

# Intelligence artificielle

# But

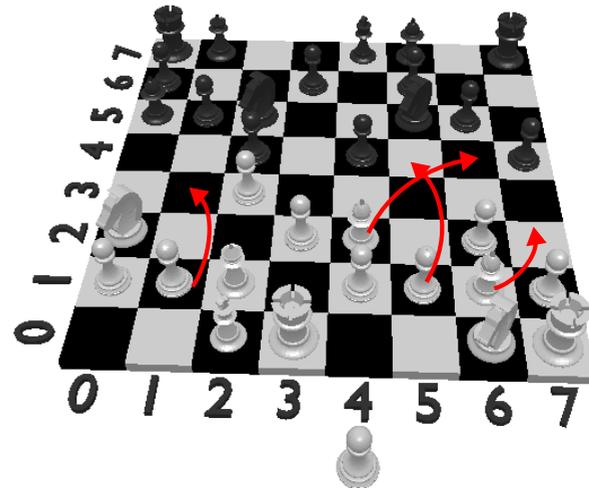
- Savoir quel prochain coup jouer

Possibilité

→ Premier coup possible trouvé,  
Coup aléatoire, ...

*Bon coup ?*

- Qu'est ce qu'un bon coup ?



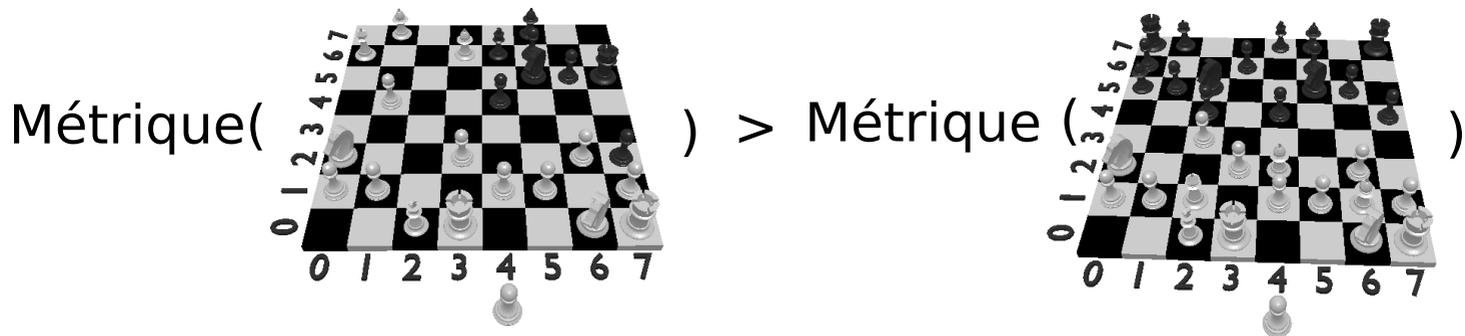
# But

- Qu'est ce qu'un bon coup ?  
Un coup qui donne un "bon" jeu

Il faut pouvoir quantifier un jeu

↖ Mettre en place une métrique

Métrique standard triviales sur un jeu d'echec:  
- nombre de pièce x coefficient



# Métriques

Plusieurs possibilités:

- Valeur pièces amies - Valeur pièces ennemies
- Valeur pièces amies / Valeur pièces ennemies
- Prise en compte de configurations connues

*Procédural  
Simple*

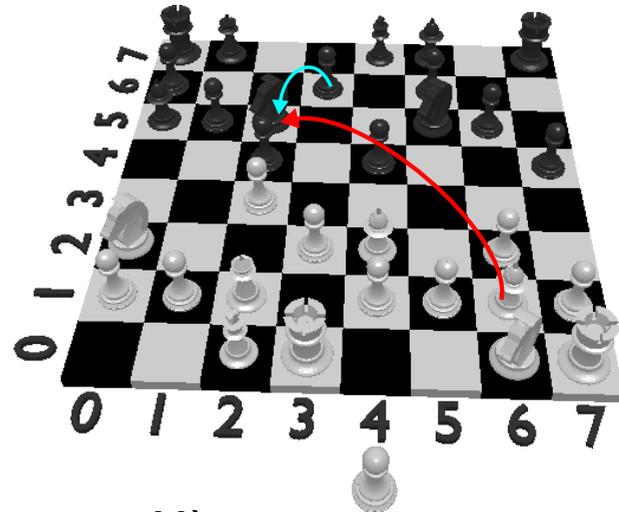
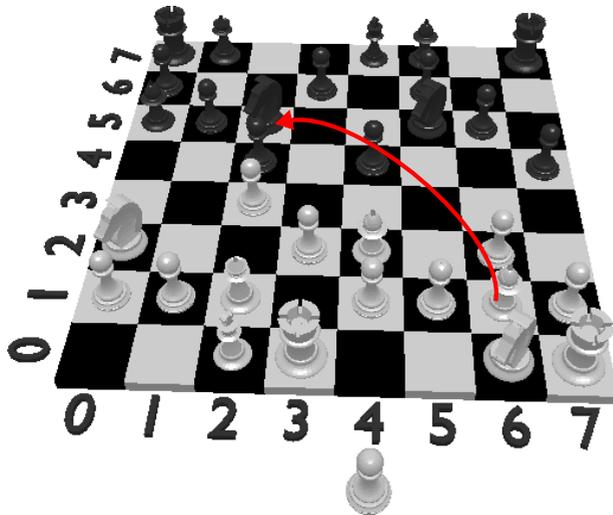
*Base de données  
Apprentissage*



valeur = 45.3

# Que rechercher ?

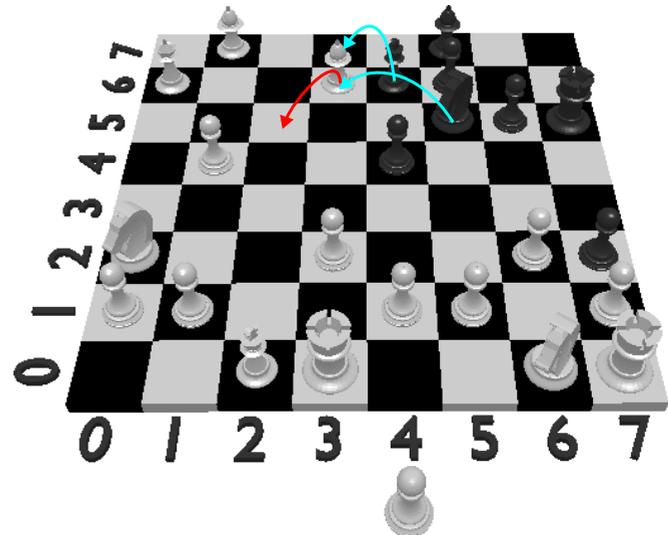
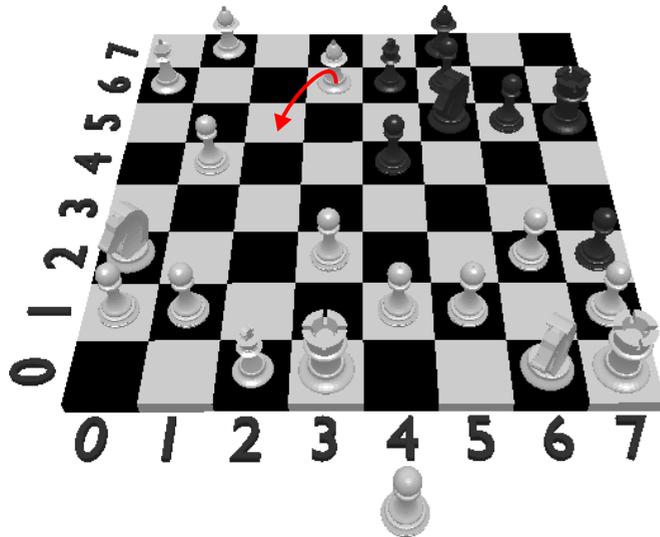
- Le coup qui maximise mon score ?



**Problème:**  
Coup adverse suivant reprend  
l'avantage

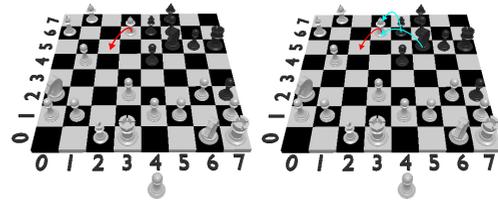
# Que rechercher ?

- Le coup qui complique la vie de l'adversaire  
c.a.d. Coup tel que coup adverse soit le moins bon:



# Que rechercher ?

- Le coup qui complique la vie de l'adversaire  
c.a.d. Coup tel que coup adverse soit le moins bon:



Trouver le coup tel que

- le meilleur coup adverse possible soit celui qui me pénalise le moins

*maximisation*

*minimisation*

- Algorithmes de type min-max

# Optimisation min-max

- Soit  $\mathcal{C}_0$ , l'ensemble des coups possibles.  $c_0$ , un coup de  $\mathcal{C}_0$
- Soit  $\mathcal{C}_1(c_0)$ , l'ensemble des coups adverses possibles après avoir joué un coup  $c_0$ . Soit  $c_1$ , un coup de  $\mathcal{C}_1(c_0)$ .
- Soit  $j(c_0, c_1)$  l'état de l'échiquier après avoir joué un coup  $c_0$  et  $c_1$ . Soit  $\sigma$  la métrique associée à un état d'échiquier  $j$ .

Le meilleur coup est le coup  $c_0^*$  tel que

$$\min_{c_1 \in \mathcal{C}_1(c_0^*)} \sigma(j(c_0^*, c_1)) \text{ soit maximal.}$$

En d'autres termes

$$c_0^* = \arg \max_{c_0 \in \mathcal{C}_0} \min_{c_1 \in \mathcal{C}_1(c_0)} \sigma(j(c_0, c_1)) .$$

# Optimisation min-max

Algorithme:

$c_0^* \leftarrow 0$

$s_{\min\_max} \leftarrow 0$

Pour tous les coups (amis)  $c_0$  possibles

    métrique  $s_{\min} \leftarrow +\infty$

    Pour tous les coups  $c_1$  ennemis possibles

$s = \text{Calculer métrique courante}$

$s_{\min} = \min(s, s_{\min})$

    Si  $s_{\min\_max} < s_{\min}$

$c_0^* = c_0$

$s_{\min\_max} = s_{\min}$

# Optimisation min-max

## Algorithme:

```
c0* <- 0
```

```
s_min_max <- 0
```

```
Pour tous les coups (amis) c0 possibles
```

```
  métrique s_min <- +inf
```

```
  Pour tous les coups c1 ennemis possibles
```

```
    s=Calculer métrique courante
```

```
    s_min=min(s,s_min)
```

```
  Si s_min_max < s_min
```

```
    c0*=c0
```

```
    s_min_max=s_min
```

## 2 Fonctions:

- Parcours des coups possibles
- Calcul de métrique pour un échiquier donné

# Profondeur de recherche

Recherche sur le prochain coup est limité

- Beaucoup de coups "égaux"
- Minimum/Maximums locaux

=> Recherche sur plus de coups

$j(c_0, c_1, c_2, c_3)$   $\longrightarrow$  Pour tous les coups  $c_0$   
 $j(c_0, c_1, c_2, c_3, \dots)$  Pour tous les coups  $c_1$   
Pour tous les coups  $c_2$   
Pour tous les coups  $c_3$

Théoriquement: converge/imbattable (-> solution optimale)

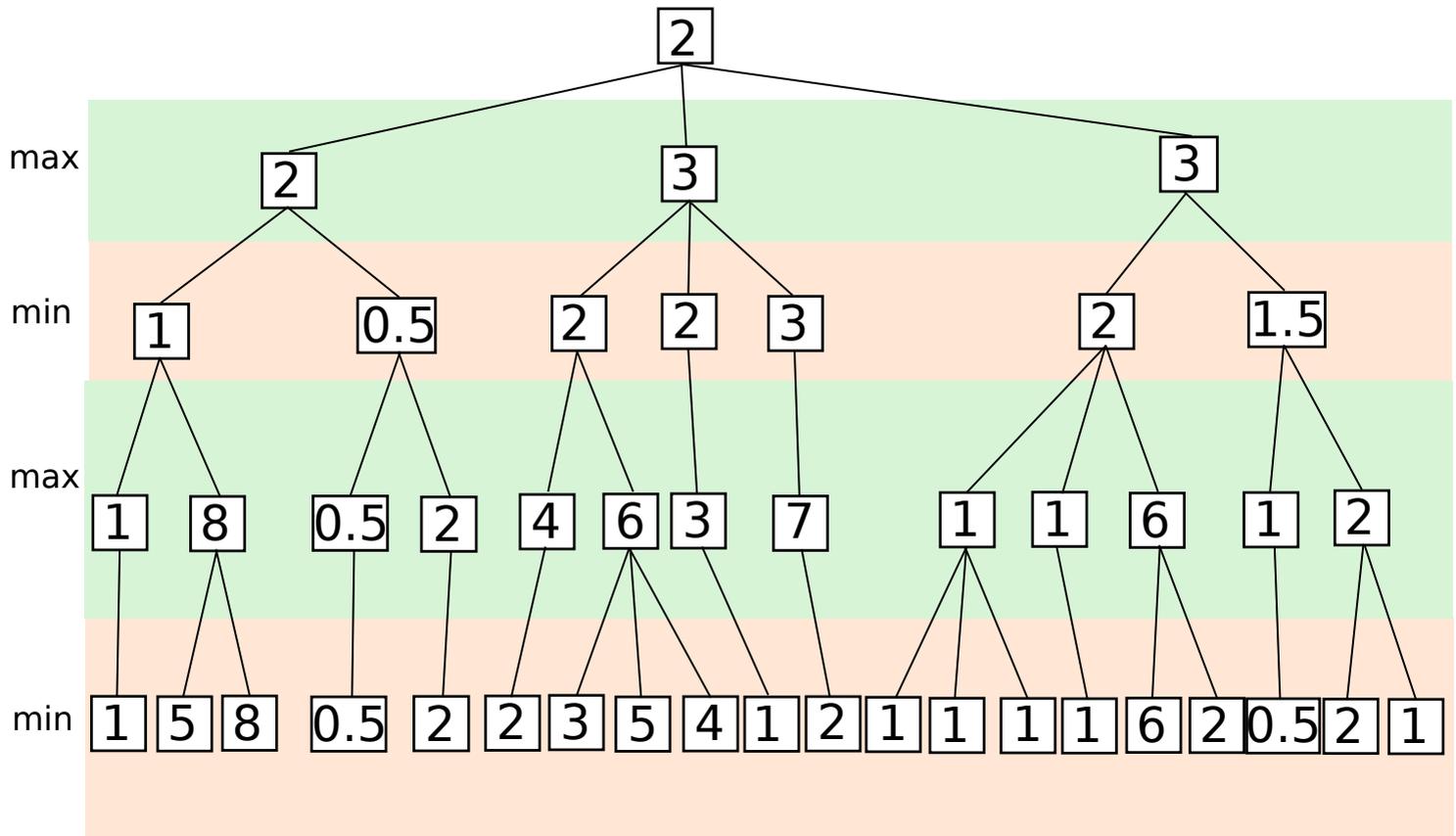
• Problème d'explosion combinatoire

~40 coups possibles par joueur

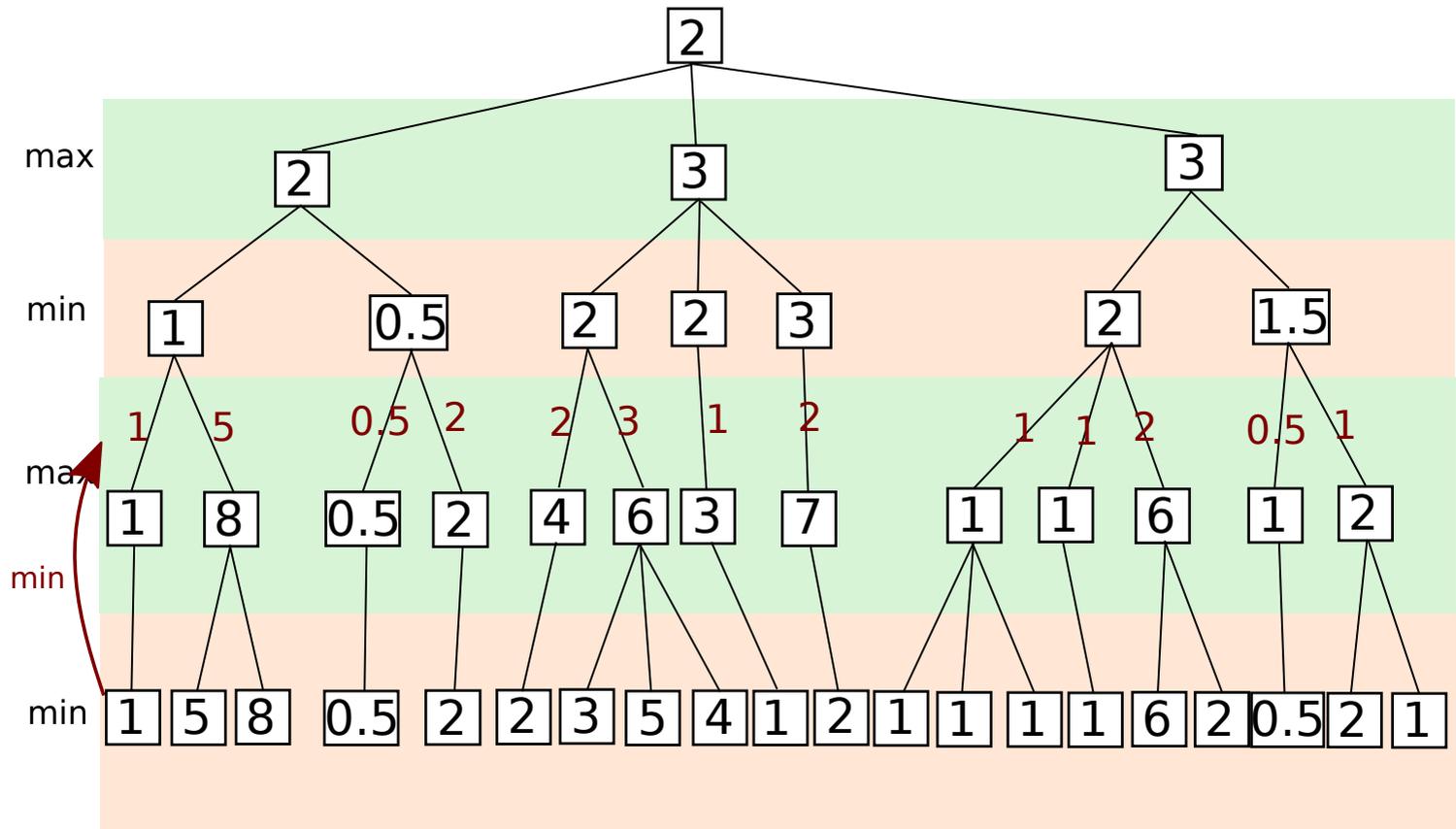
Niveau 2:  $40^4 \sim 2$  millions possibilités

Niveau 3:  $40^6 \sim 4$  milliards possibilités

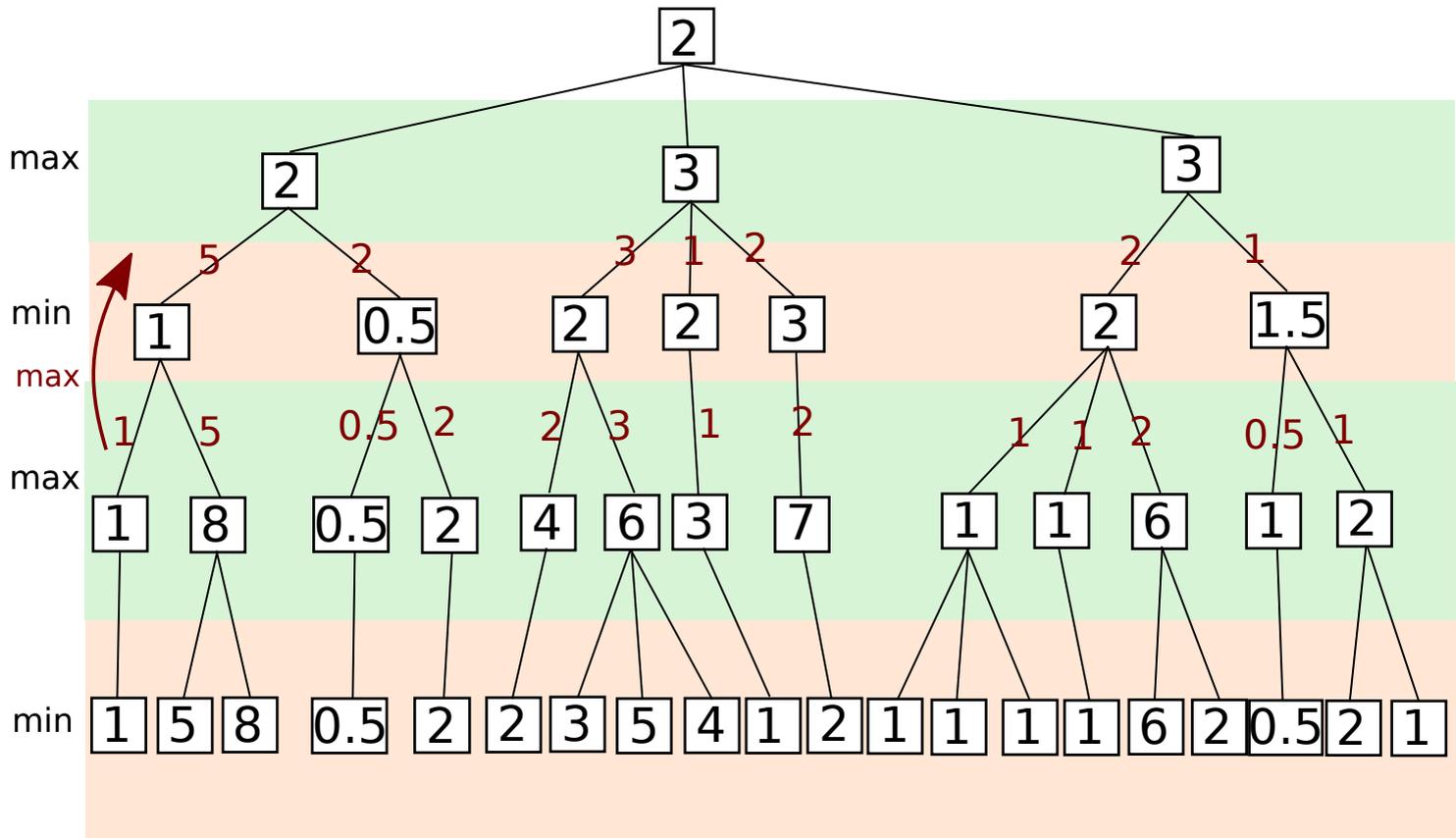
# Profondeur de recherche



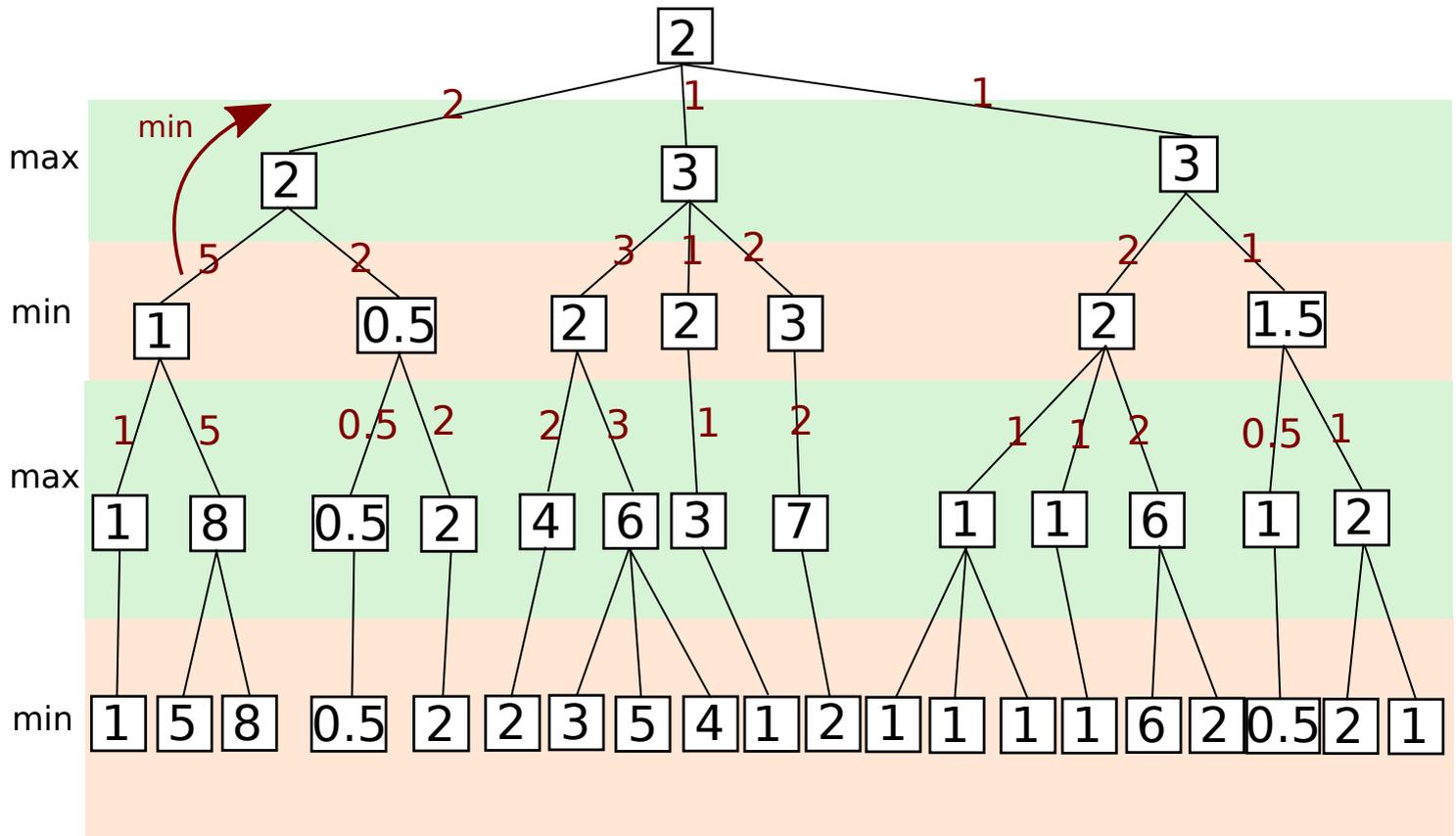
# Profondeur de recherche



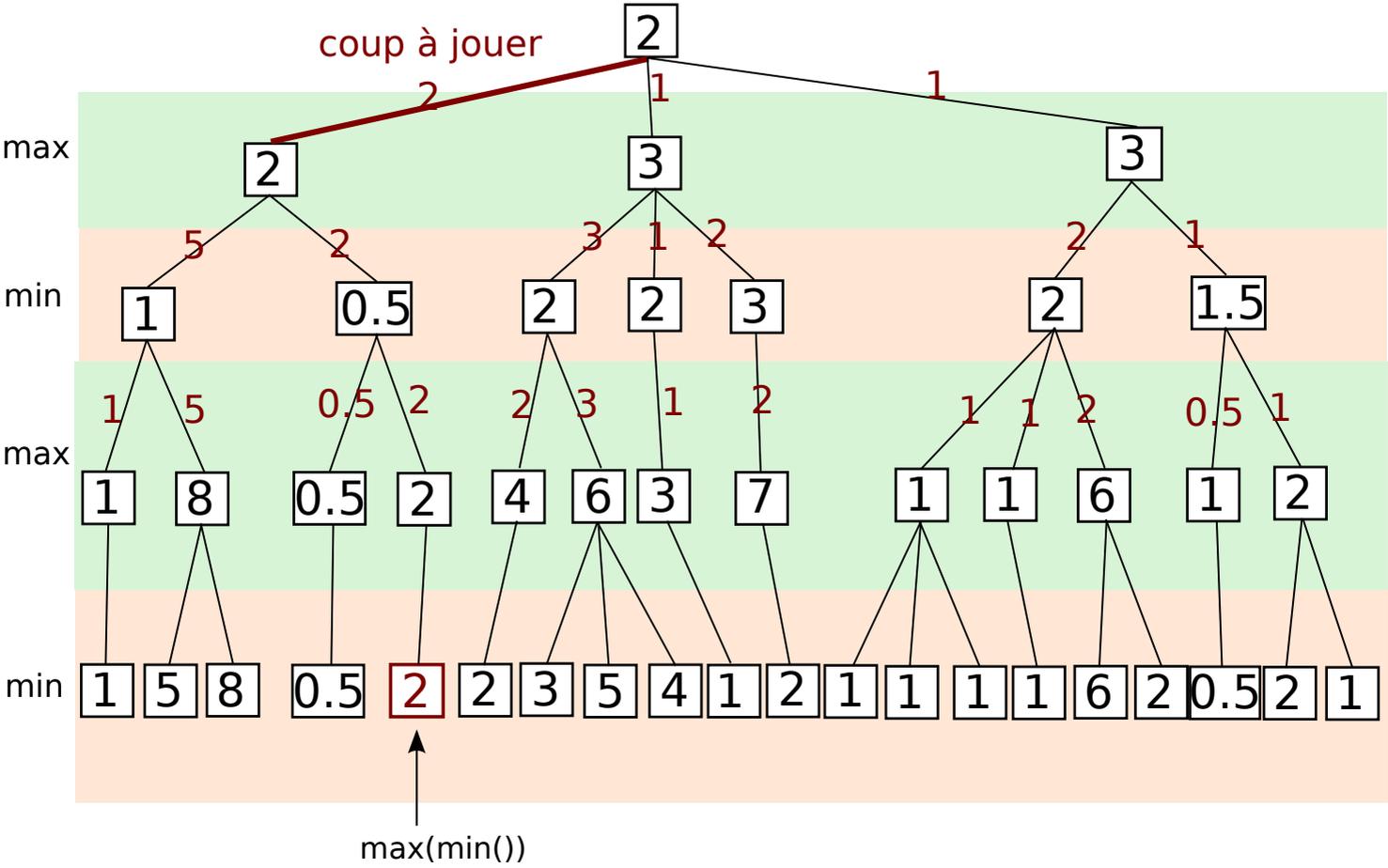
# Profondeur de recherche



# Profondeur de recherche



# Profondeur de recherche



# Optimisation

Elagage alpha/beta (/pruning)

