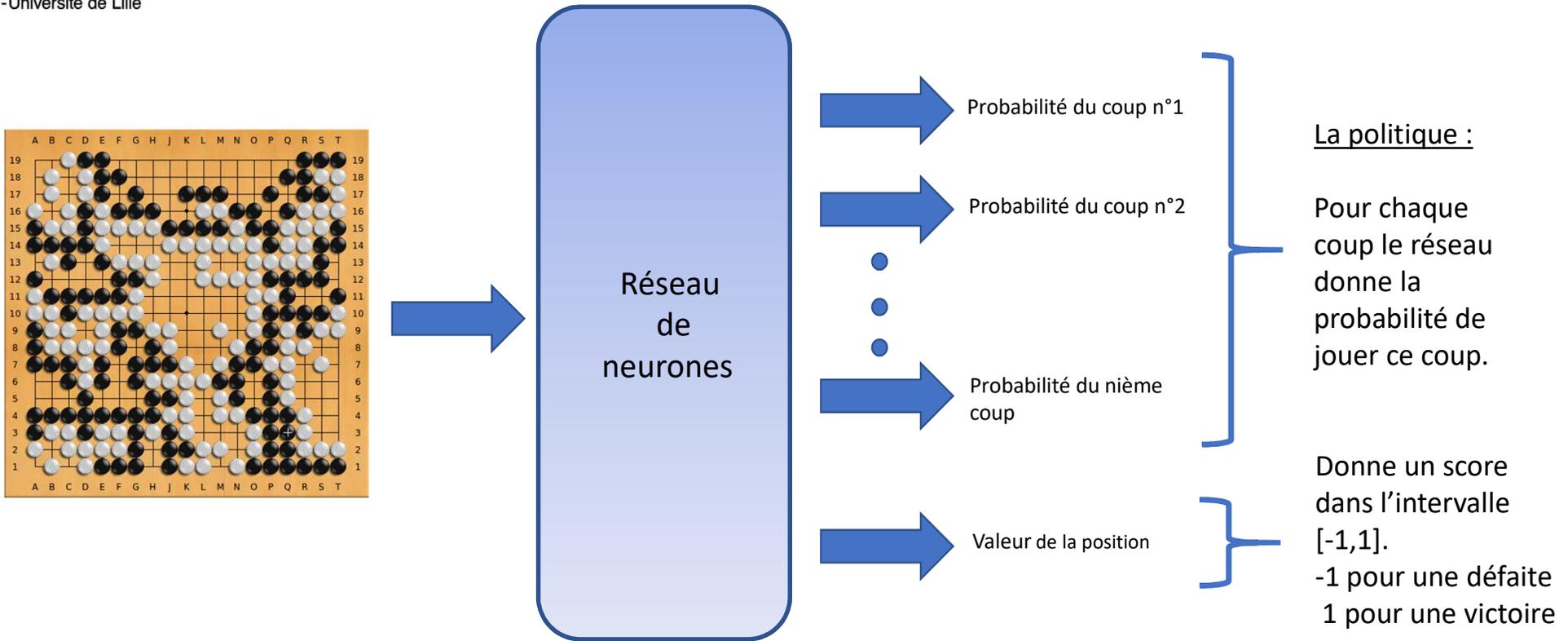
#IMTNordEurope

- Les parties sont jouées de manières aléatoires.
Beaucoup de parties jouées sont donc peu réalistes.
- Une partie est jouée jusqu'au bout pour avoir une évaluation de la position.

Réseau de neurones
couplé avec le MCTS



- Utilisation d'un réseau de neurones pour :
 - Aider l'algorithme MCTS à jouer des parties réalistes en ne sélectionnant plus au hasard les coups.
 - Donner une évaluation de la position du jeu sans devoir jouer la partie jusqu'à son terme.



Comment faire apprendre le réseau de neurones ?

Alpha Go : Première version de Deepmind couplant un MCTS et les réseaux de neurones. C'est cette version qui à battu Lee Sedol.

- Apprentissage du réseau de neurones à partir de millions de parties jouées par des experts.
- 2 réseaux : un pour donner les probabilités de chaque coup et un autre pour donner la valeur de la position.

Comment faire apprendre le réseau de neurones ?

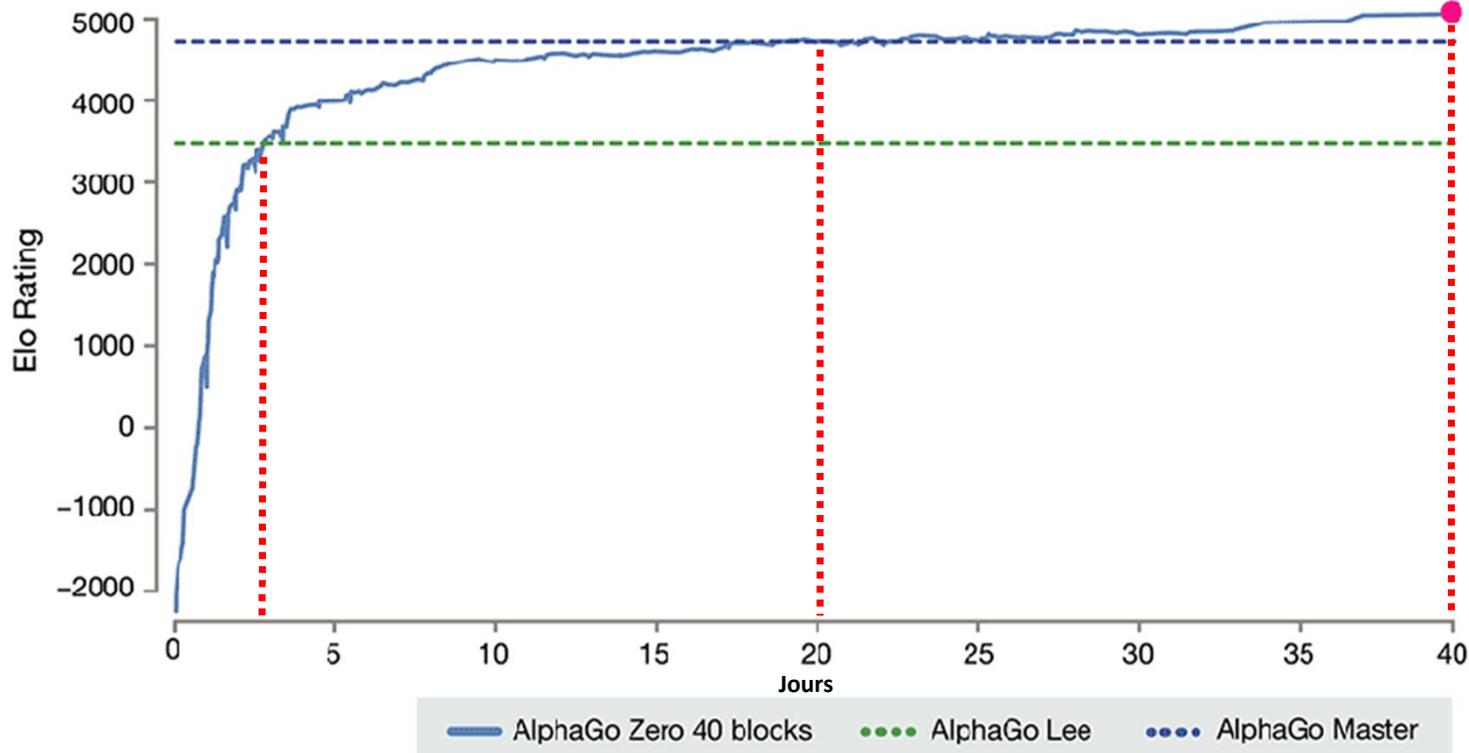
Alpha Zéro : Version améliorée d'Alpha Go.

- Apprentissage du réseau de neurones en jouant des parties contre lui-même.
- Un seul réseau pour les probabilités des coups et la valeur de la position.



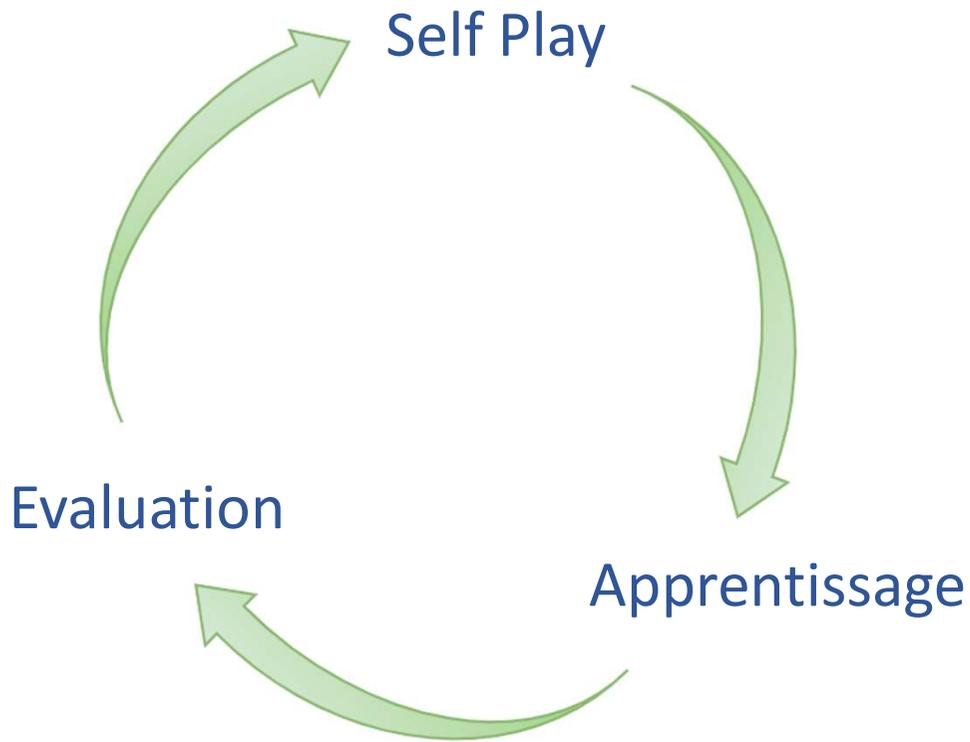
IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Alpha Zéro



Alpha Zéro joue mieux en apprenant par lui-même qu'en apprenant à partir de parties jouées par des experts.

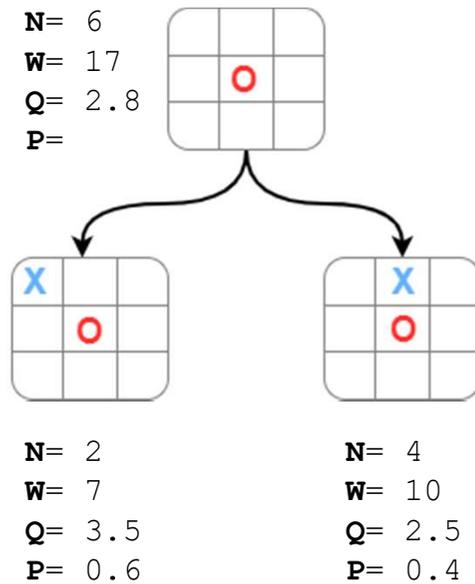
Cette manière d'apprendre lui à permis de découvrir de nouvelles stratégies.



Les trois étapes s'enchainent jusqu'à obtenir une IA d'un niveau suffisant.

Self Play

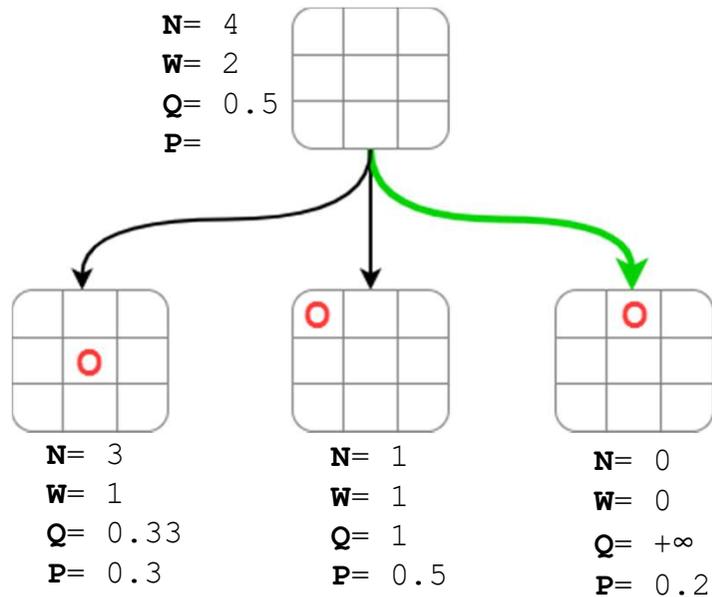
Constitution du jeu de données.



L'IA utilise l'algorithme MCTS pour jouer des parties contre elle-même.

Il y a 4 indicateurs associés à chaque nœud :

- N:** Nombre de fois que ce coup à été joué.
- W:** Score associé à ce coup.
- Q:** W/N
- P:** Probabilité associée à ce coup.



Phase de sélection du MCTS

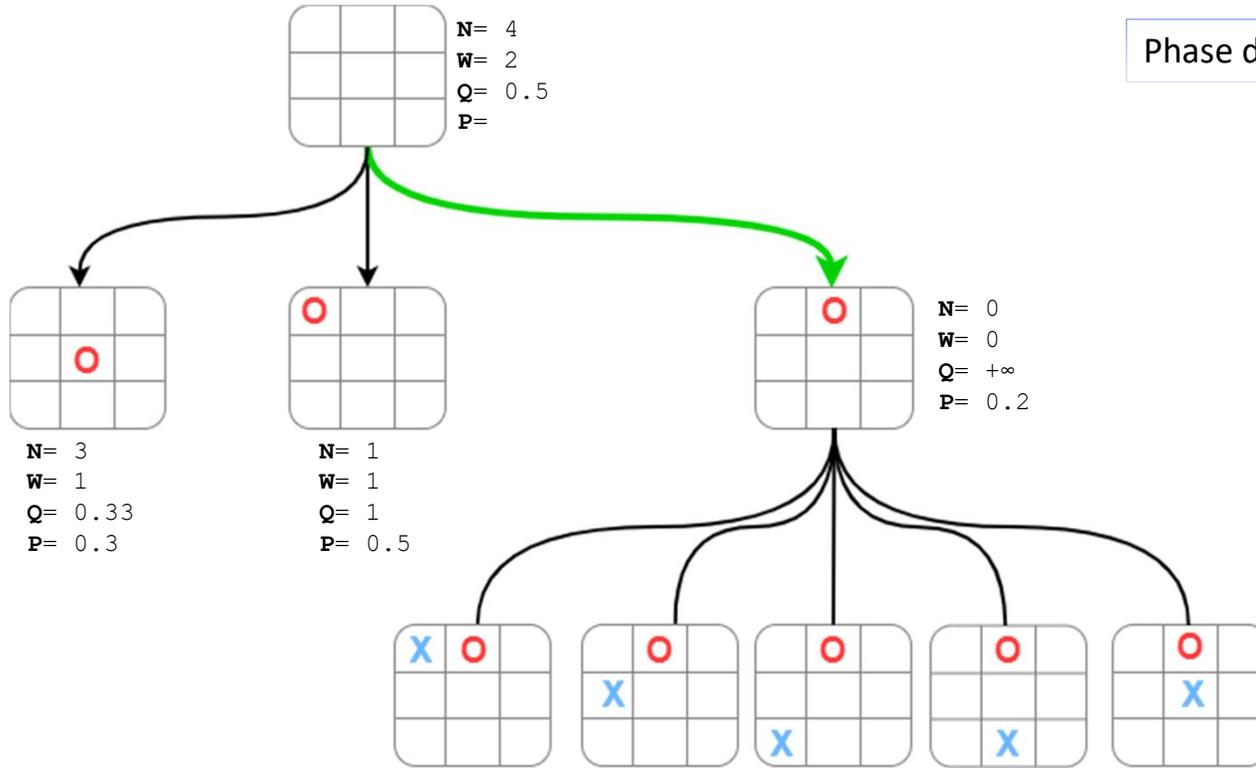
Sélectionne l'action qui maximise :

$$\underbrace{Q}_{\text{Exploitation}} + \underbrace{U(P, N)}_{\text{Exploration}}$$

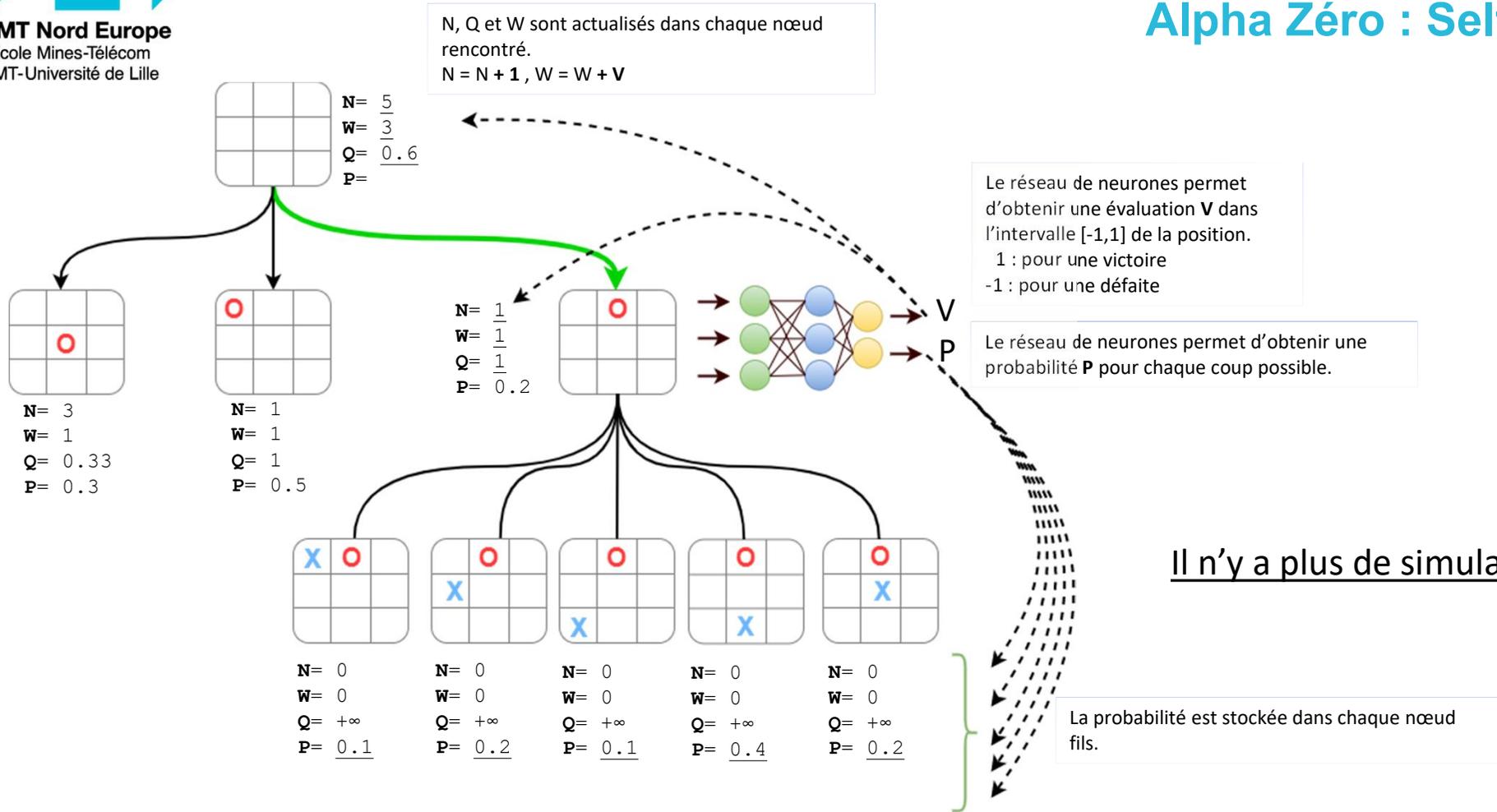
Q : Score moyen de la prochaine position.

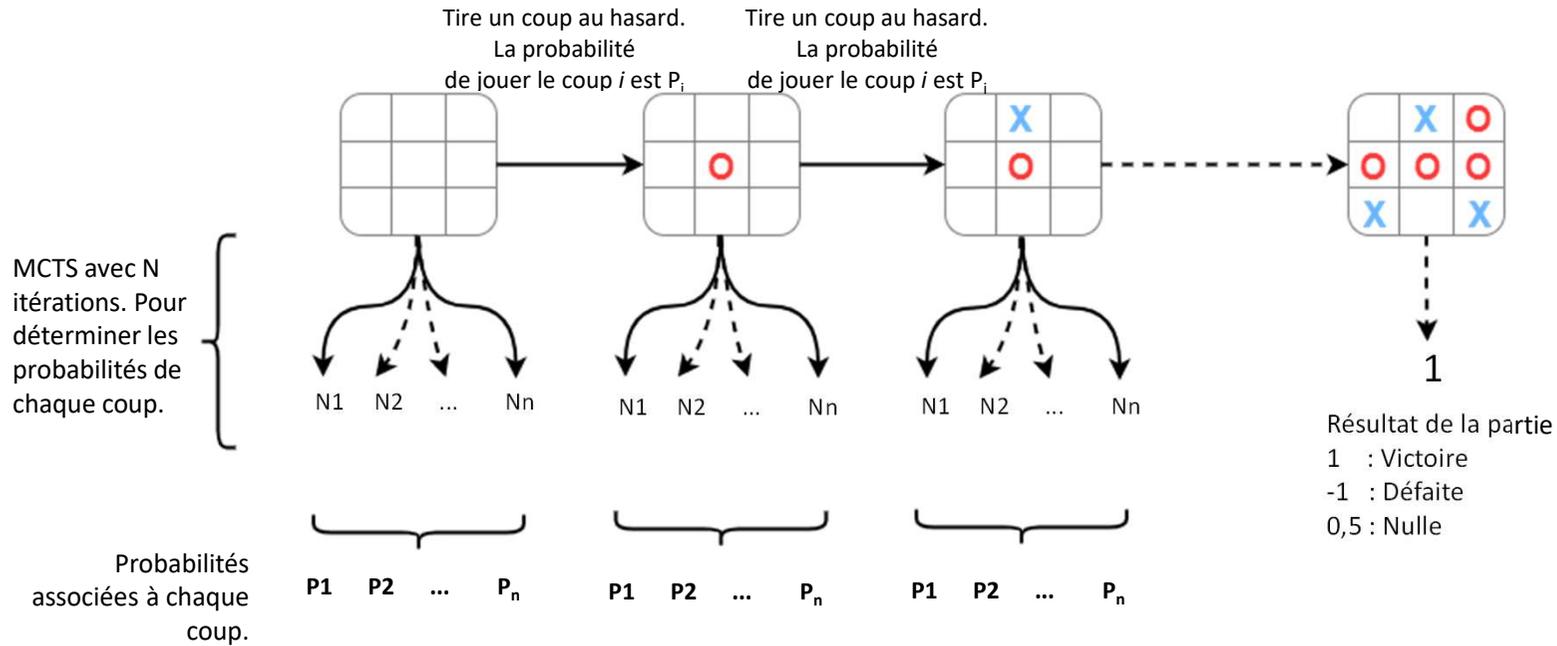
U : Une fonction dont la valeur augmente si un coup a été peu joué et si la probabilité du coup est élevée.

$$\underbrace{Q(s, a) + C * P(s, a)}_{Q} + \underbrace{\frac{\sqrt{N_{pere}}}{1 + N_{fils}}}_{U(P, N)}$$



Phase d'expansion du MCTS





$$P_i = \frac{N_i^{\frac{1}{t}}}{\sum_k N_k^{\frac{1}{t}}}$$

Une température t faible accorde plus d'importance au coup le plus souvent joué.
 Une température t élevée va rendre chaque coup de plus en plus équiprobable.
 Ce paramètre permet donc de faire varier le niveau d'exploration lors du self play.



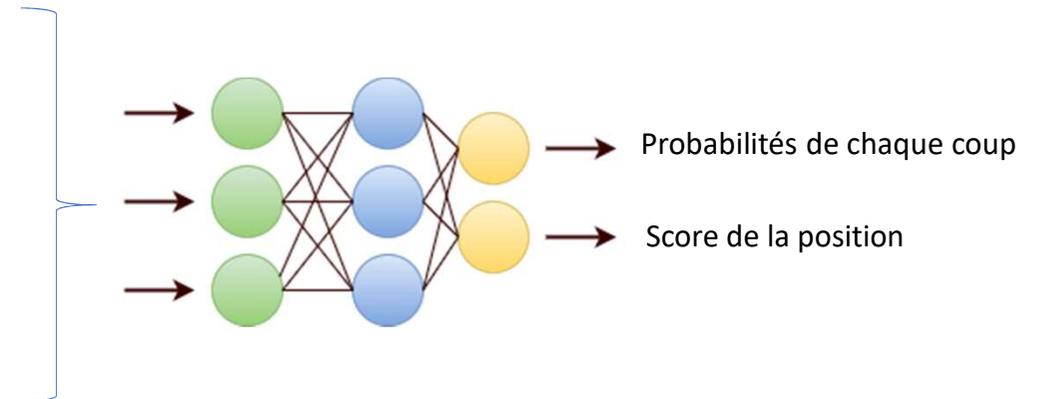
IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Apprentissage du réseau de neurones



Positions	Probabilités de chaque coup. Obtenues avec le MCTS	Score Résultat de la partie
	P1,P2,....,	1
	P1,P2,....,	1
	P1,P2,...	1
...	P1,P2,...	1

Jeu de données utilisé pour l'apprentissage du réseau de neurones.





IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

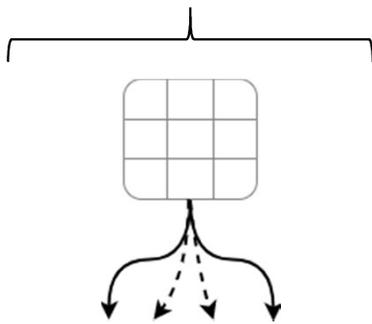
Evaluation du réseau de neurones





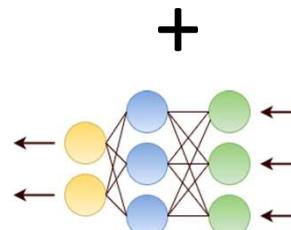
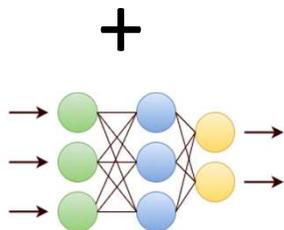
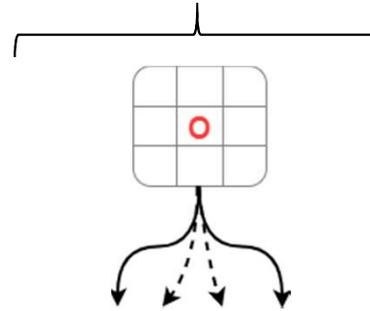
IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

MCTS utilisant le meilleur
réseau de neurones actuel.



VS

MCTS utilisant le
réseau de neurones issu
de l'apprentissage précédent.



Alpha Zéro : Evaluation

Plusieurs parties sont jouées. Le réseau de neurones issu de l'apprentissage précédent devient le nouveau réseau de neurones de référence s'il gagne au moins 55% des parties.

Pendant ces parties, l'IA joue les coups ayant le nombre de simulations N le plus élevé à l'issue du MCTS.