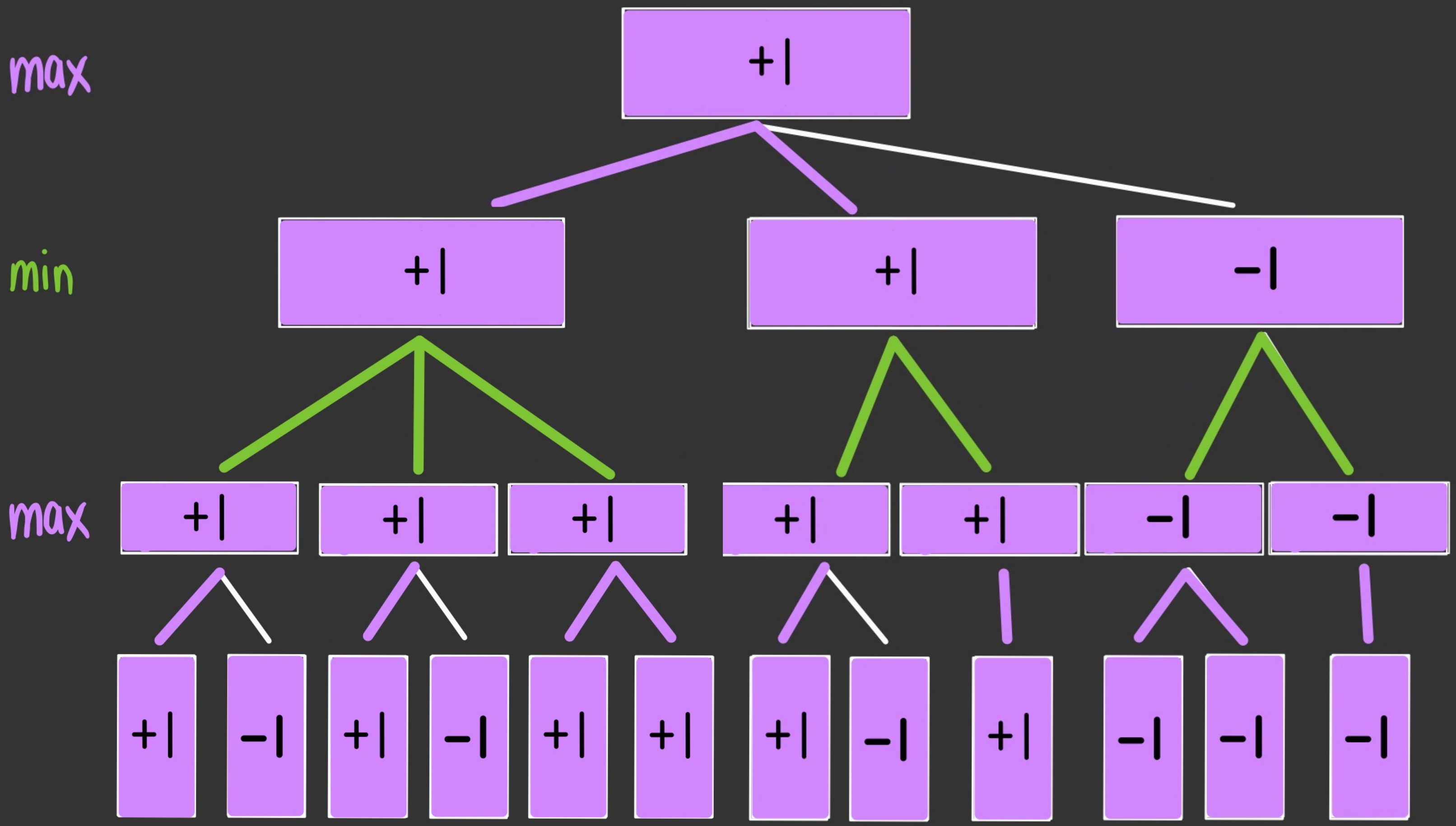
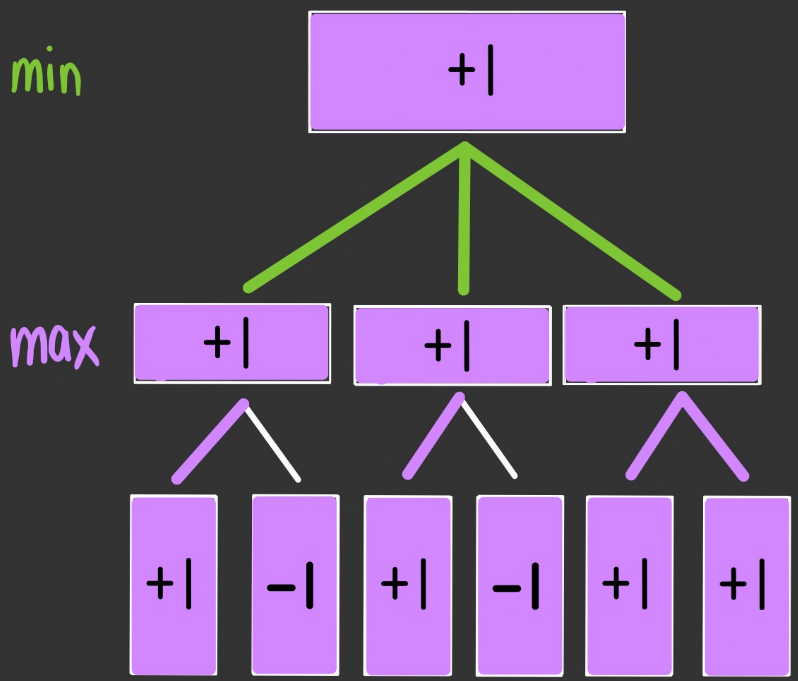
alpha-beta pruning

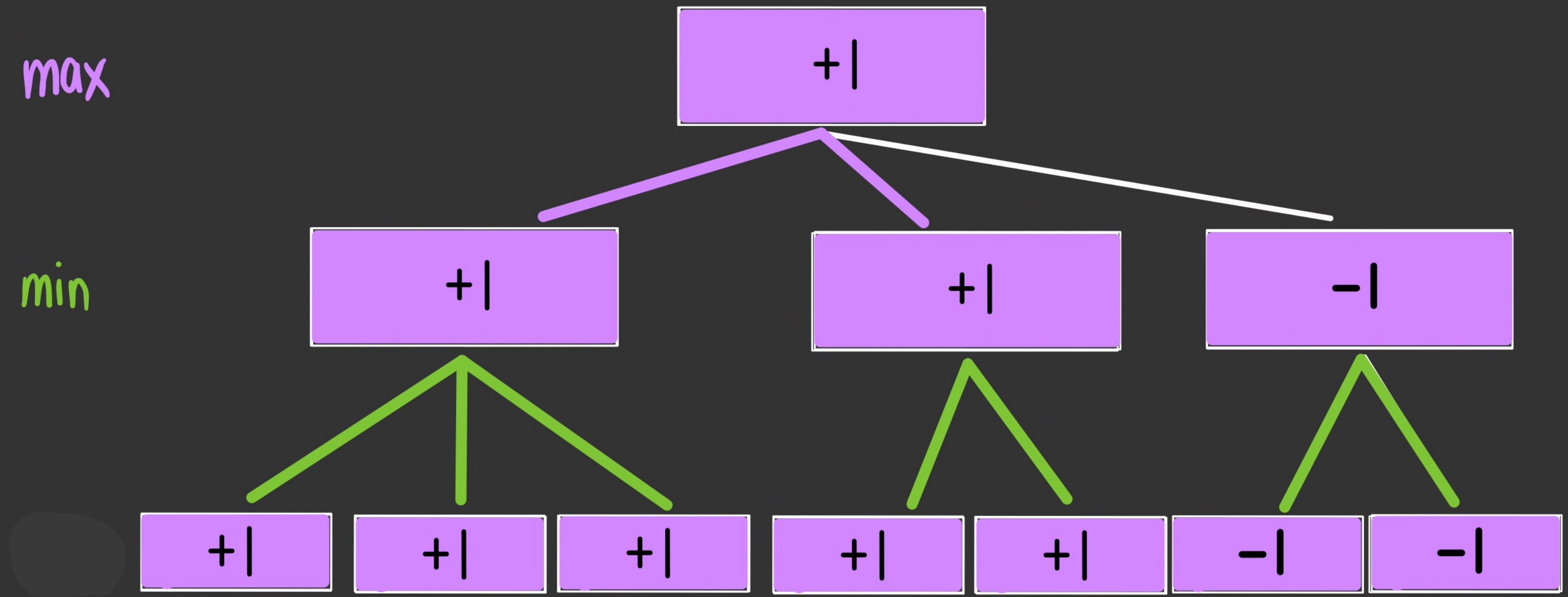
21 sept 2022

CSCI 373


max
min
max
+1
+1
+1
-1
+1
+1
+1
+1
+1
-1
-1
+1
-1
+1
-1
+1
+1
+1
-1
+1
-1
-1
-1



consider a min layer followed by a max layer



or a max layer followed by min layer

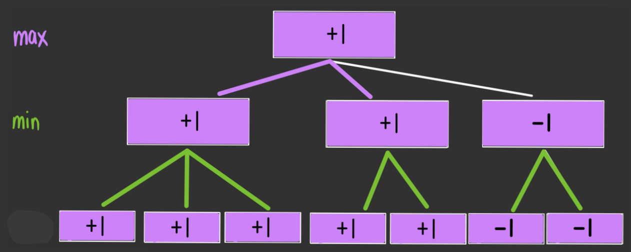
SCIENTIFIC AMERICAN MIND

BEHAVIOR

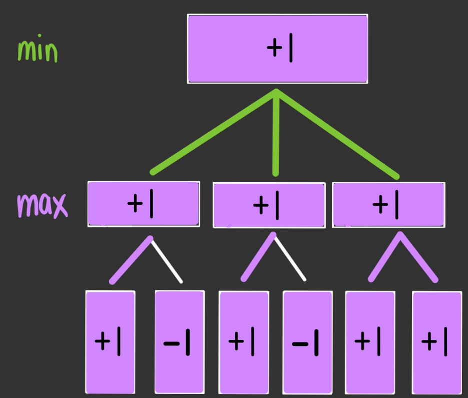
Are People Naturally Inclined to Cooperate or Be Selfish?

By Matthew Robison on September 1, 2014

It seems that human nature supports both prosocial and selfish traits. Genetic studies have made some progress toward identifying their biological roots. By comparing identical twins, who share nearly 100 percent of their genes, and fraternal twins, who share about half, researchers have found overwhelming evidence for genetic effects on behaviors such as sharing and empathy. In these twin studies, identical and fraternal twins are placed in hypothetical scenarios and asked, for example, to split a sum of money with a peer. Such studies often also rely on careful psychological assessments and DNA analysis.



max followed by min:
people are fundamentally
selfish



min followed by max:
people are fundamentally
selfless

max followed by min:

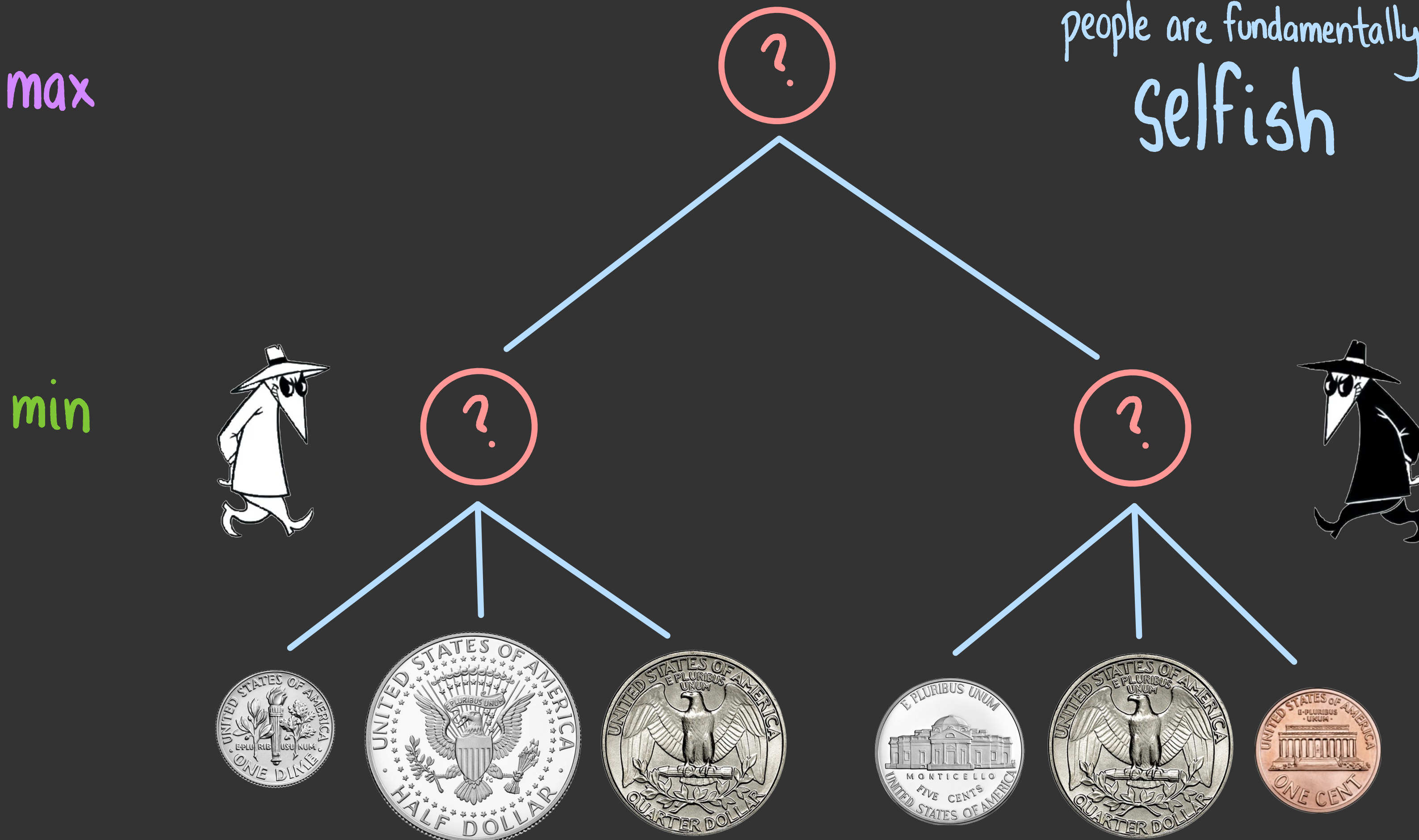
people are fundamentally

selfish

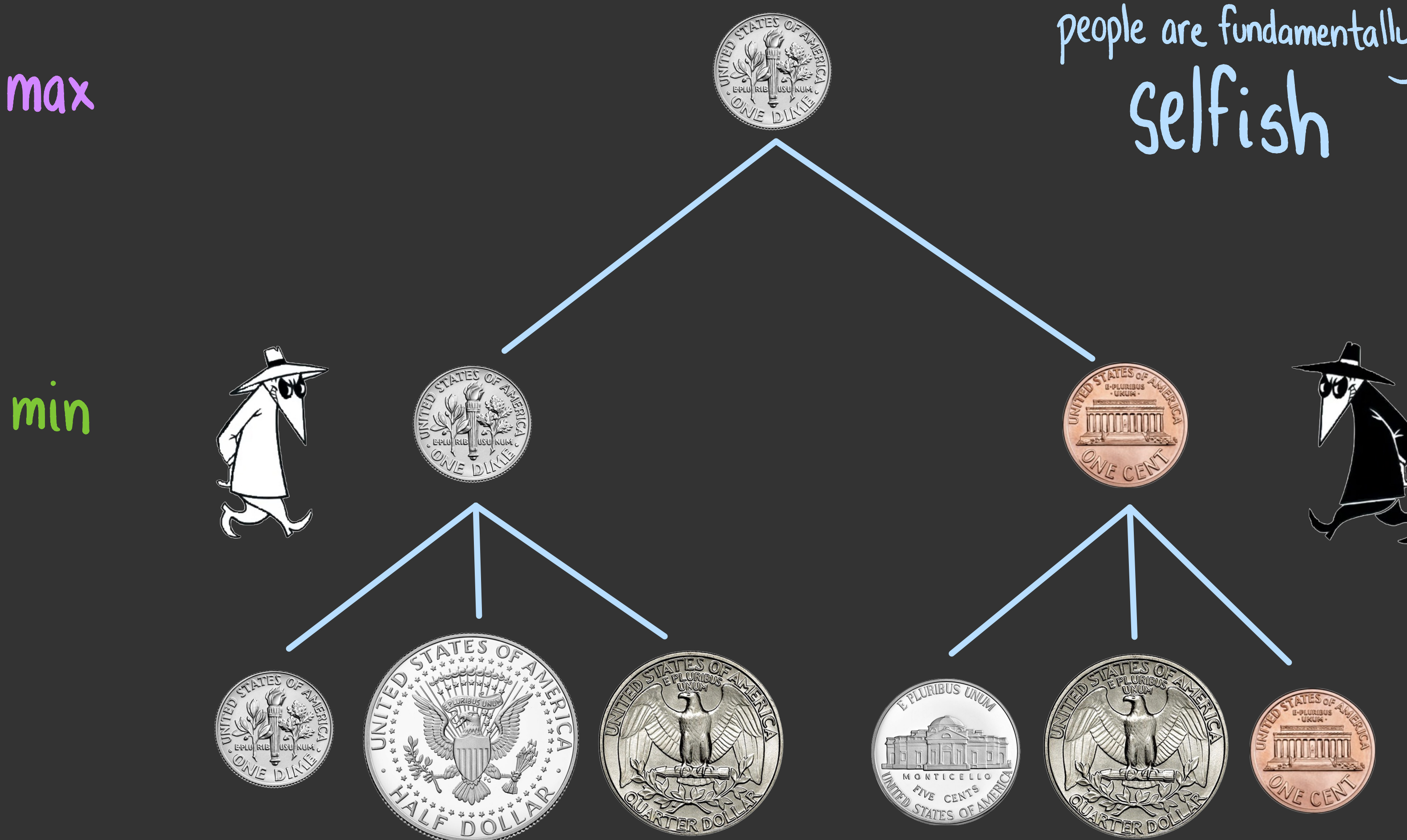
an edict comes down:

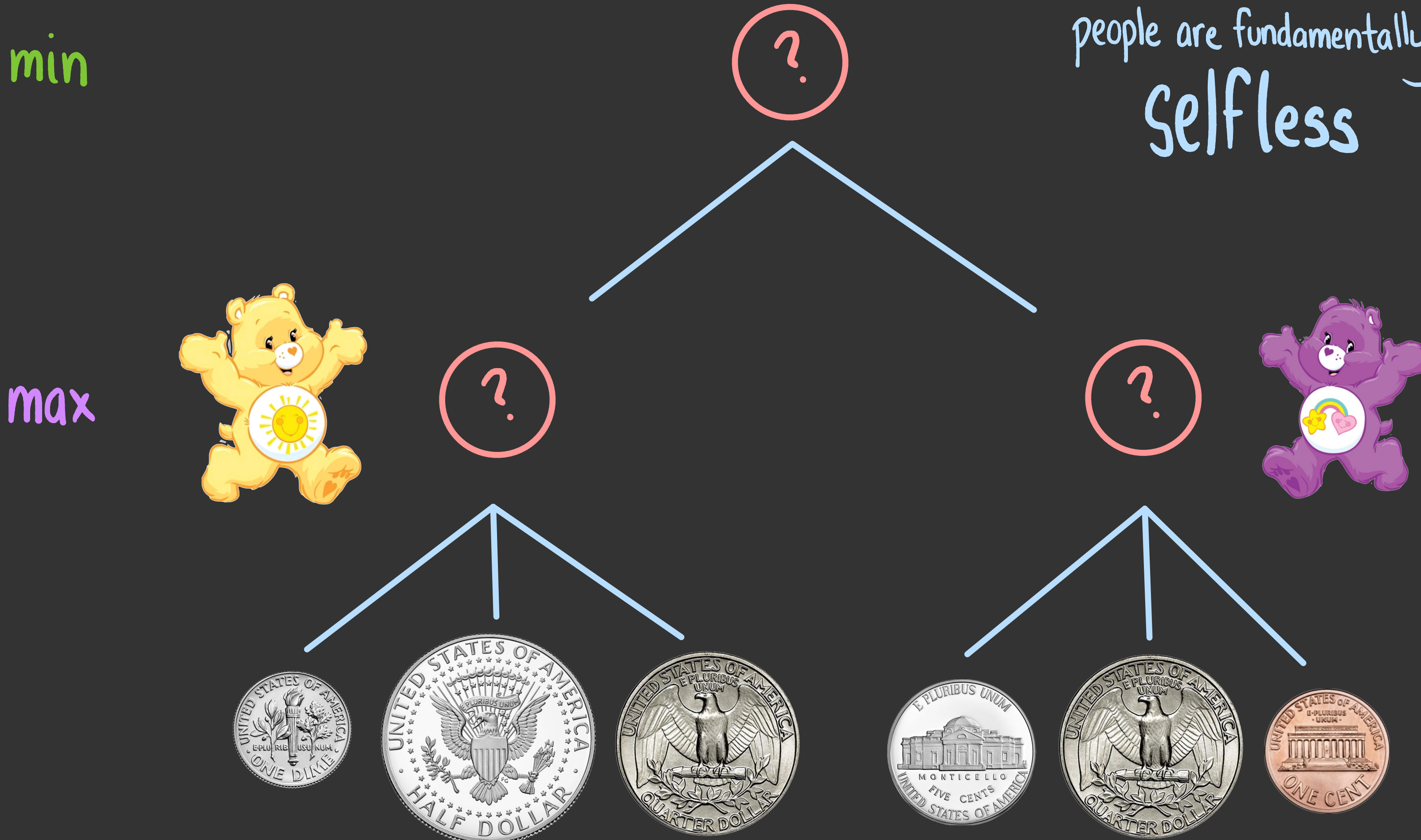
someone in the room

must give me a coin



max
min
?
?
?
people are fundamentally selfish

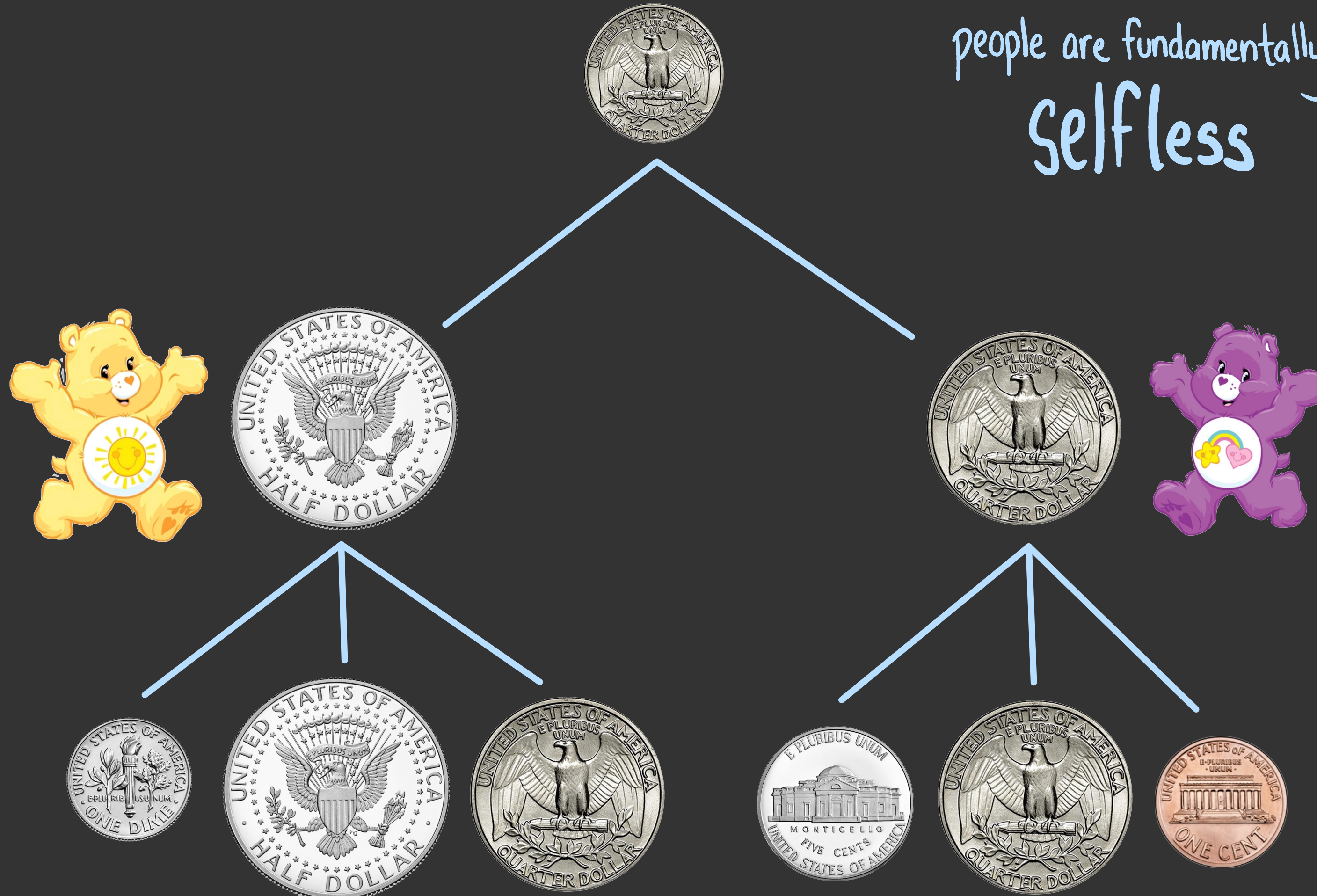

people are fundamentally
selfish
max
min

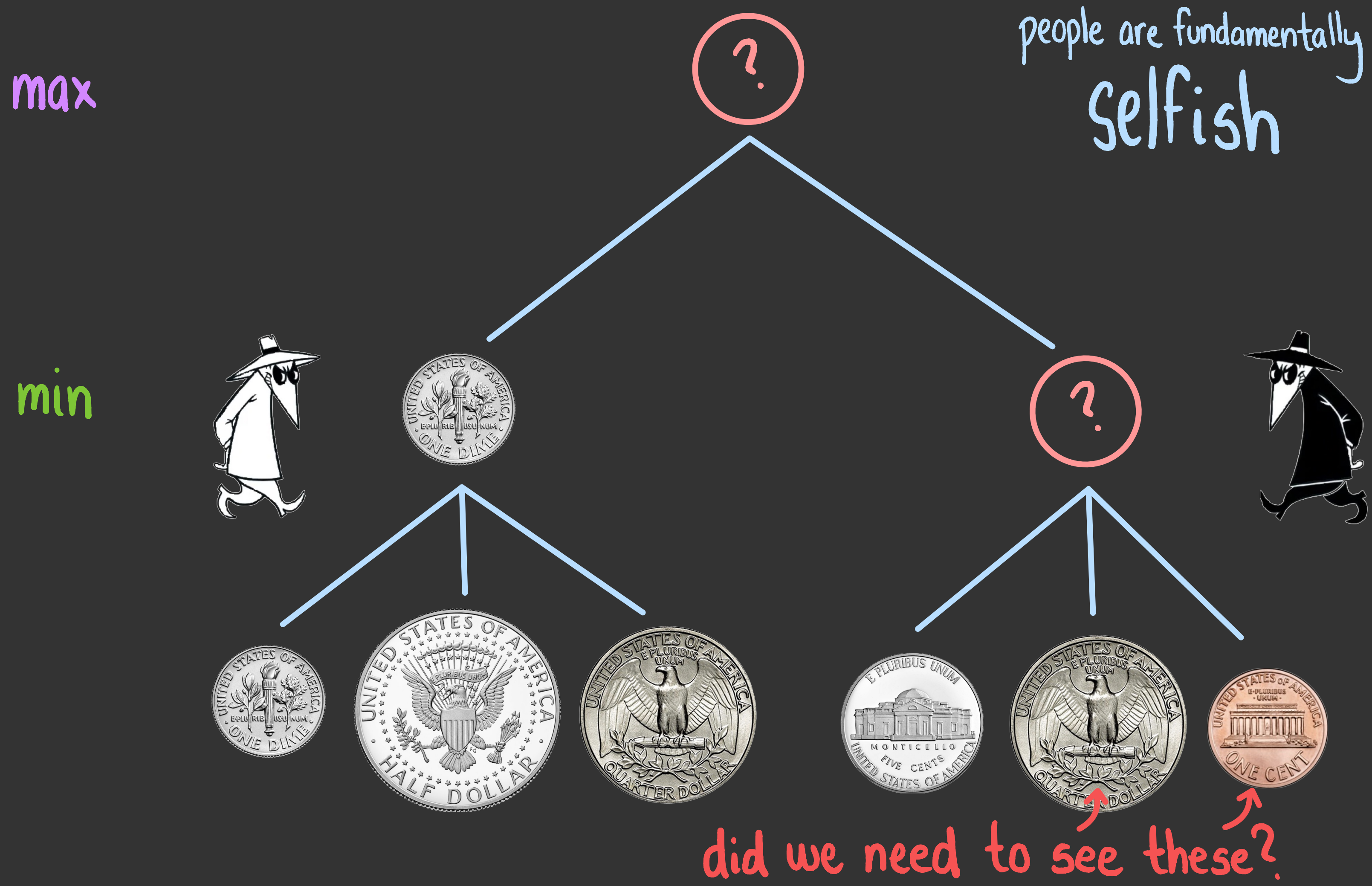

min
max
?
?
?
people are fundamentally
selfless

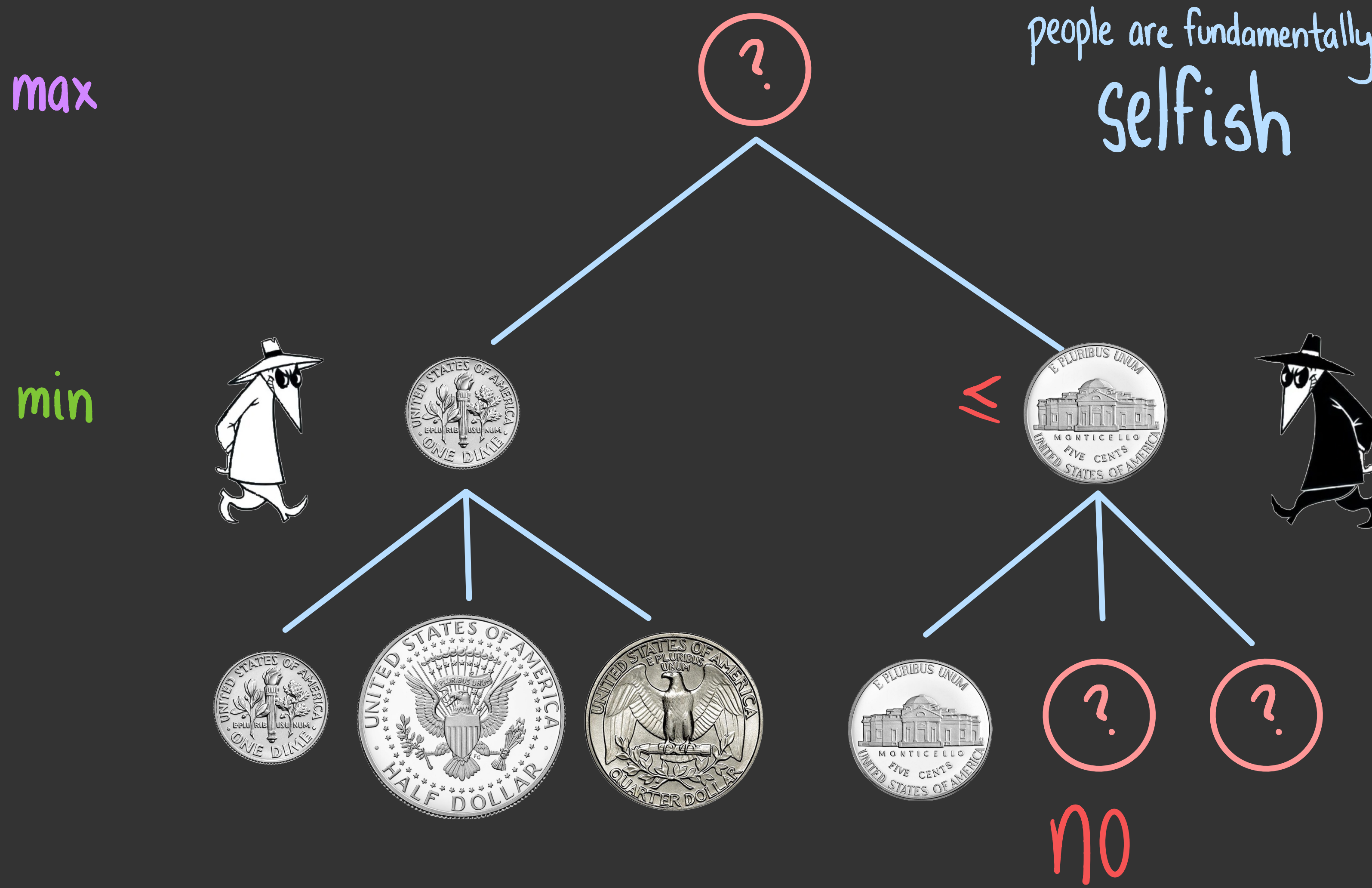
people are fundamentally selfless

min

max



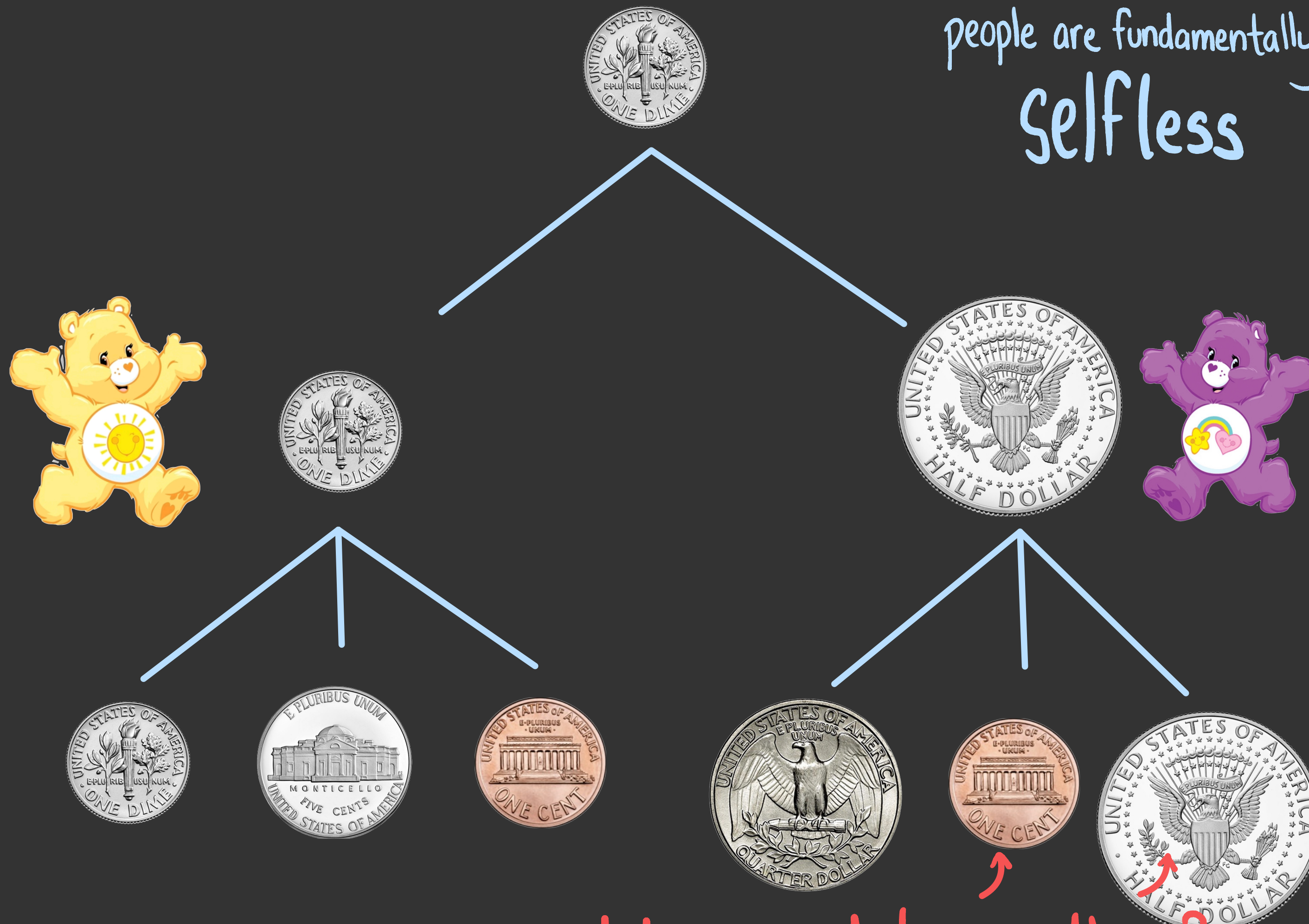

max
min
?
?
people are fundamentally
selfish
did we need to see these?



people are fundamentally
selfish
max
min
?
≤
?
?
no

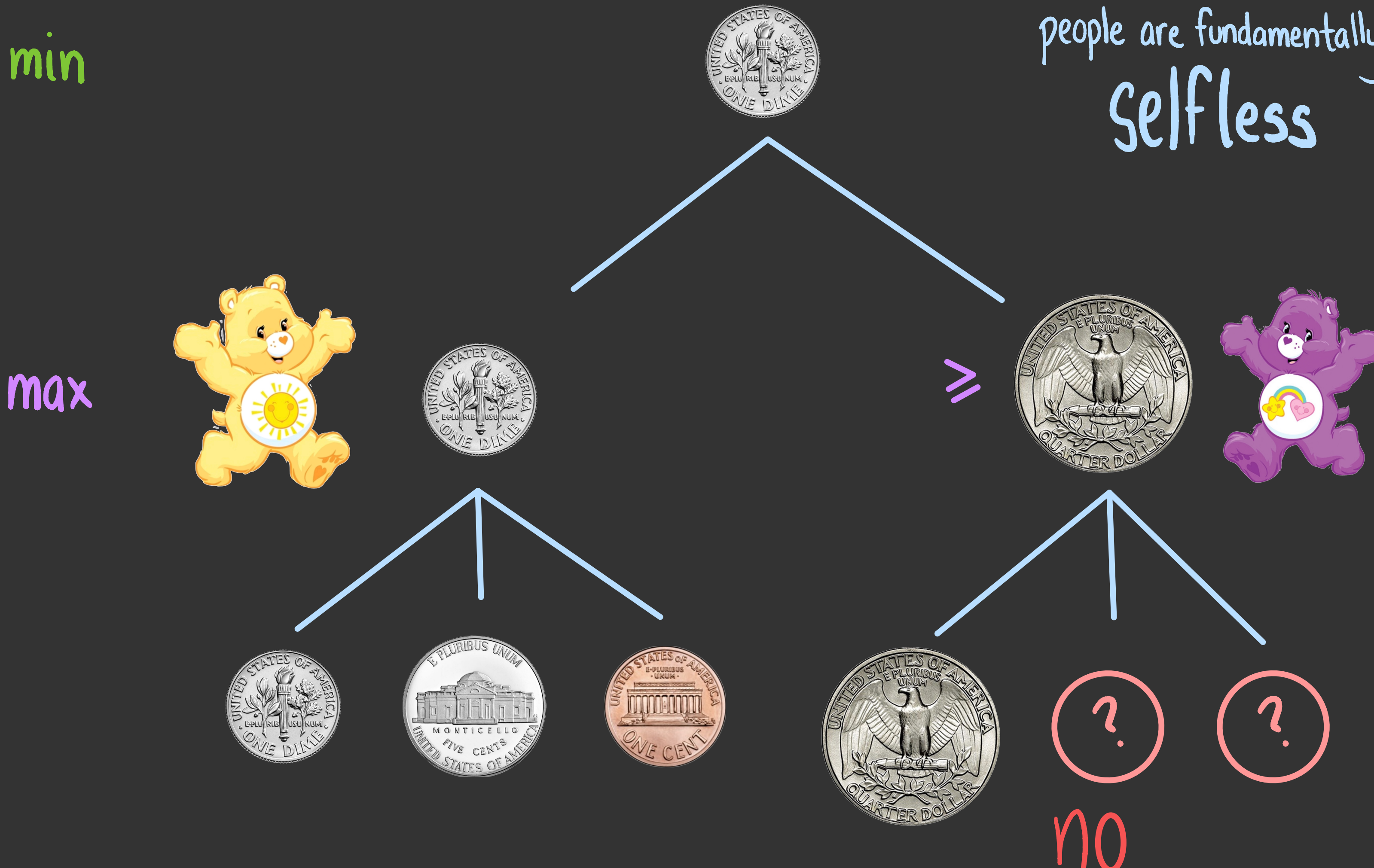
min

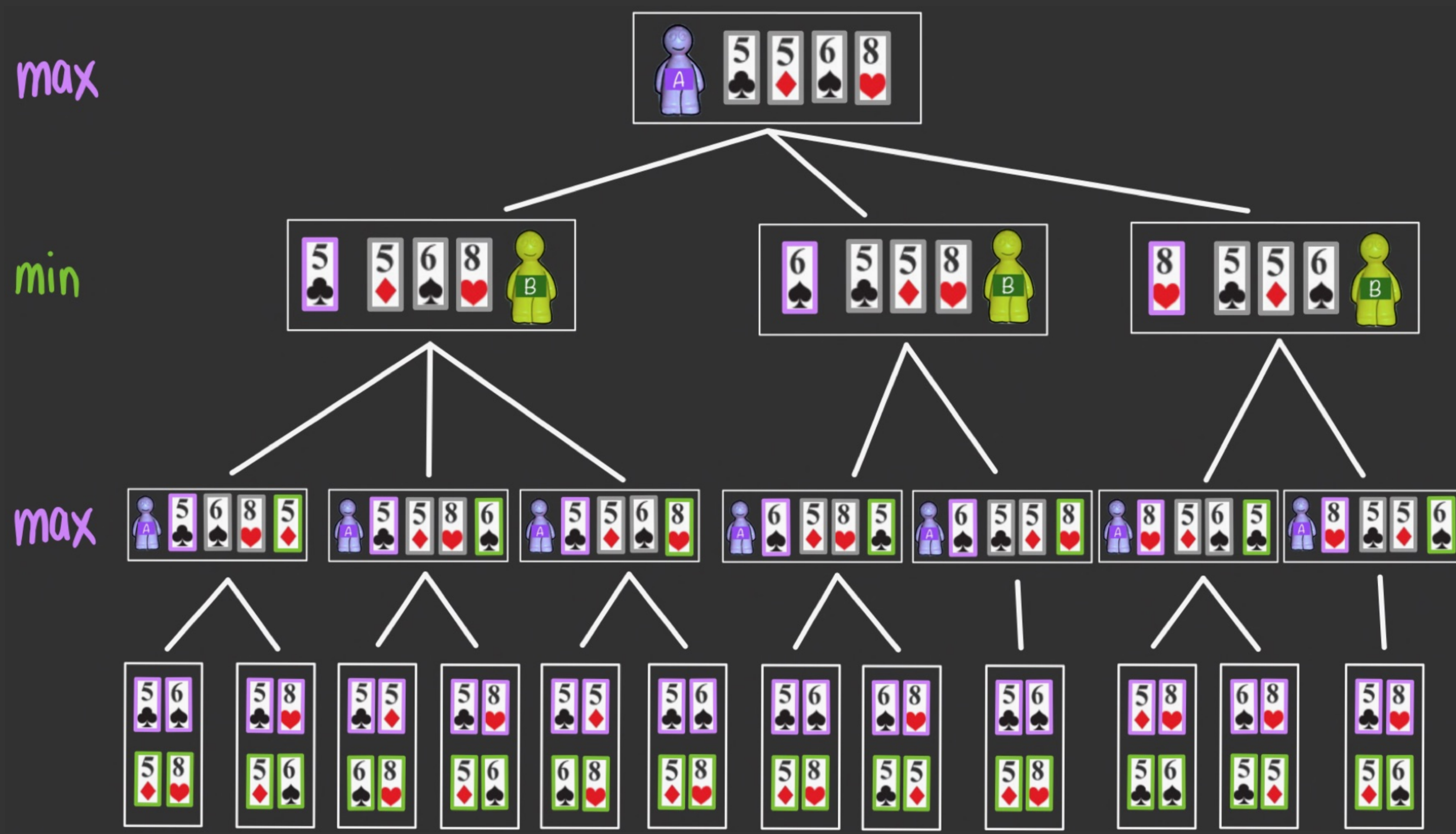
max

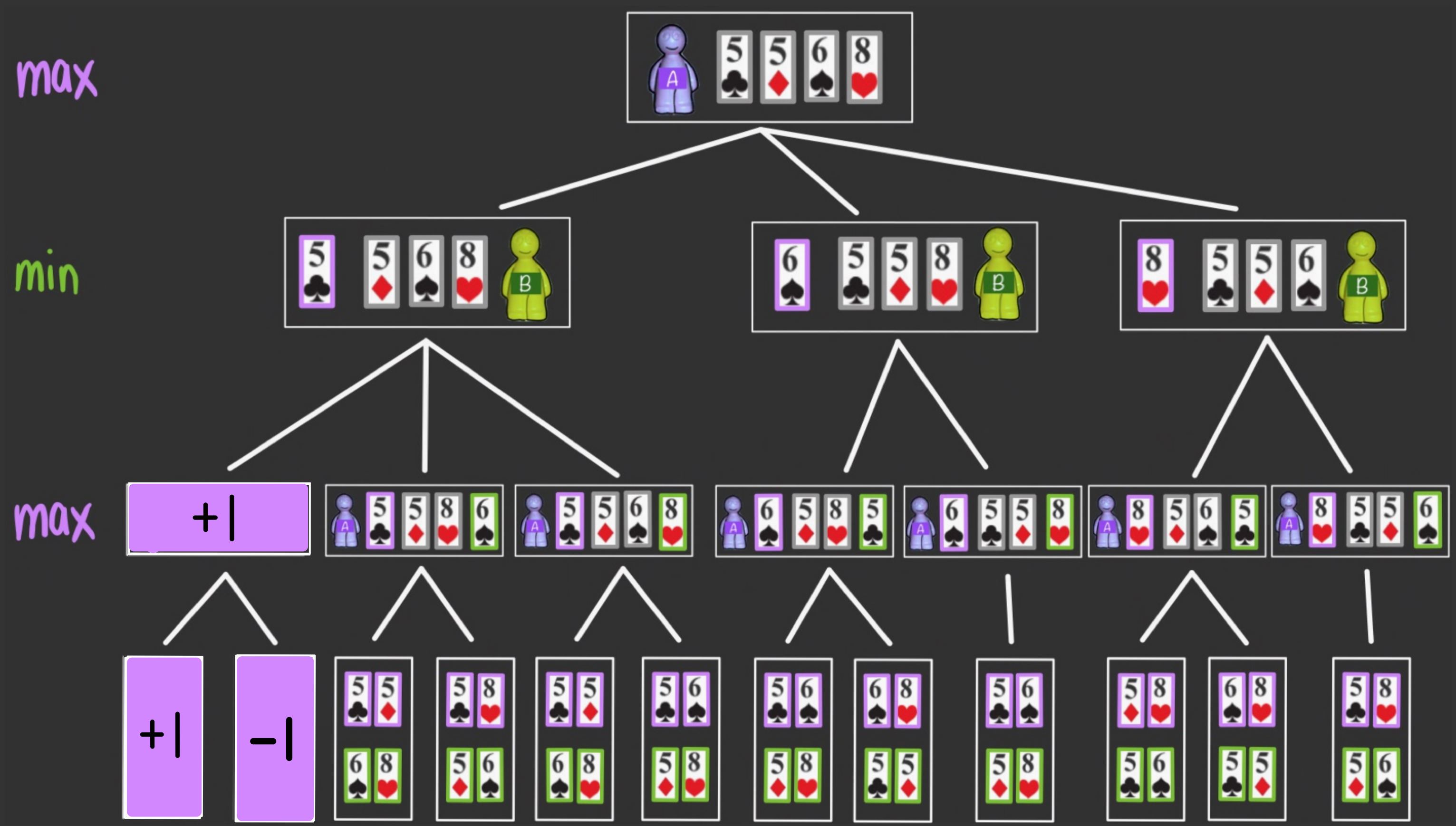


people are fundamentally selfless

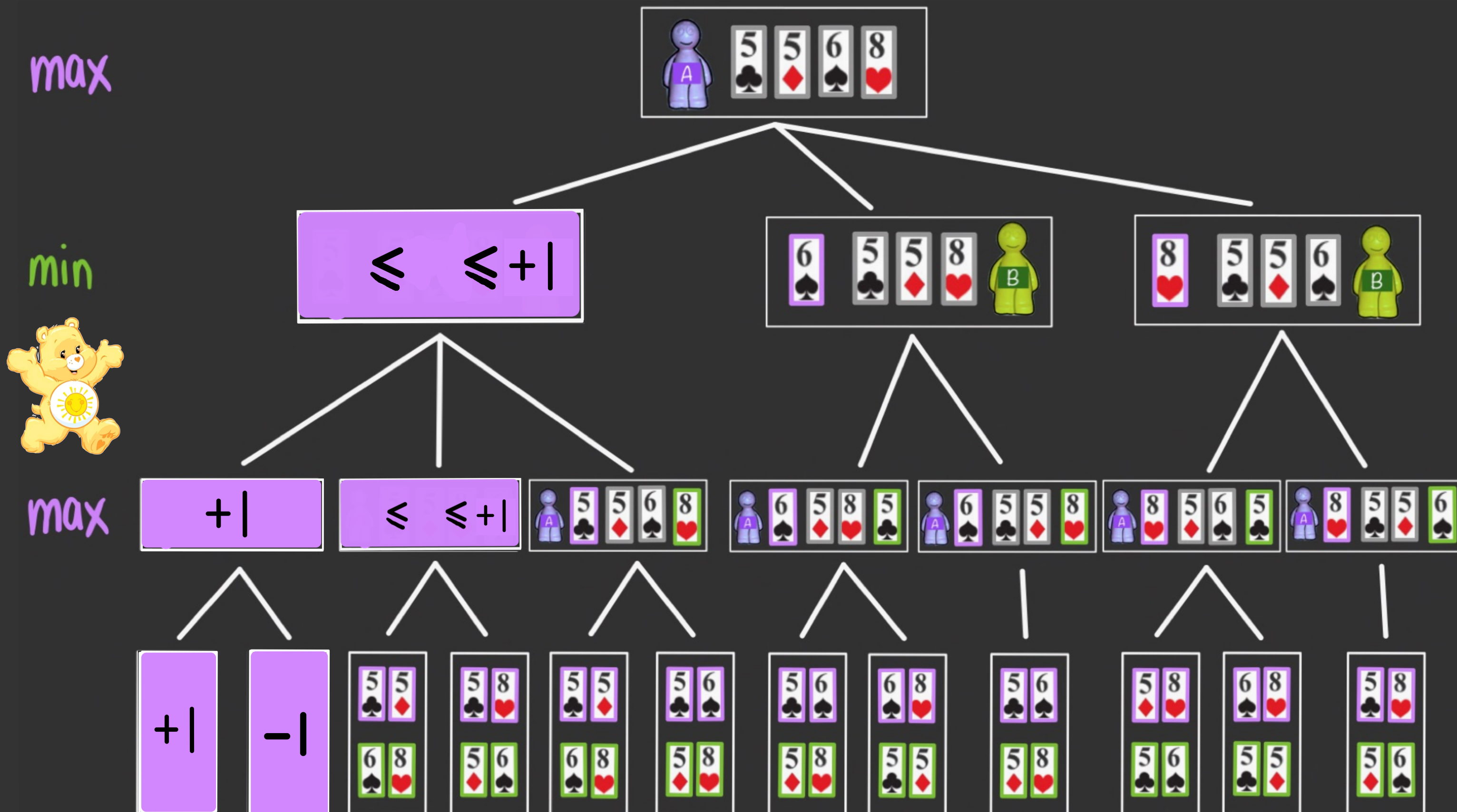
did we need to see these?

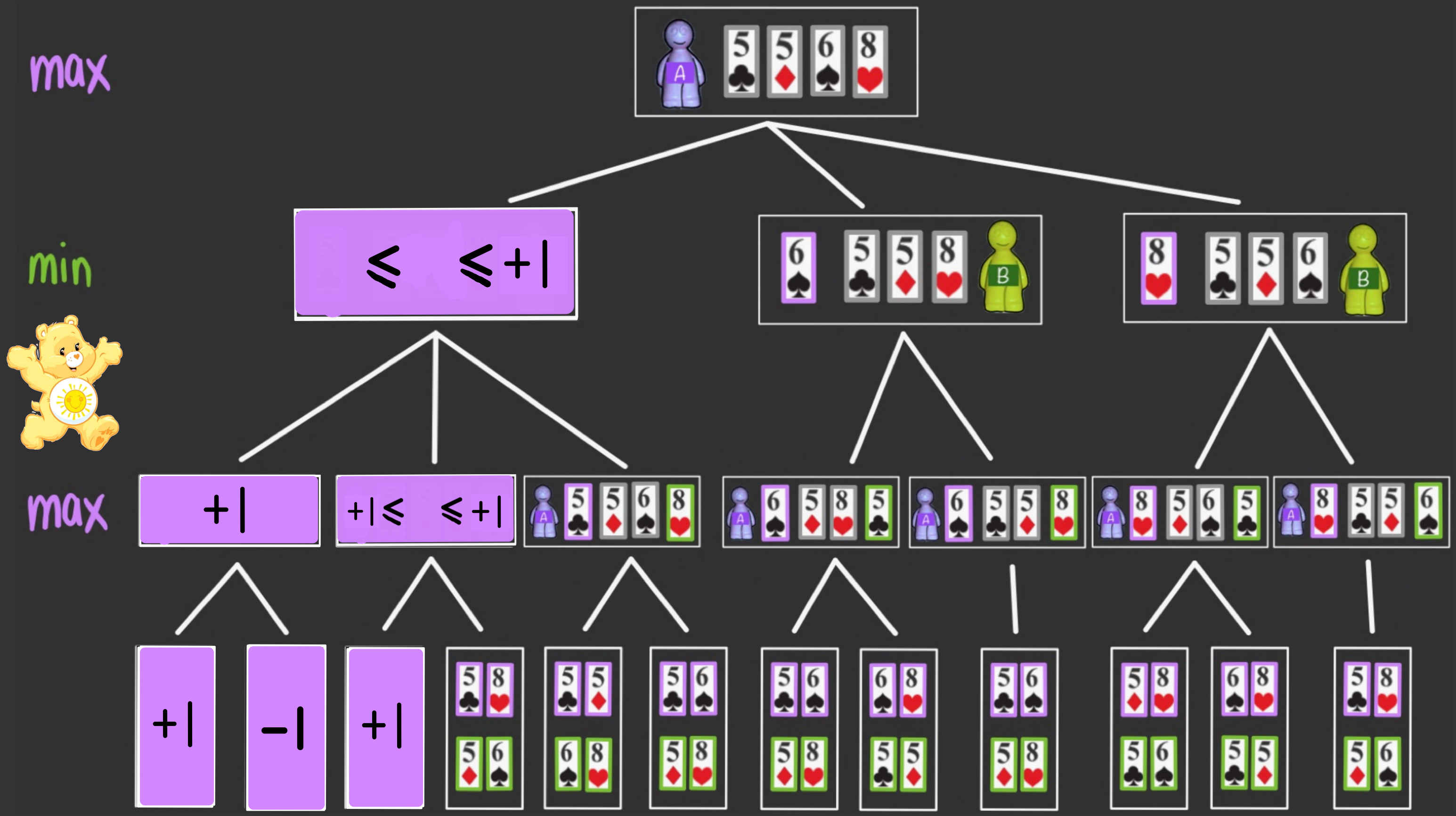

min
max
people are fundamentally selfless
≥
?
?
no

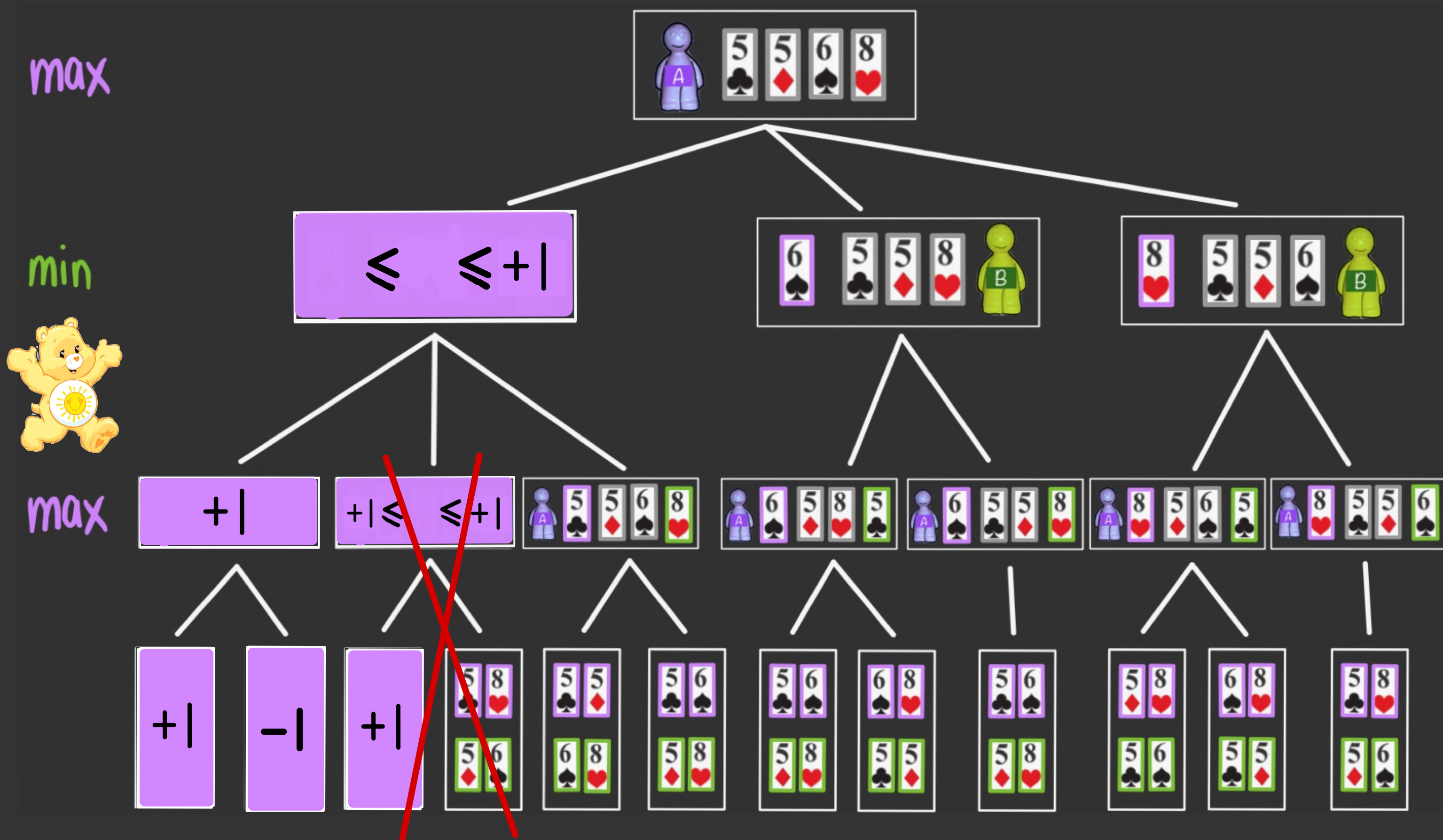

max
min
max
A
B

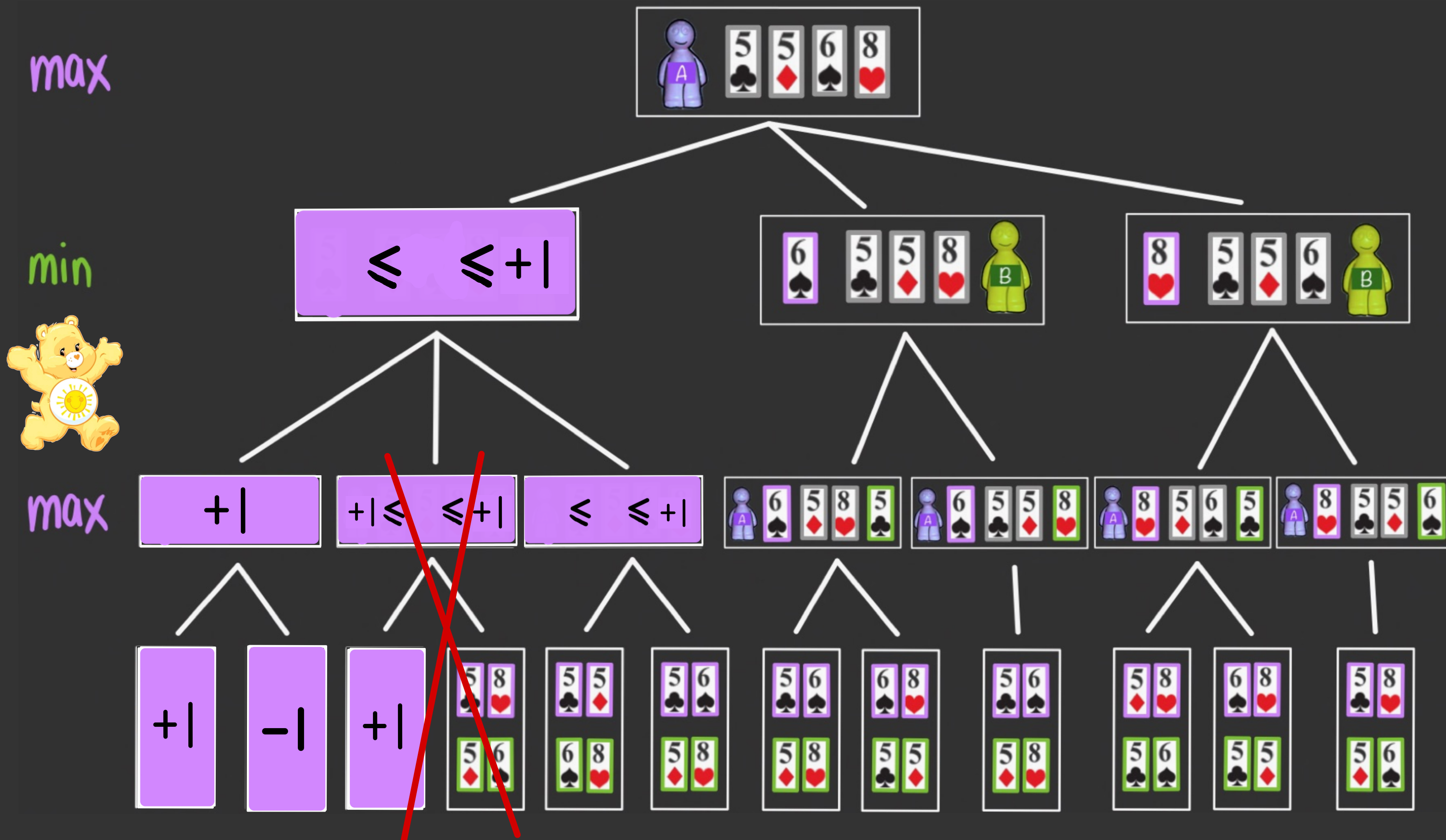


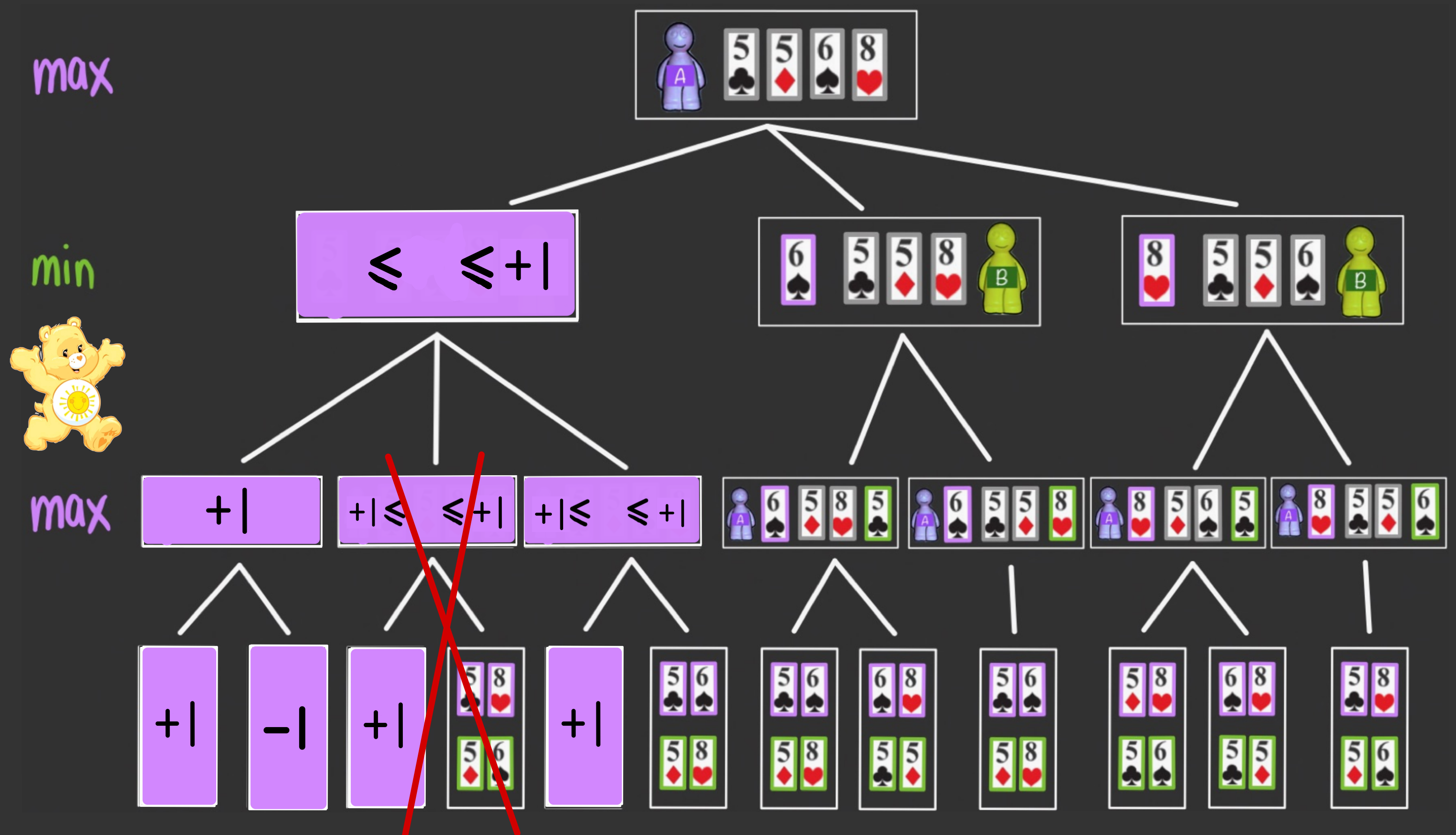
max
min
max
+1
+1
−1

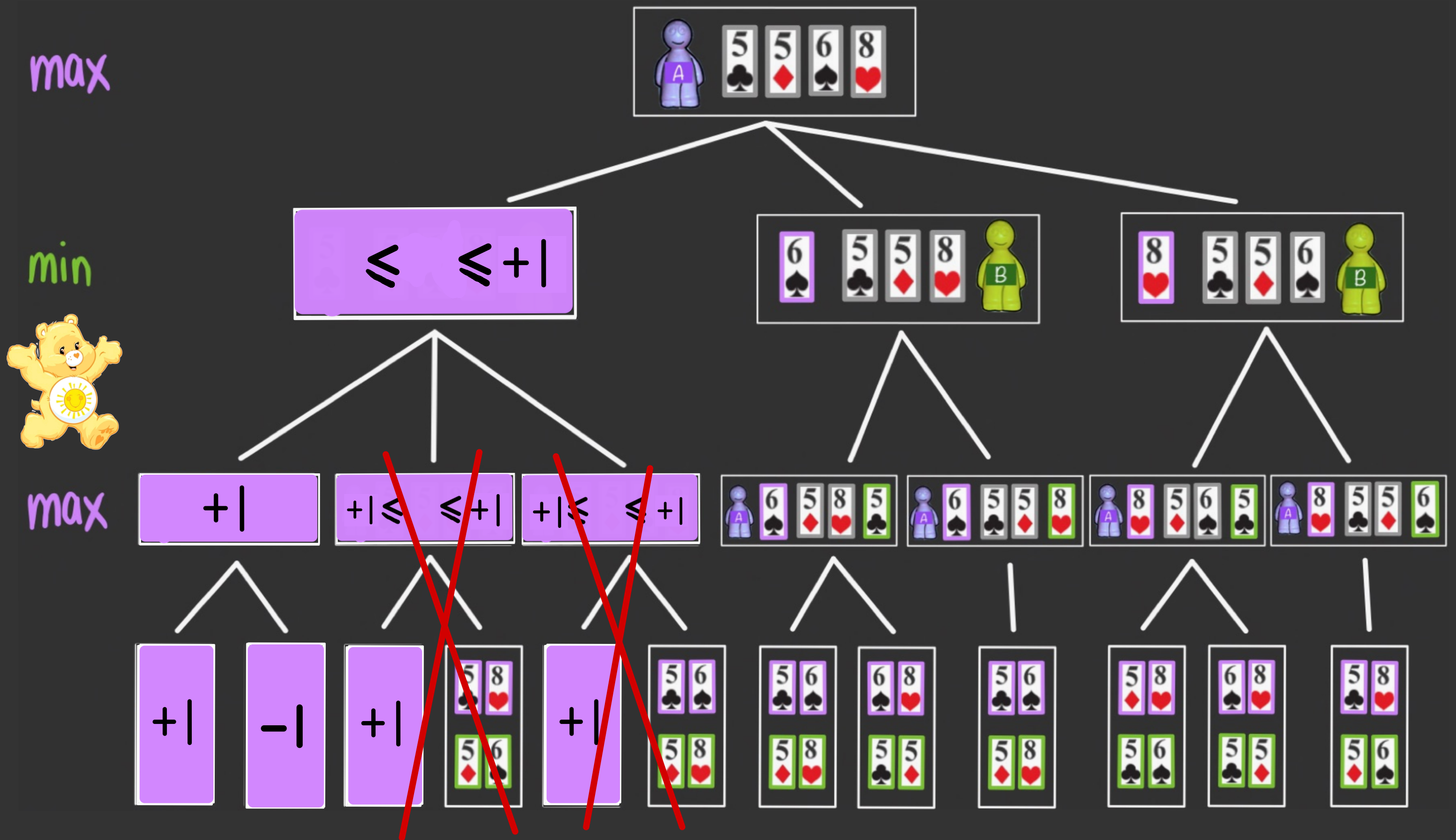

max
min
max
⩽ ⩽+1
+1
⩽ ⩽+1
+1
−1

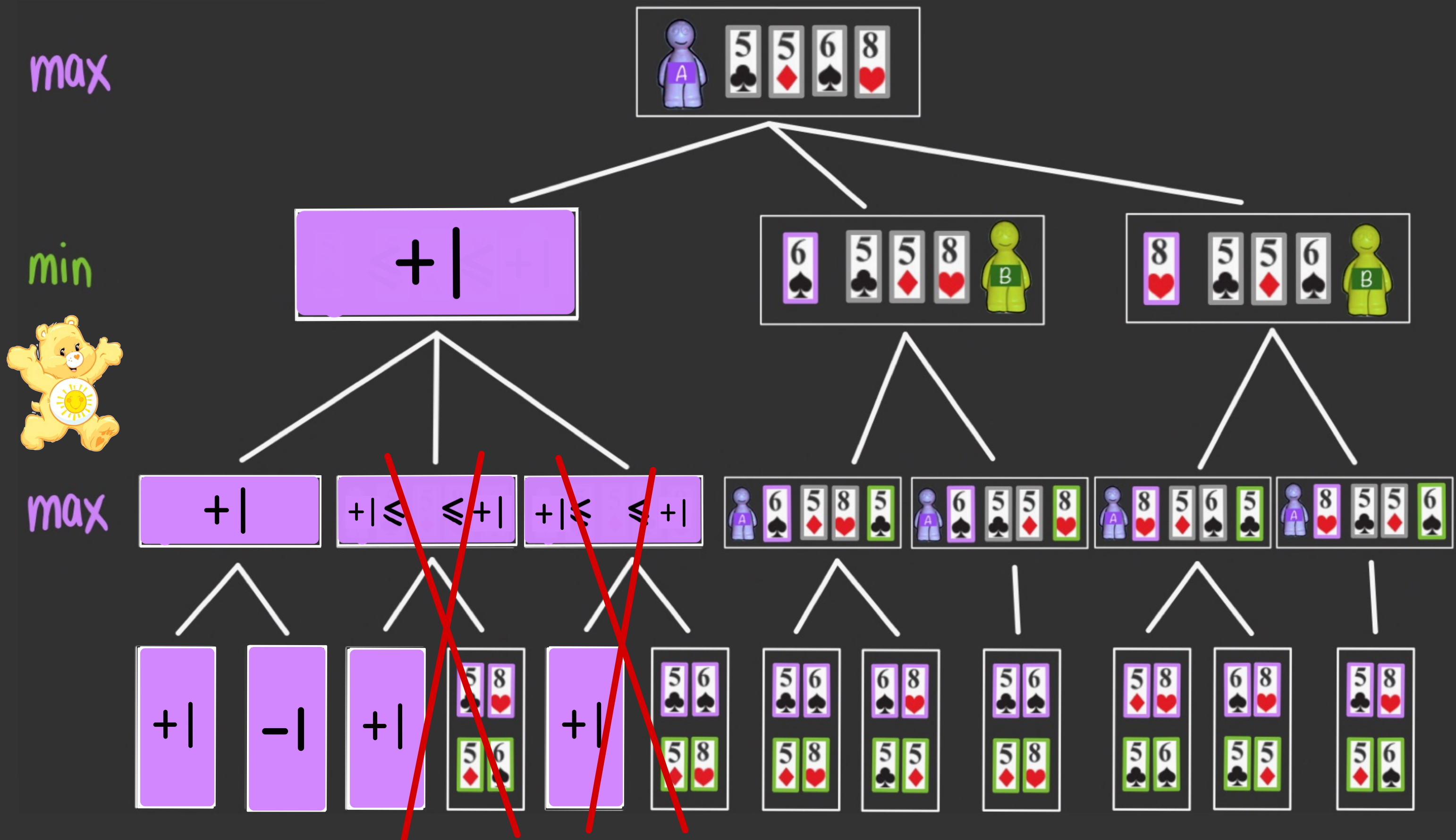

max
min
max


max
min
max

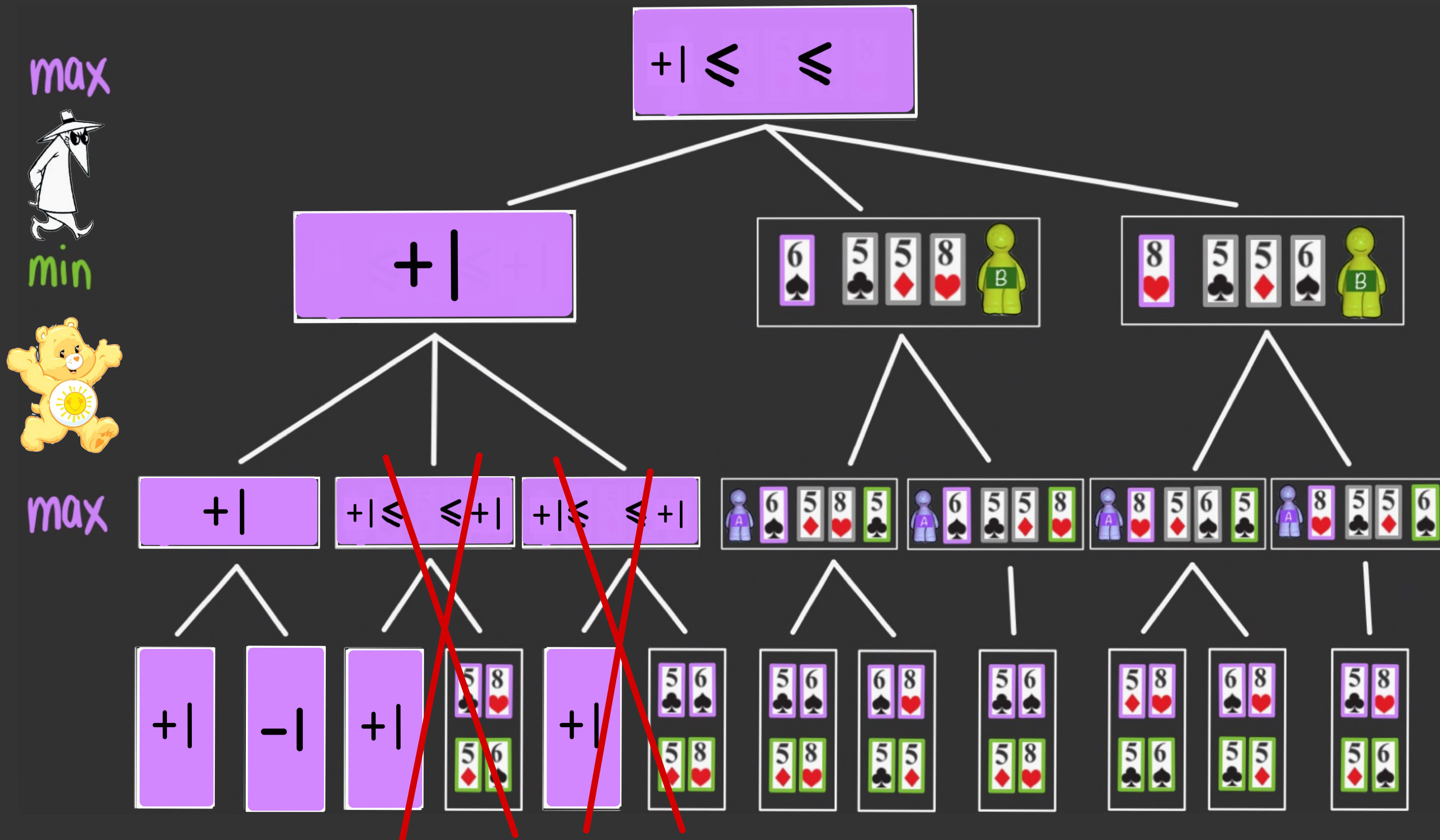

max
min
max
A
B
⩽ ⩽+1
+1
+1 ⩽ ⩽+1
⩽ ⩽+1
+1
−1
+1

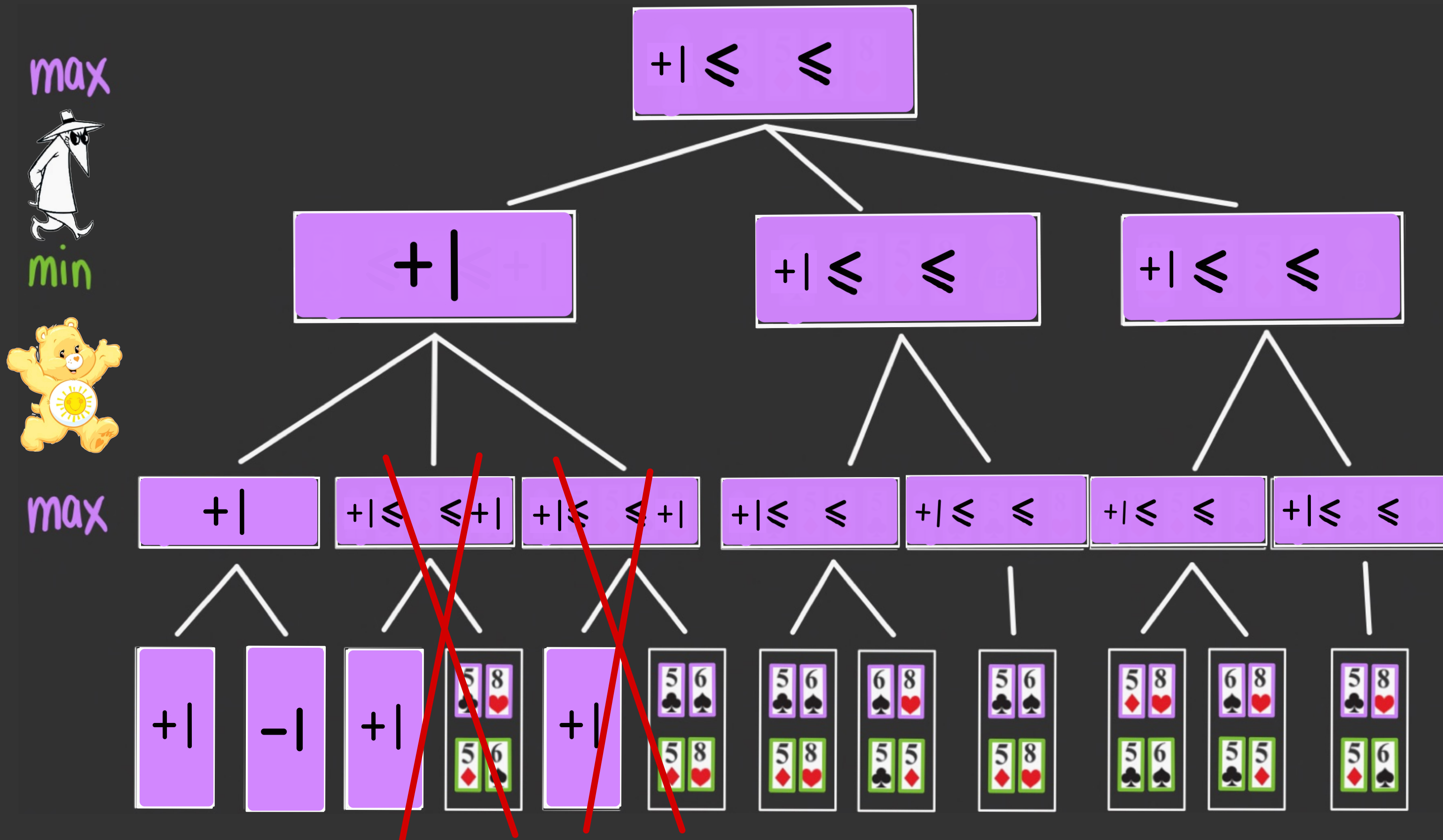

max
min
max
A
B
⩽
⩽+|
+|
+|⩽
⩽+|
-|


max
min
max
≤ ≤+1
+1
+1 ≤ ≤+1
+1 ≤ ≤+1
+1
−1
+1
+1

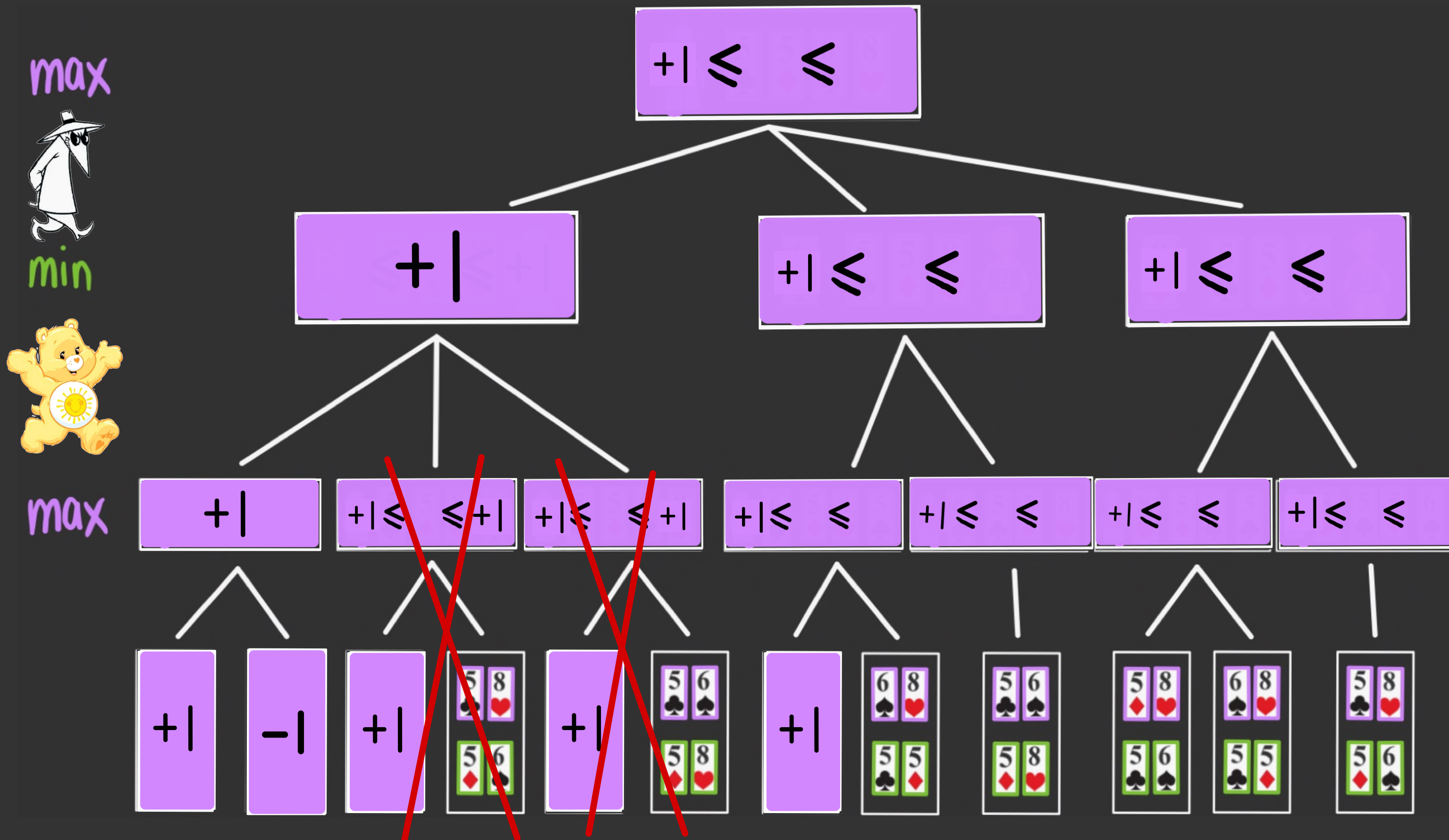


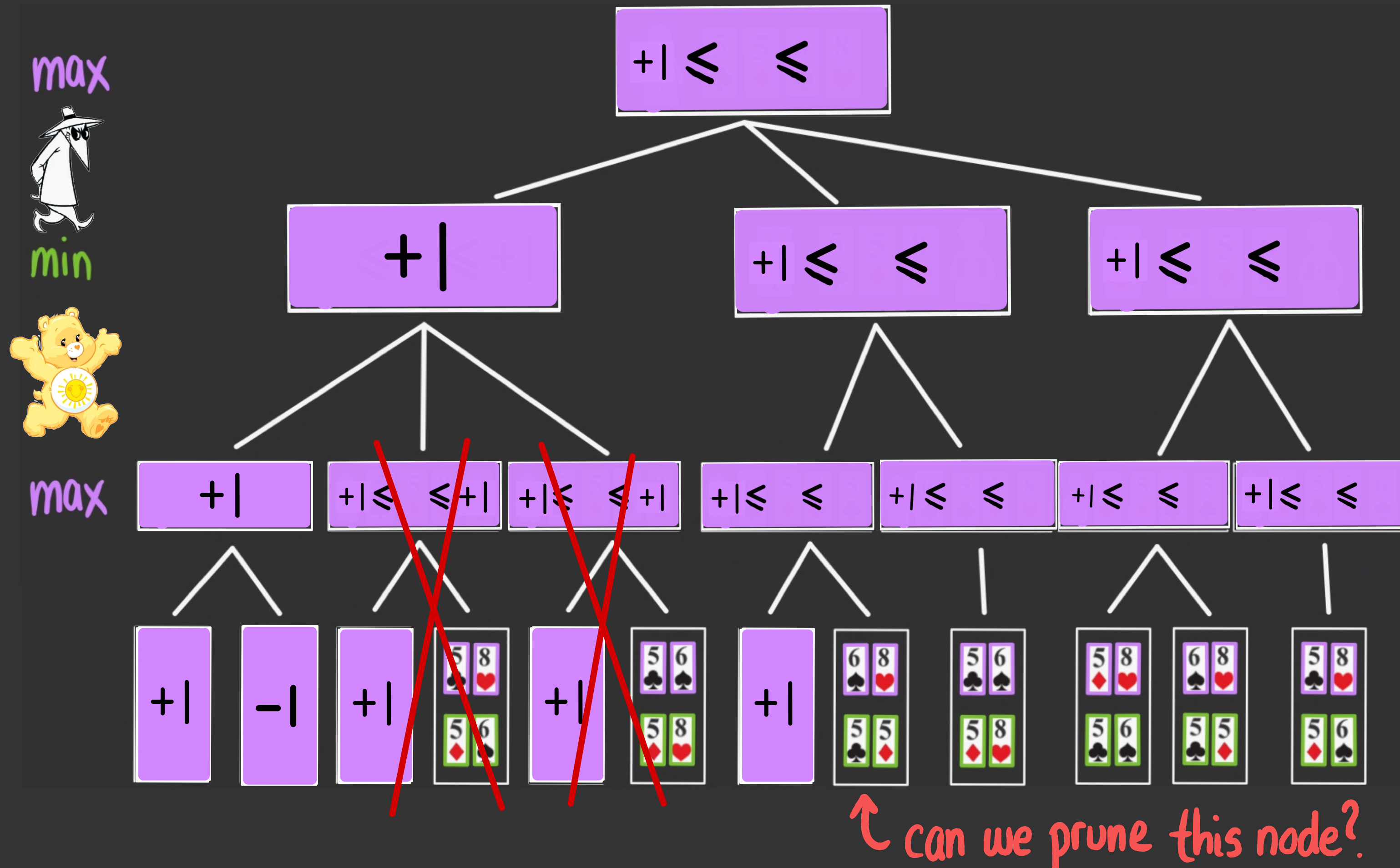
max
min
max


max
min
max

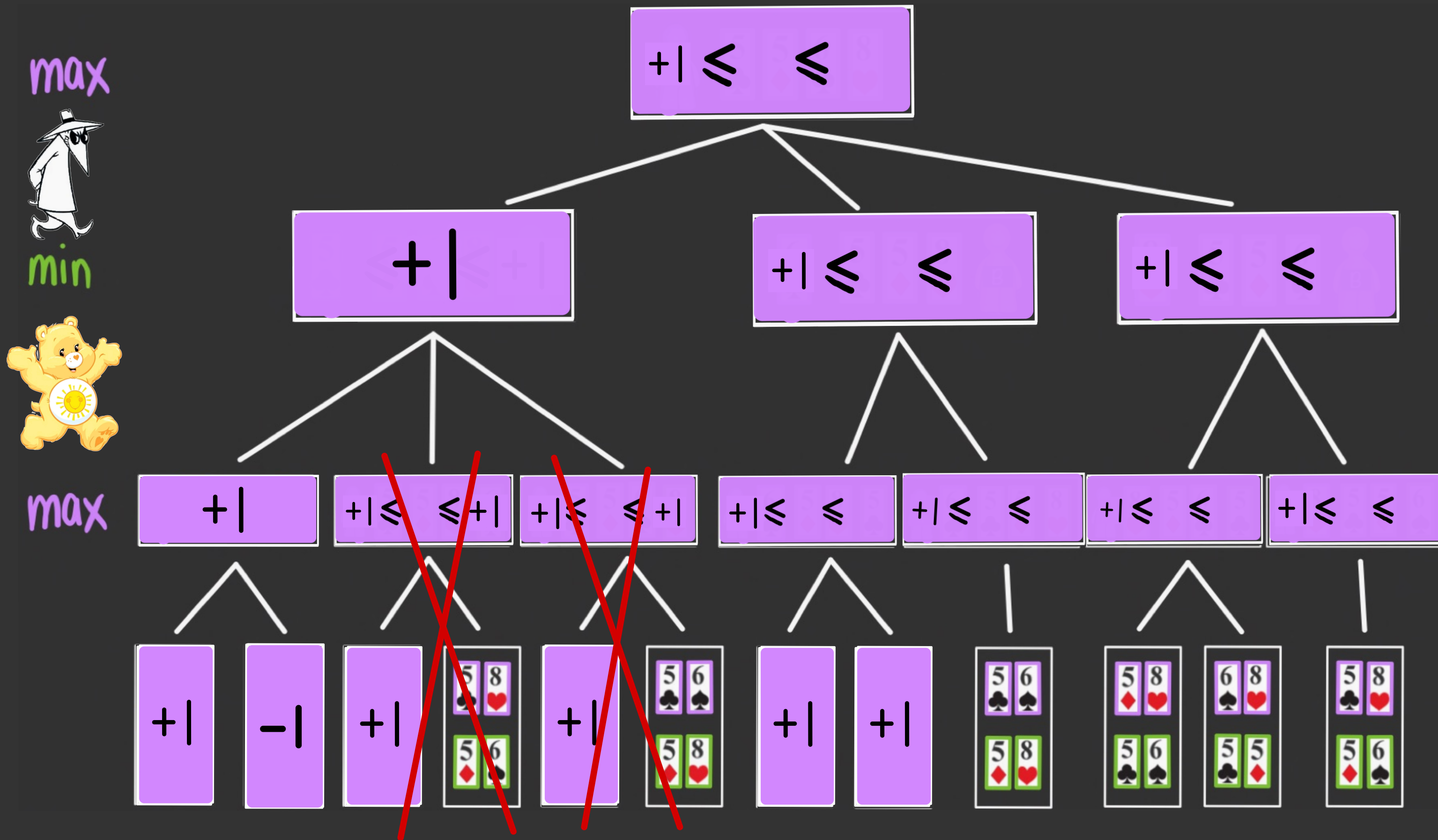


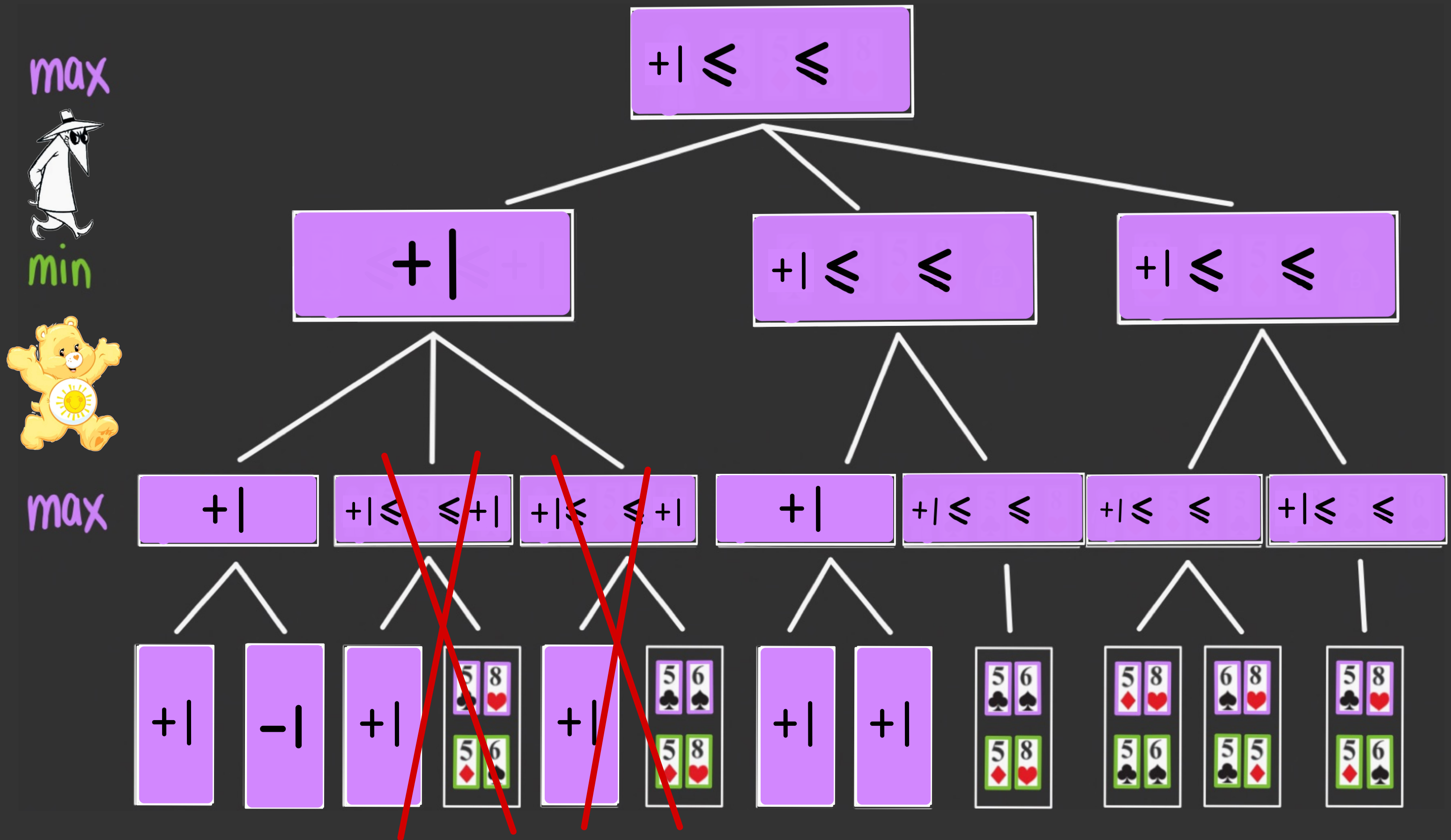
max
min
max

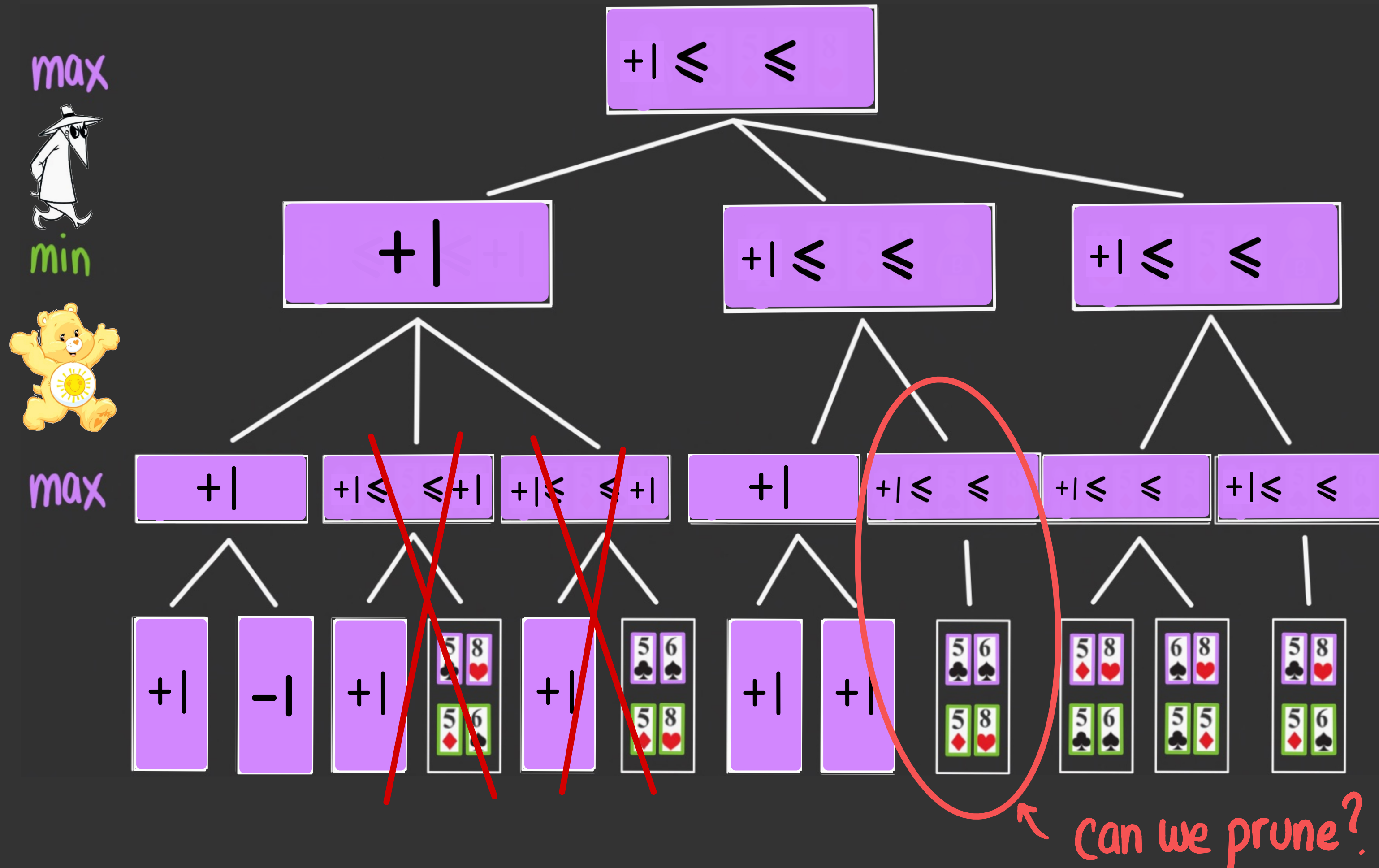

max
min
max



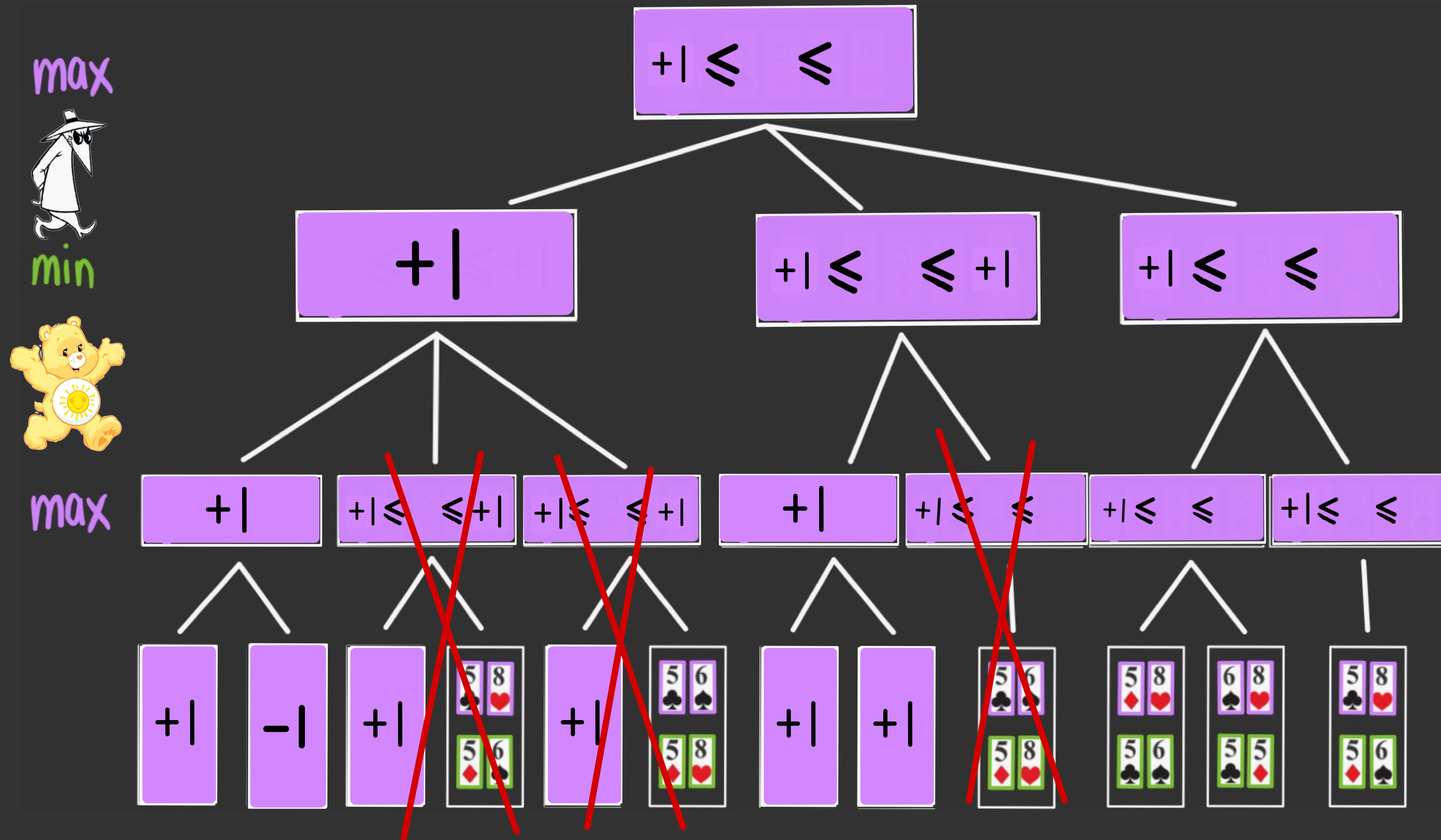
max
min
max
can we prune this node?

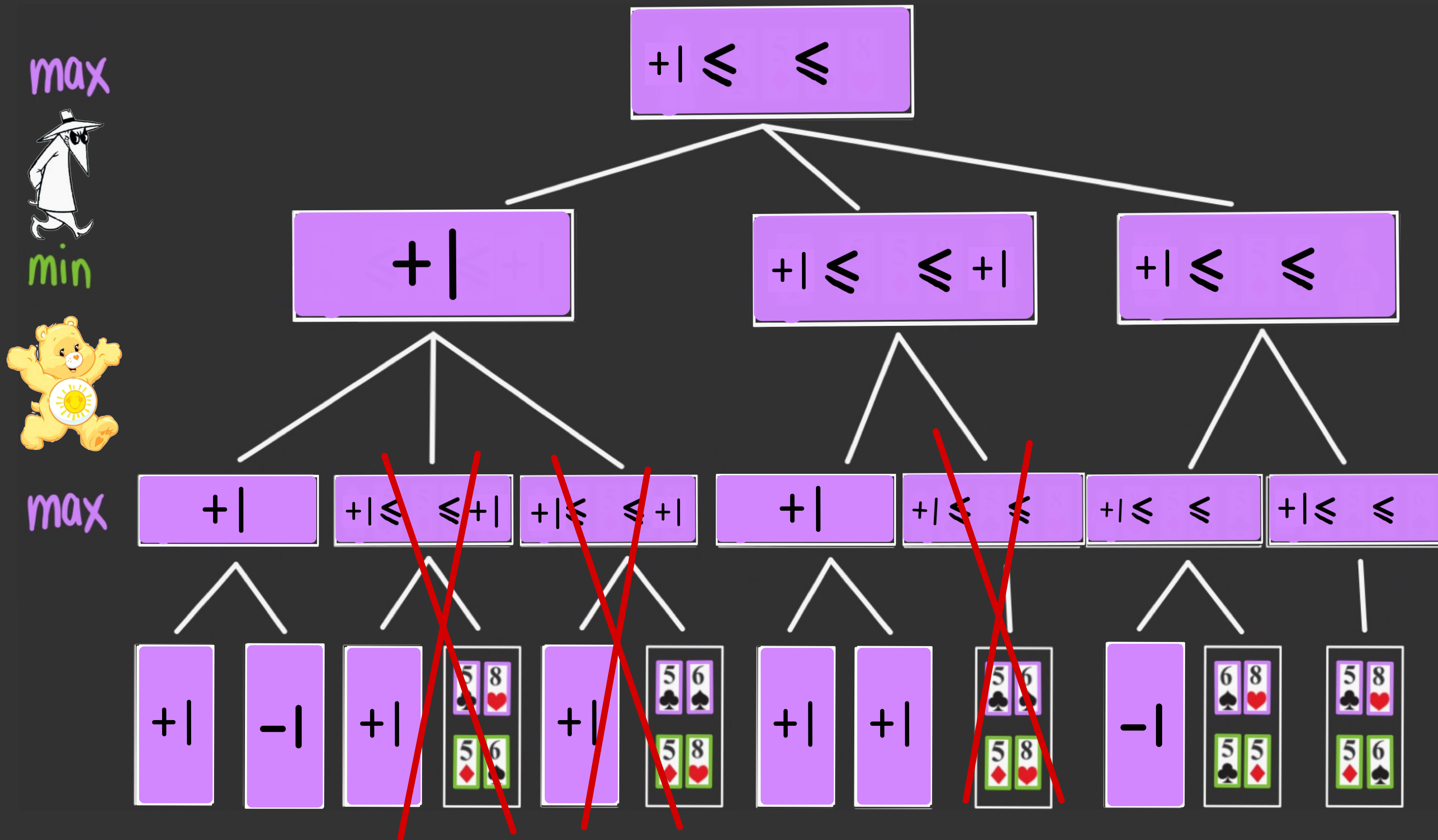

max
min
max

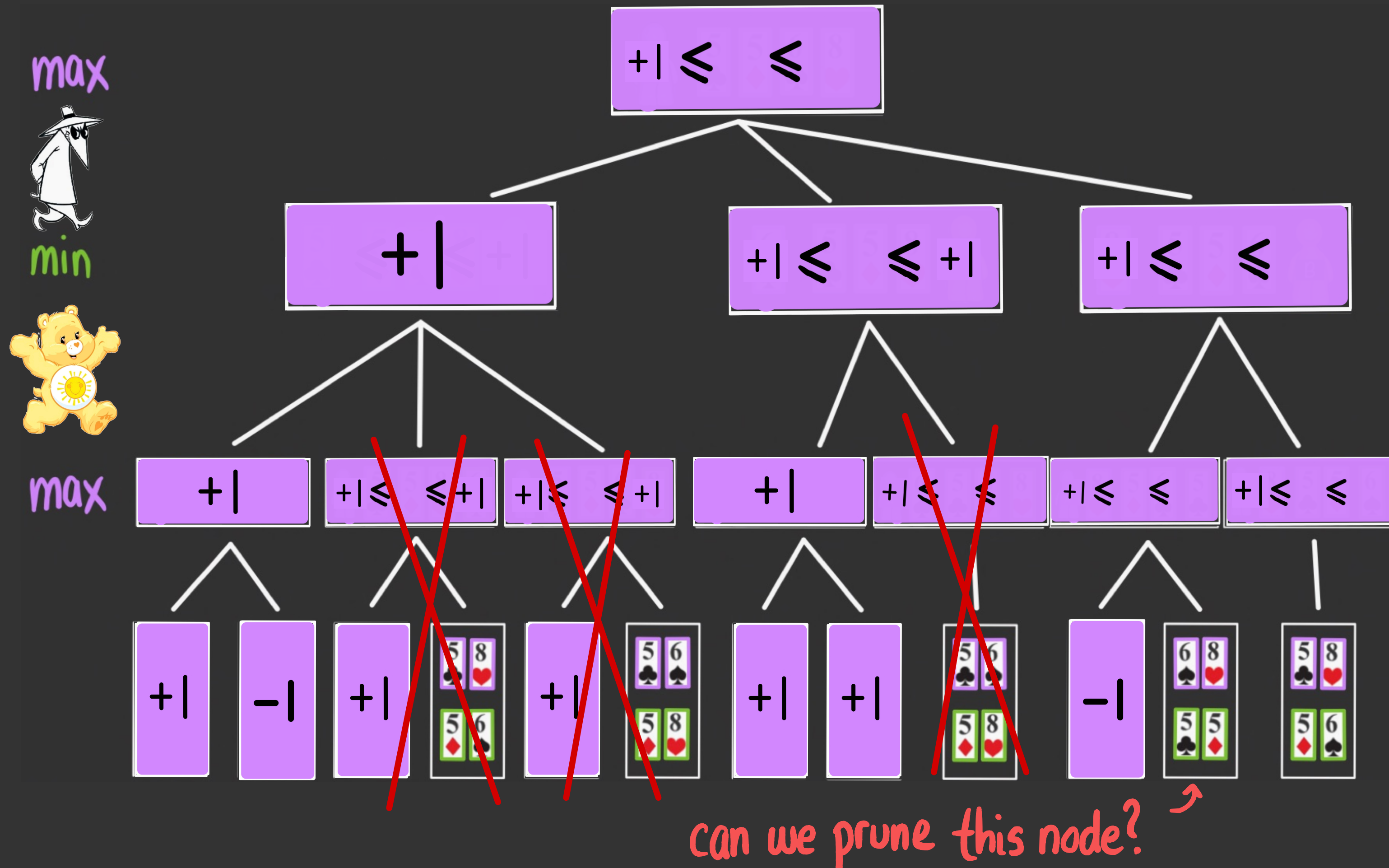

max
min
max

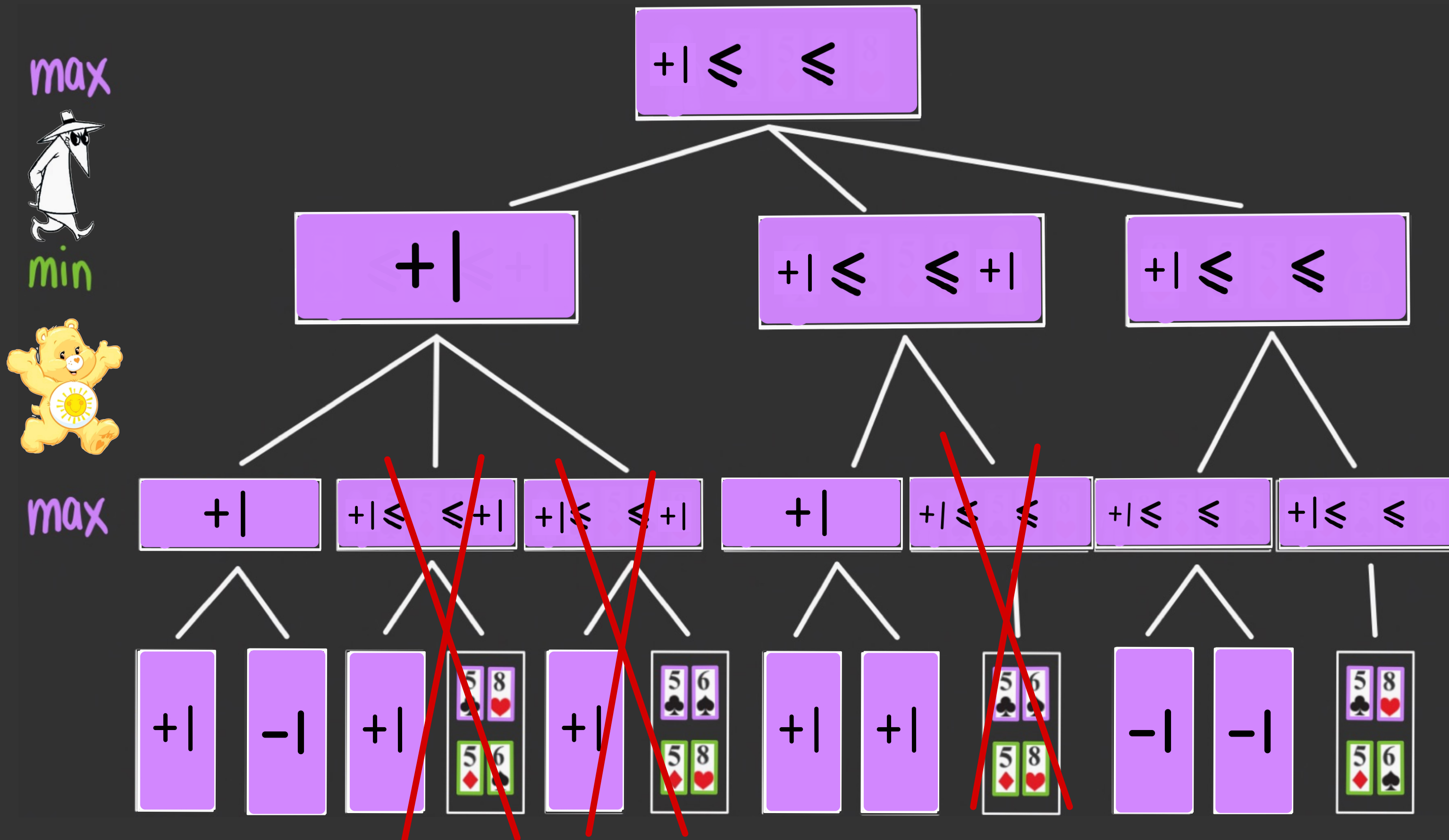


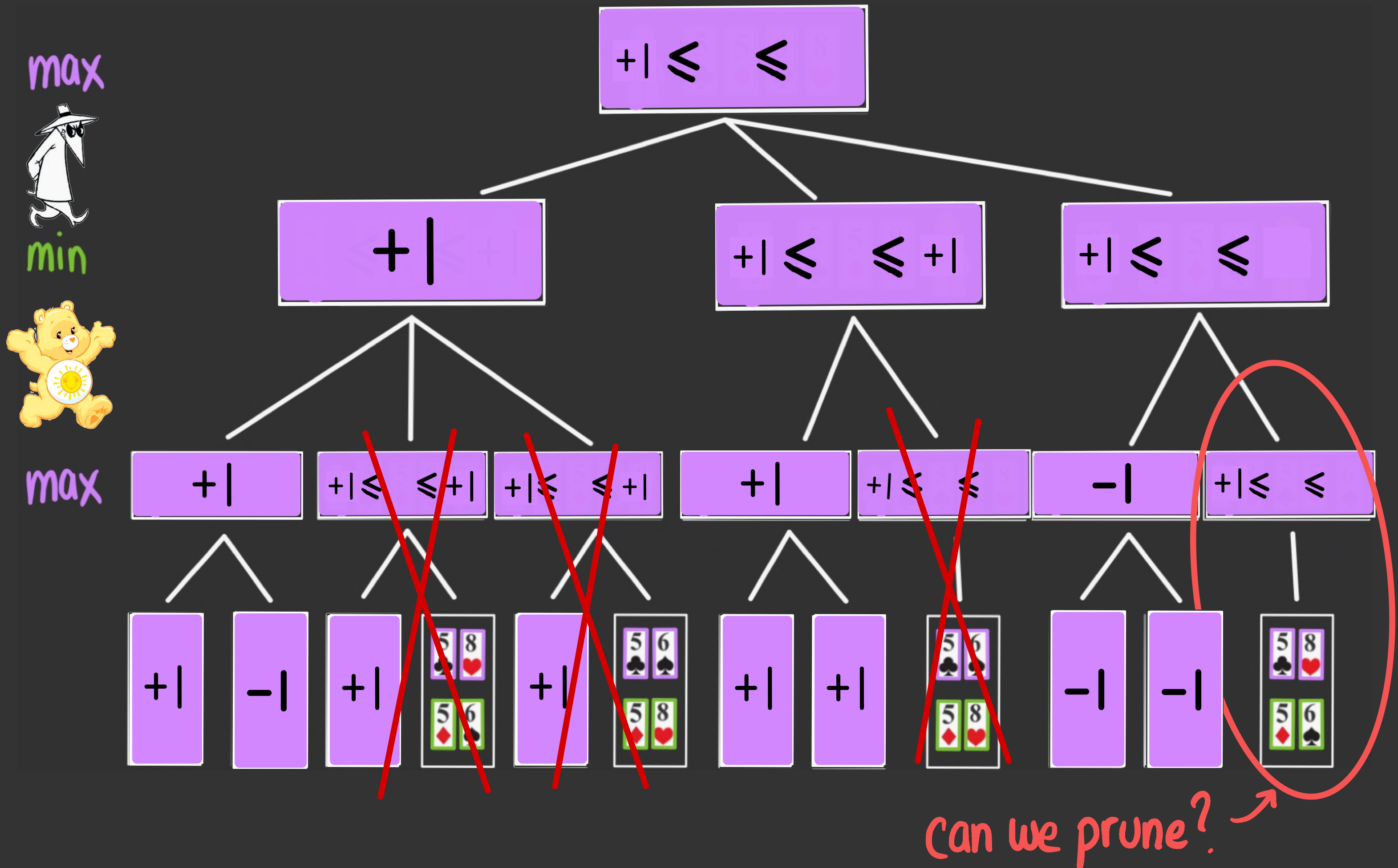
max
min
max
can we prune?


max
min
max

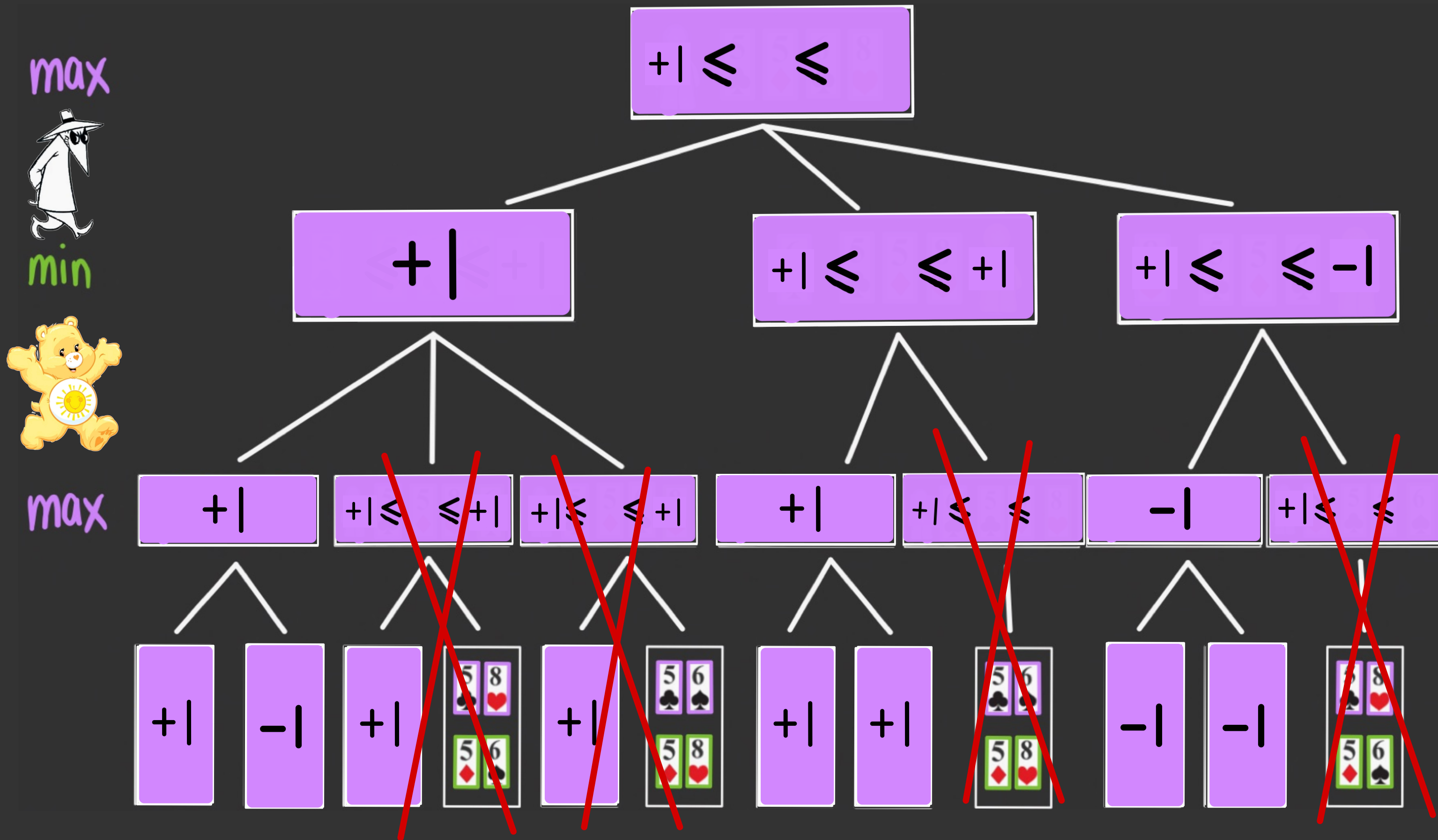

max
min
max


max
min
max
+1 ≤ ≤
+1
+1 ≤ ≤ +1
+1 ≤ ≤
-1
can we prune this node?

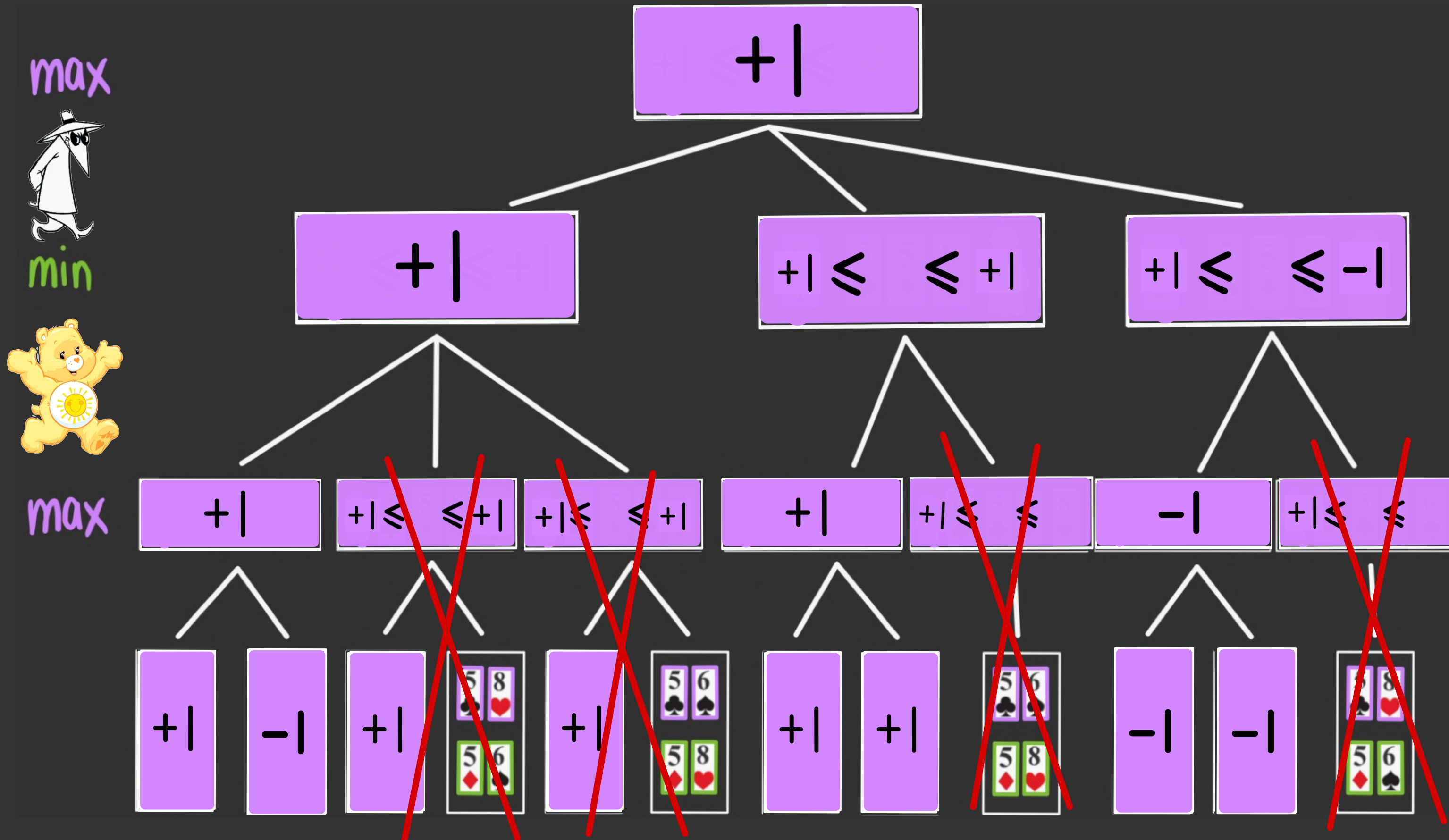

max
min
max
+|
-|



max
min
max
can we prune?



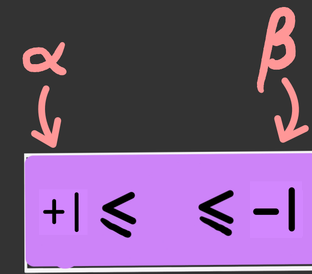
max
min
max



max
min
max

these bounds we maintain are called α and β, and therefore this optimization to minimax is called

alpha-beta pruning



MINIMAX$(q, p, (\mathcal{M}, U))$: # $\mathcal{M} = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue = $\boxed{?}$ if $p(q) = p$ else $\boxed{?}$
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - childvalue = MINIMAX$(q', p, (\mathcal{M}, U))$
  - if $p(q) = p$:
    - bestvalue = $\boxed{?}$(bestvalue, childvalue)
  - else
    - bestvalue = $\boxed{?}$(bestvalue, childvalue)
  - return bestvalue

```
MINIMAX(q, p, (M, U)):   # M = (Q, Σ, Δ, q₀, F)
  ▸ if q ∈ F: return U(q, p)
  ▸ bestvalue = -∞ if p(q) = p else ∞
  ▸ for ⟨q, σ, q′⟩ ∈ Δ:
      ▸ childvalue = MINIMAX(q′, p, (M, U))
      ▸ if p(q) = p:
          ▸ bestvalue = max(bestvalue, childvalue)
      ▸ else
          ▸ bestvalue = min(bestvalue, childvalue)
      ▸ return bestvalue
```

```
MINIMAX(q, p, (M, U), α, β):  # M = (Q, Σ, Δ, q₀, F)
  ▸ if q ∈ F: return U(q, p)
  ▸ bestvalue = -∞ if p(q) = p else ∞
  ▸ for ⟨q, σ, q'⟩ ∈ Δ:
      ▸ ?
      ▸ childvalue = MINIMAX(q', p, (M, U), α, β)
      ▸ if p(q) = p:
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ ?
      ▸ else:
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ ?
  ▸ return bestvalue
```

$\textsc{Minimax}(q, p, (m, U), \alpha, \beta)$: # $m = (Q, \Sigma, \Delta, q_0, F)$

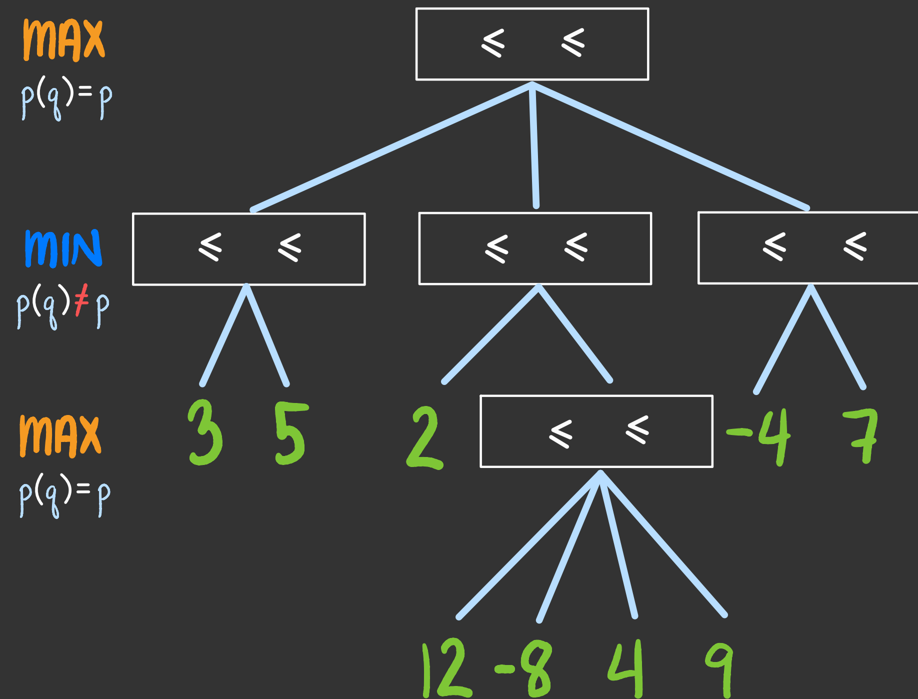
- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - ?
  - childvalue $= \textsc{Minimax}(q', p, (m, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue $= \max(\text{bestvalue}, \text{childvalue})$
    - $\alpha = \max(\alpha, \text{childvalue})$
  - else:
    - bestvalue $= \min(\text{bestvalue}, \text{childvalue})$
    - $\beta = \min(\beta, \text{childvalue})$
- return bestvalue

```
MINIMAX(q, p, (m, U), α, β):   # m = (Q, Σ, Δ, q₀, F)
  ▸ if q ∈ F: return U(q, p)
  ▸ bestvalue = -∞ if p(q) = p else ∞
  ▸ for ⟨q, σ, q′⟩ ∈ Δ:
      ▸ if α ≥ β: return bestvalue
      ▸ childvalue = MINIMAX(q′, p, (m, U), α, β)
      ▸ if p(q) = p:
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ α = max(α, childvalue)
      ▸ else:
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ β = min(β, childvalue)
      ▸ return bestvalue
```

try alpha-beta yourself

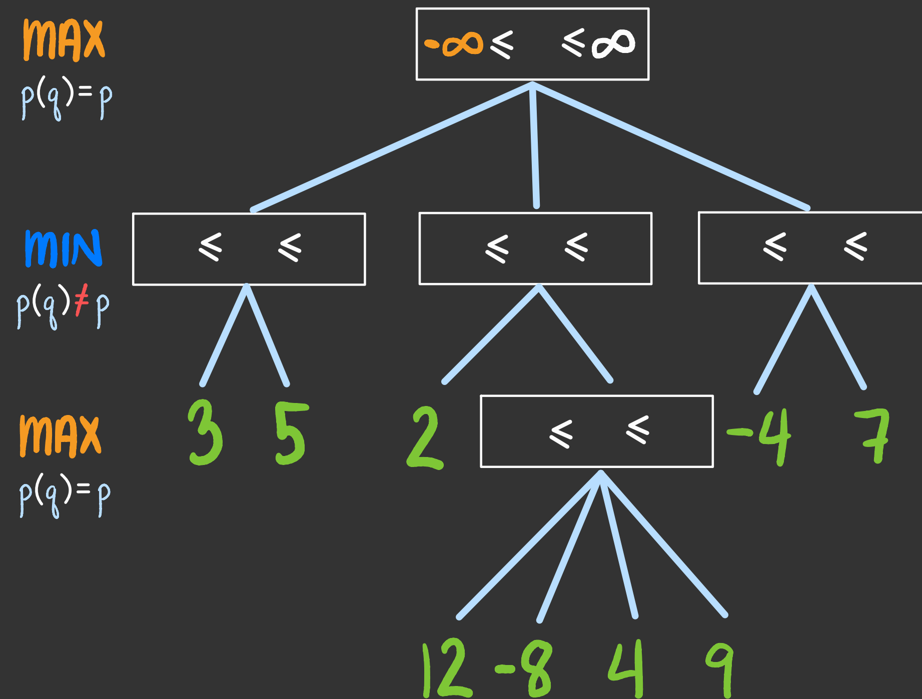
$\text{MINIMAX}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue $= \text{MINIMAX}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue $= \max(\text{bestvalue}, \text{childvalue})$
    - $\alpha = \max(\alpha, \text{childvalue})$
  - else:
    - bestvalue $= \min(\text{bestvalue}, \text{childvalue})$
    - $\beta = \min(\beta, \text{childvalue})$
- return bestvalue



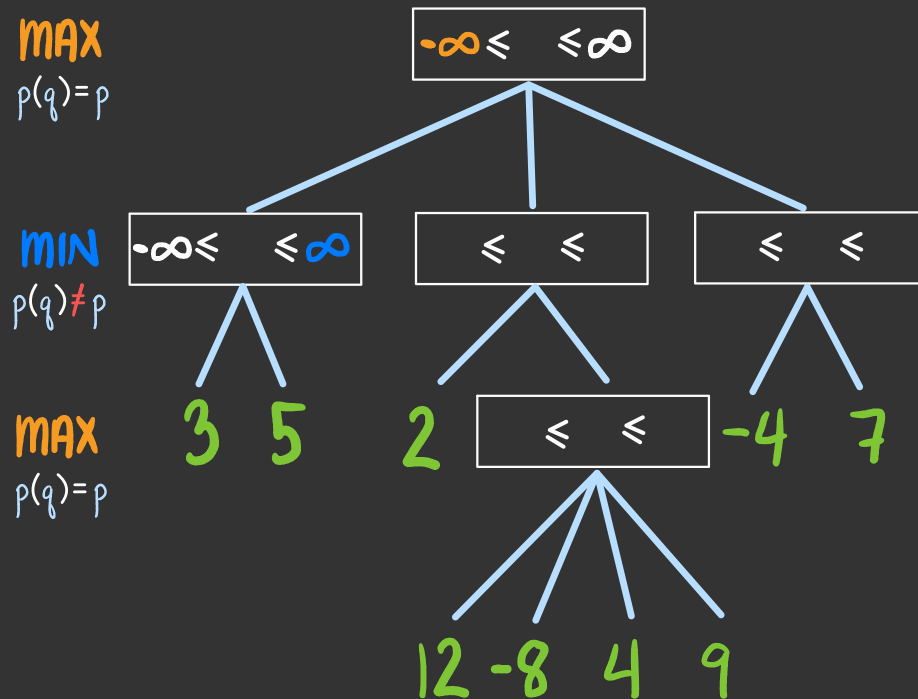
$\text{MINIMAX}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue $= \text{MINIMAX}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue $= \max$(bestvalue, childvalue)
    - $\alpha = \max(\alpha$, childvalue$)$
  - else:
    - bestvalue $= \min$(bestvalue, childvalue)
    - $\beta = \min(\beta$, childvalue$)$
  - return bestvalue



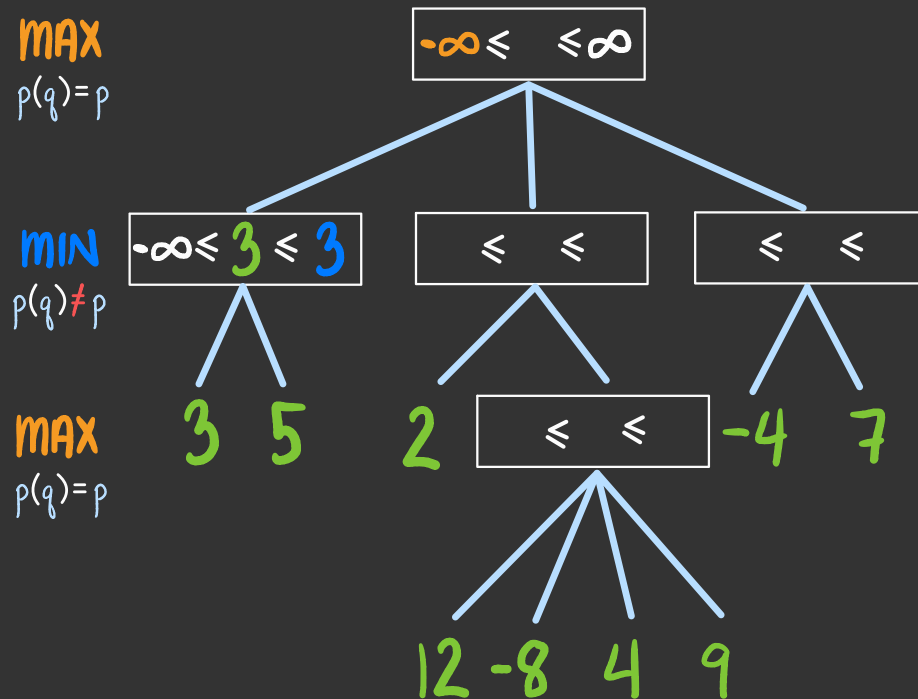
$\text{Minimax}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue $=$ $\text{Minimax}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue $=$ max(bestvalue, childvalue)
    - $\alpha = \max(\alpha, \text{childvalue})$
  - else:
    - bestvalue $=$ min(bestvalue, childvalue)
    - $\beta = \min(\beta, \text{childvalue})$
- return bestvalue

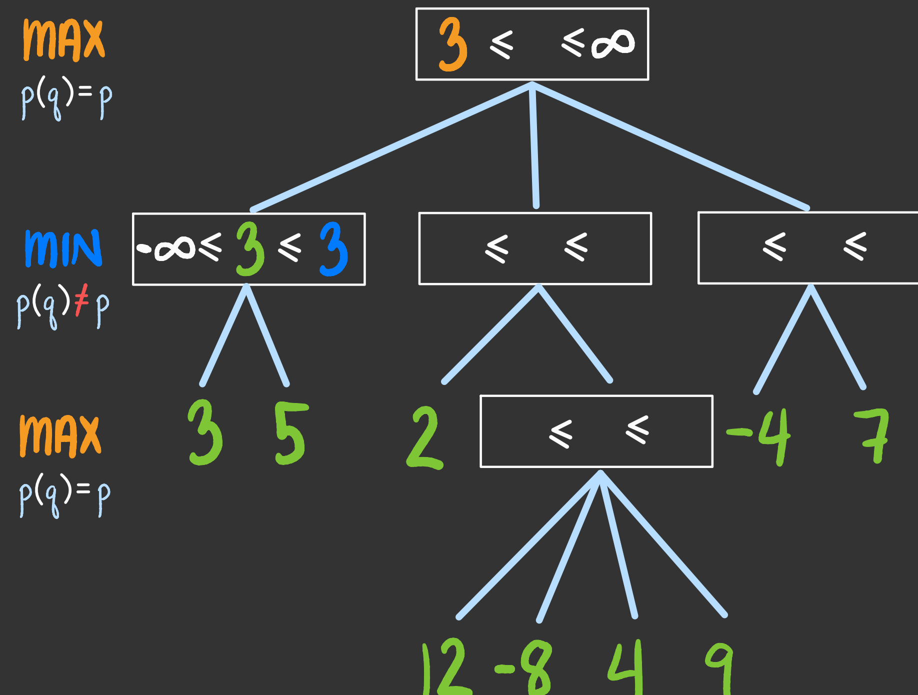


$\text{Minimax}(q, p, (M,U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

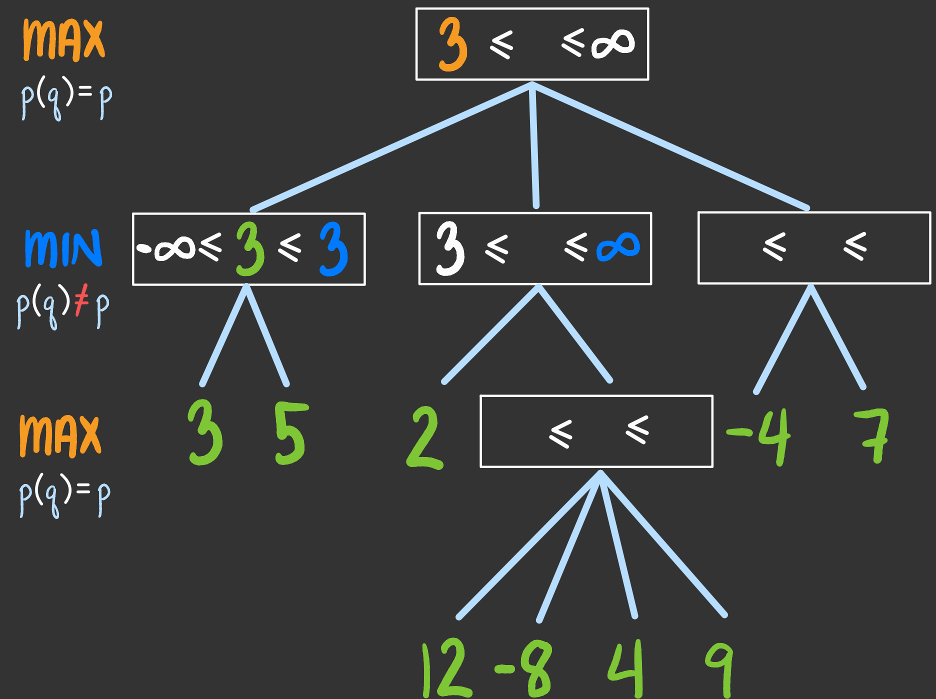
- if $q \in F$: return $U(q,p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue = $\text{Minimax}(q', p, (M,U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue = max(bestvalue, childvalue)
    - $\alpha = \max(\alpha, \text{childvalue})$
  - else:
    - bestvalue = min(bestvalue, childvalue)
    - $\beta = \min(\beta, \text{childvalue})$
- return bestvalue



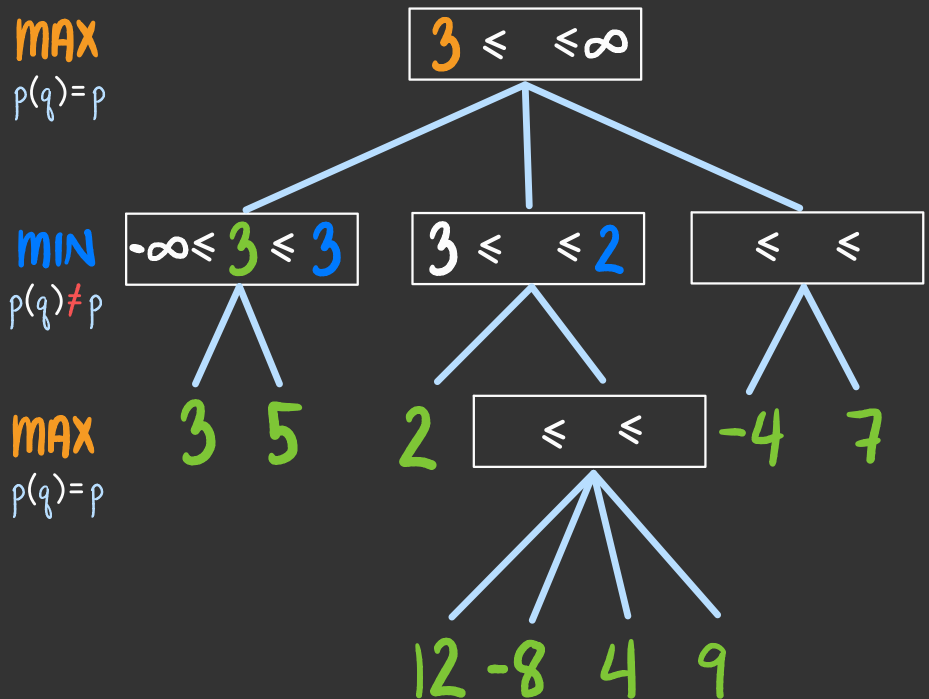
```
Minimax(q, p, (M,U), α, β):  # M = (Q,Σ,Δ,q₀,F)
▸ if q ∈ F: return U(q,p)
▸ bestvalue = -∞ if p(q)=p else ∞
▸ for ⟨q,σ,q'⟩ ∈ Δ:
    ▸ if α ≥ β: return bestvalue
    ▸ childvalue = Minimax(q', p, (M,U), α, β)
    ▸ if p(q)=p:
        ▸ bestvalue = max(bestvalue, childvalue)
        ▸ α = max(α, childvalue)
    ▸ else:
        ▸ bestvalue = min(bestvalue, childvalue)
        ▸ β = min(β, childvalue)
▸ return bestvalue
```



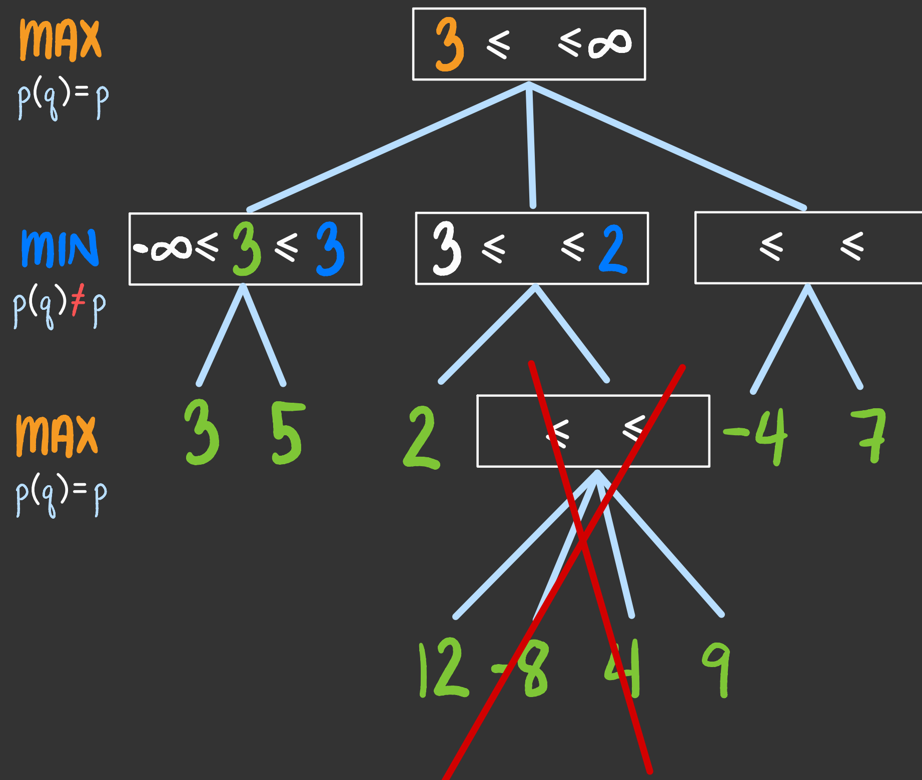
```
Minimax(q, p, (M,U), α, β):  # M = (Q,Σ,Δ,q0,F)
  ▸ if q ∈ F: return U(q,p)
  ▸ bestvalue = -∞ if p(q)=p else ∞
  ▸ for ⟨q,σ,q'⟩ ∈ Δ:
      ▸ if α ≥ β: return bestvalue
      ▸ childvalue = Minimax(q', p, (M,U), α, β)
      ▸ if p(q)=p:
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ α = max(α, childvalue)
      ▸ else:
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ β = min(β, childvalue)
  ▸ return bestvalue
```



```
Minimax(q, p, (M,U), α, β): # M = (Q,Σ,Δ,q₀,F)
  ▸ if q ∈ F: return U(q,p)
  ▸ bestvalue = -∞ if p(q)=p else ∞
  ▸ for <q,σ,q'> ∈ Δ:
      ▸ if α ≥ β: return bestvalue
      ▸ childvalue = Minimax(q', p, (M,U), α, β)
      ▸ if p(q)=p:
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ α = max(α, childvalue)
      ▸ else:
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ β = min(β, childvalue)
  ▸ return bestvalue
```

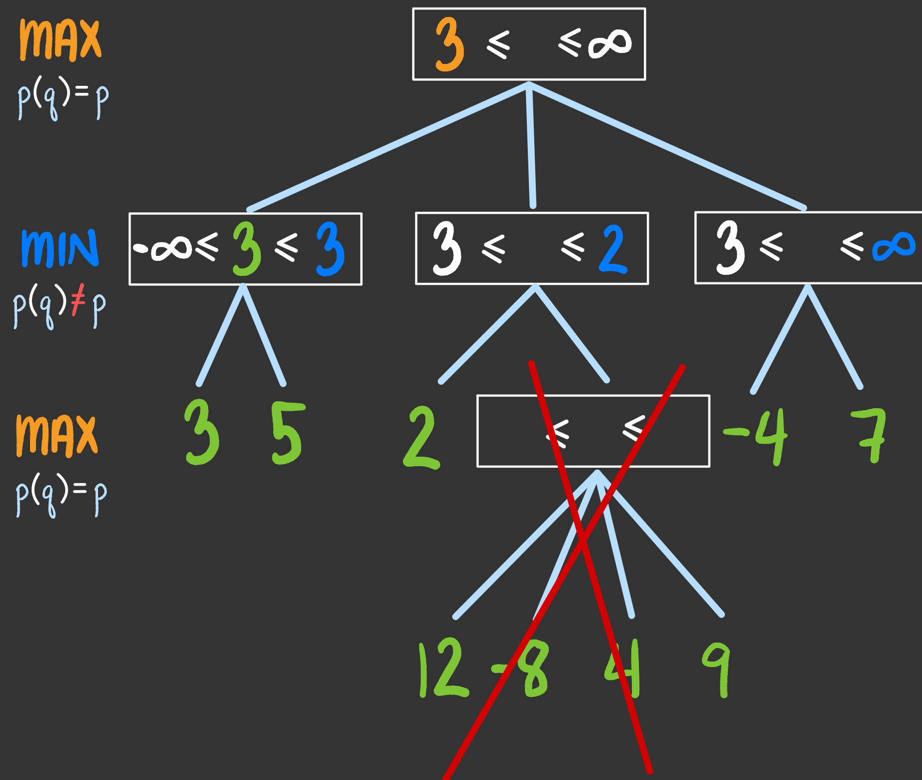


```
MINIMAX(q, p, (M,U), α, β):  # M = (Q, Σ, Δ, q₀, F)
  ▸ if q ∈ F: return U(q,p)
  ▸ bestvalue = -∞ if p(q)=p else ∞
  ▸ for ⟨q, σ, q'⟩ ∈ Δ:
      ▸ if α ≥ β: return bestvalue
      ▸ childvalue = MINIMAX(q', p, (M,U), α, β)
      ▸ if p(q)=p :
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ α = max(α, childvalue)
      ▸ else :
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ β = min(β, childvalue)
  ▸ return bestvalue
```



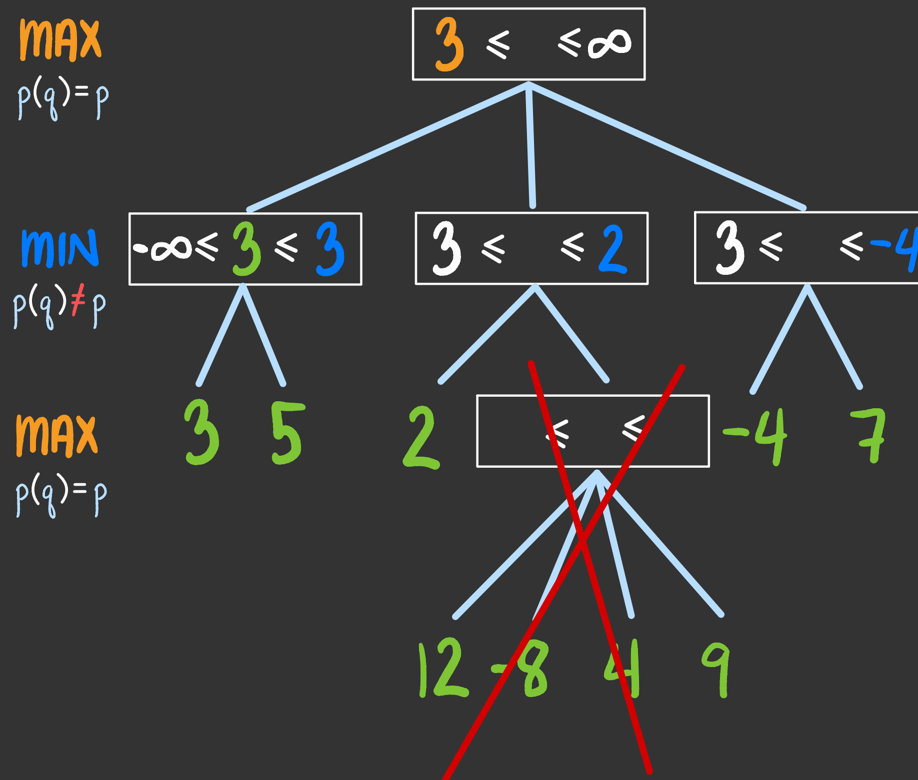
$\mathrm{MINIMAX}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue = $\mathrm{MINIMAX}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue = max(bestvalue, childvalue)
    - α = max(α, childvalue)
  - else:
    - bestvalue = min(bestvalue, childvalue)
    - β = min(β, childvalue)
- return bestvalue



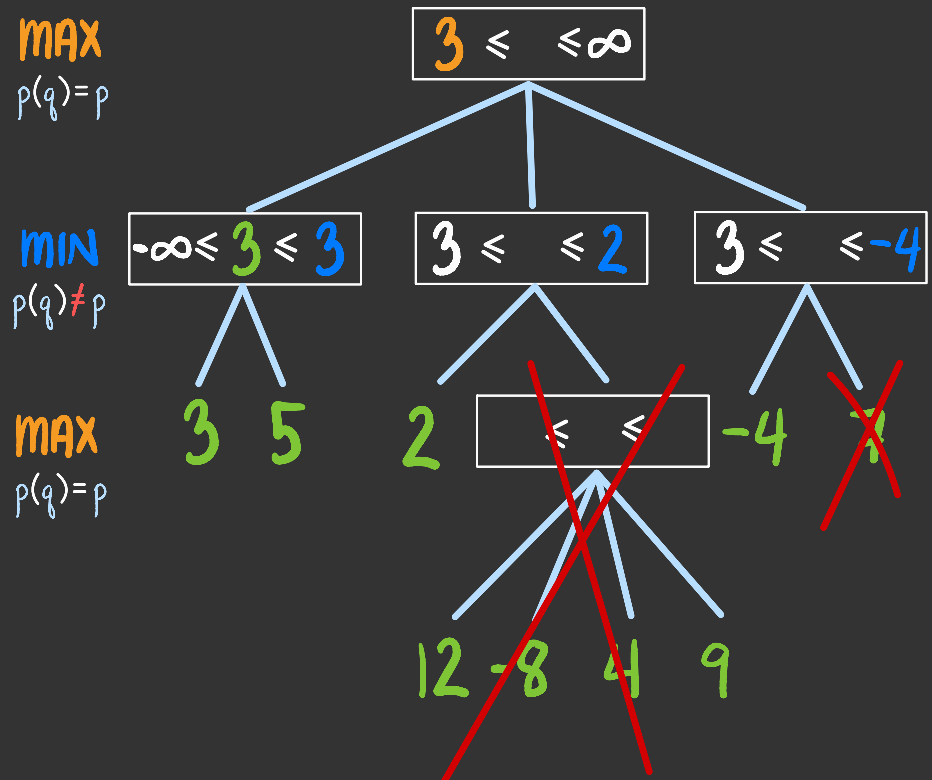
$\textsc{Minimax}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

- if $q \in F$: return $U(q, p)$
- bestvalue $= -\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue $= \textsc{Minimax}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue $= \max($bestvalue, childvalue$)$
    - $\alpha = \max(\alpha$, childvalue$)$
  - else:
    - bestvalue $= \min($bestvalue, childvalue$)$
    - $\beta = \min(\beta$, childvalue$)$
- return bestvalue

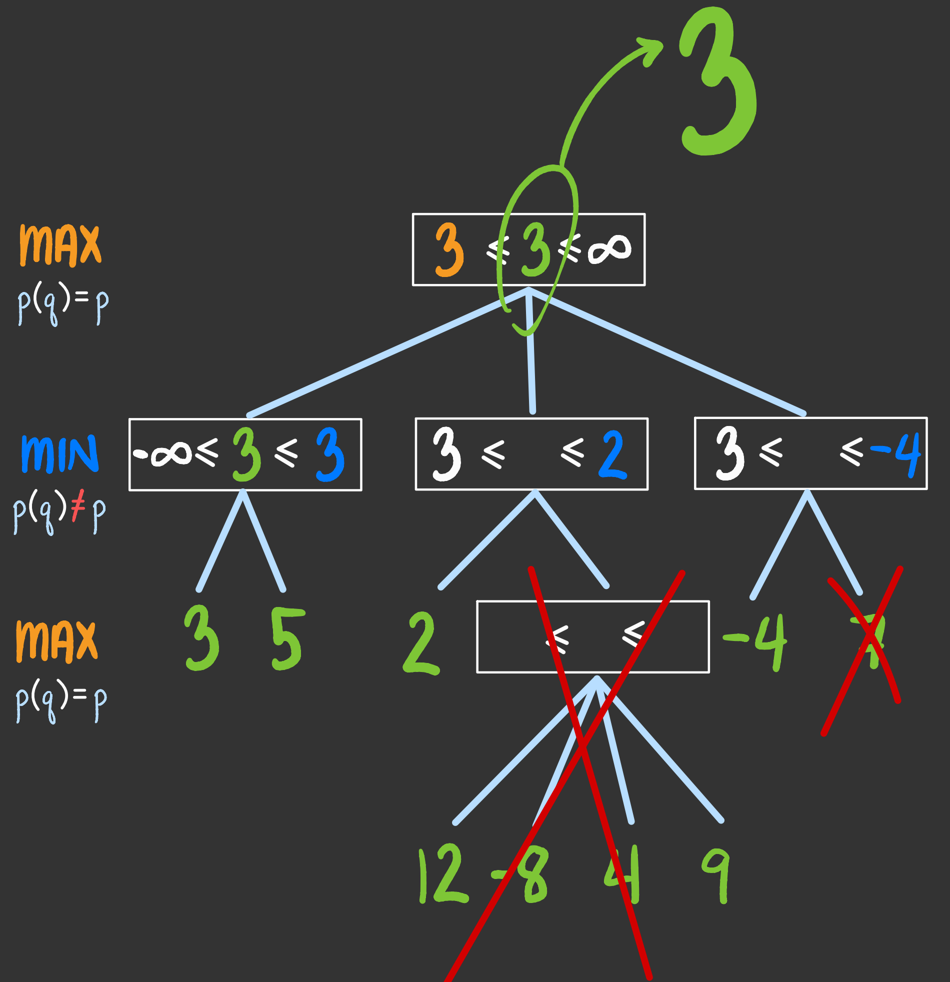


$\text{MINIMAX}(q, p, (M, U), \alpha, \beta)$: # $M = (Q, \Sigma, \Delta, q_0, F)$

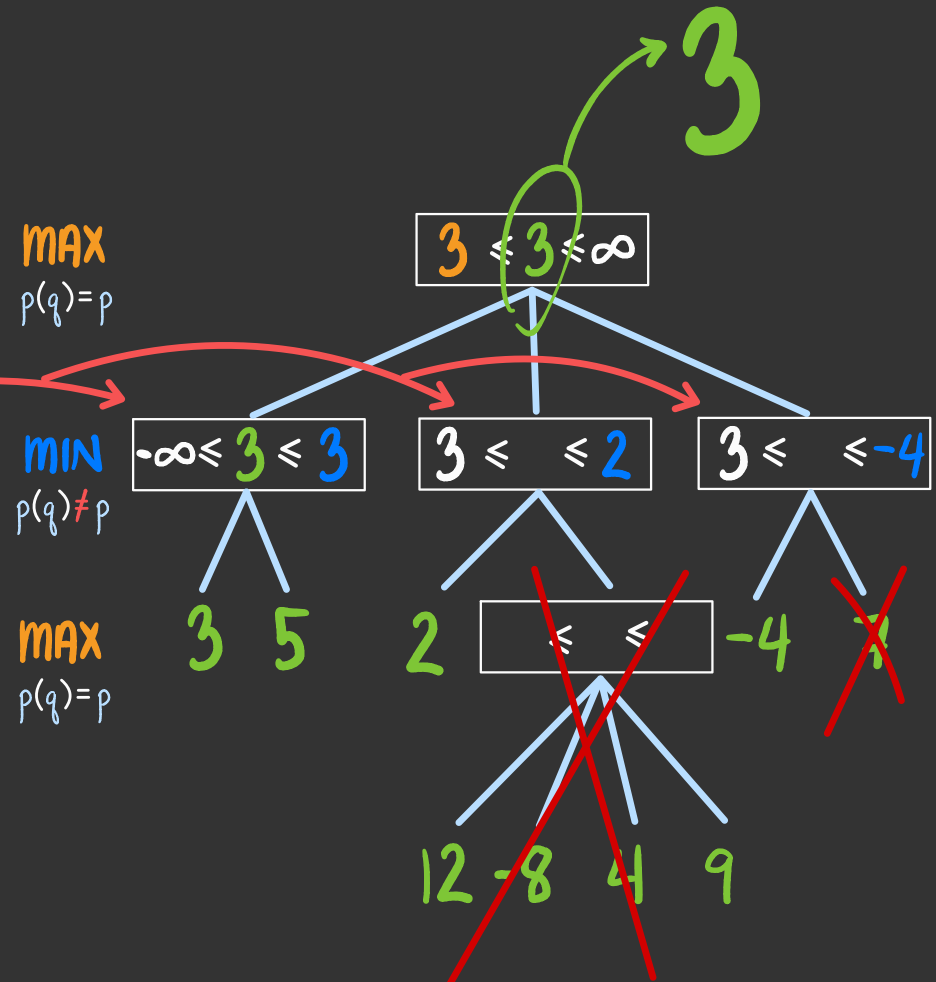
- if $q \in F$: return $U(q, p)$
- bestvalue = $-\infty$ if $p(q) = p$ else ∞
- for $\langle q, \sigma, q' \rangle \in \Delta$:
  - if $\alpha \geq \beta$: return bestvalue
  - childvalue = $\text{MINIMAX}(q', p, (M, U), \alpha, \beta)$
  - if $p(q) = p$:
    - bestvalue = max(bestvalue, childvalue)
    - α = max(α, childvalue)
  - else:
    - bestvalue = min(bestvalue, childvalue)
    - β = min(β, childvalue)
- return bestvalue



```
MINIMAX(q, p, (M,U), α, β):  # M = (Q, Σ, Δ, q₀, F)
  ▸ if q ∈ F: return U(q,p)
  ▸ bestvalue = -∞ if p(q)=p else ∞
  ▸ for ⟨q, σ, q'⟩ ∈ Δ:
      ▸ if α ≥ β: return bestvalue
      ▸ childvalue = MINIMAX(q', p, (M,U), α, β)
      ▸ if p(q)=p :
          ▸ bestvalue = max(bestvalue, childvalue)
          ▸ α = max(α, childvalue)
      ▸ else :
          ▸ bestvalue = min(bestvalue, childvalue)
          ▸ β = min(β, childvalue)
  ▸ return bestvalue
```

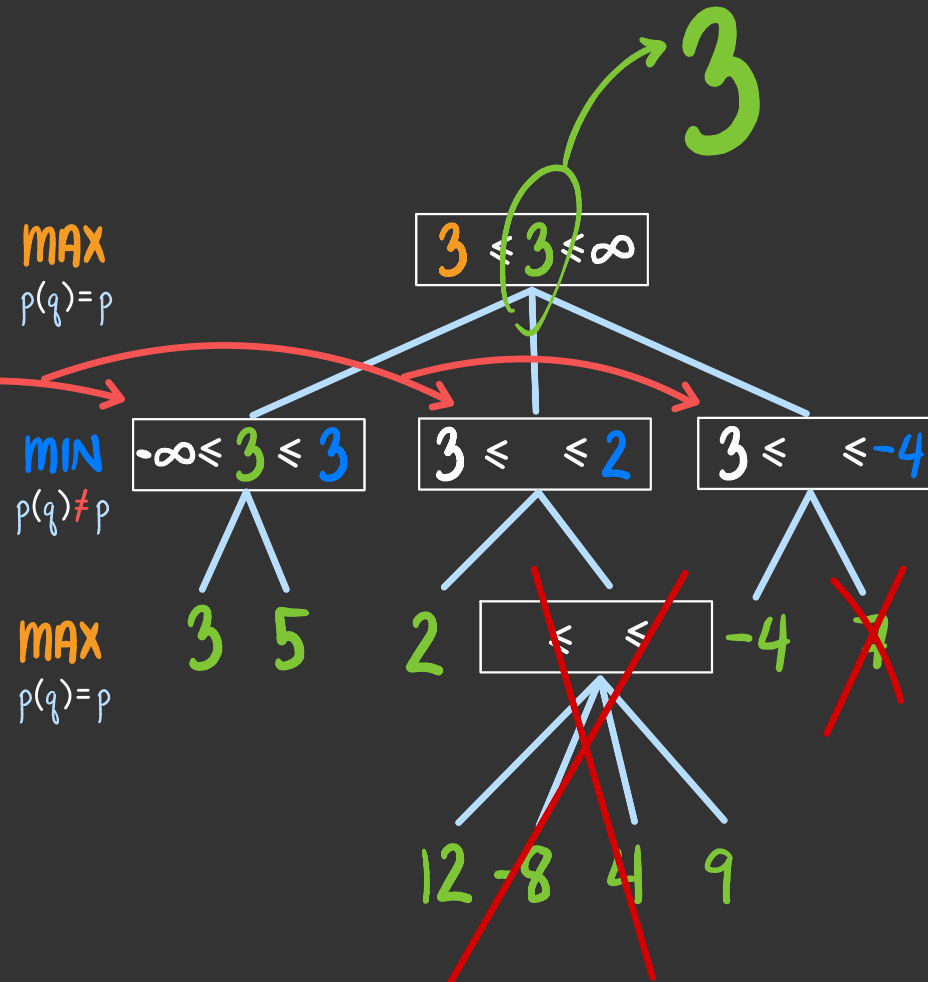


what do these **bounds** describe?



your answer **here**

what do these **bounds** describe?



conditions under which this subtree can affect the value of its parent