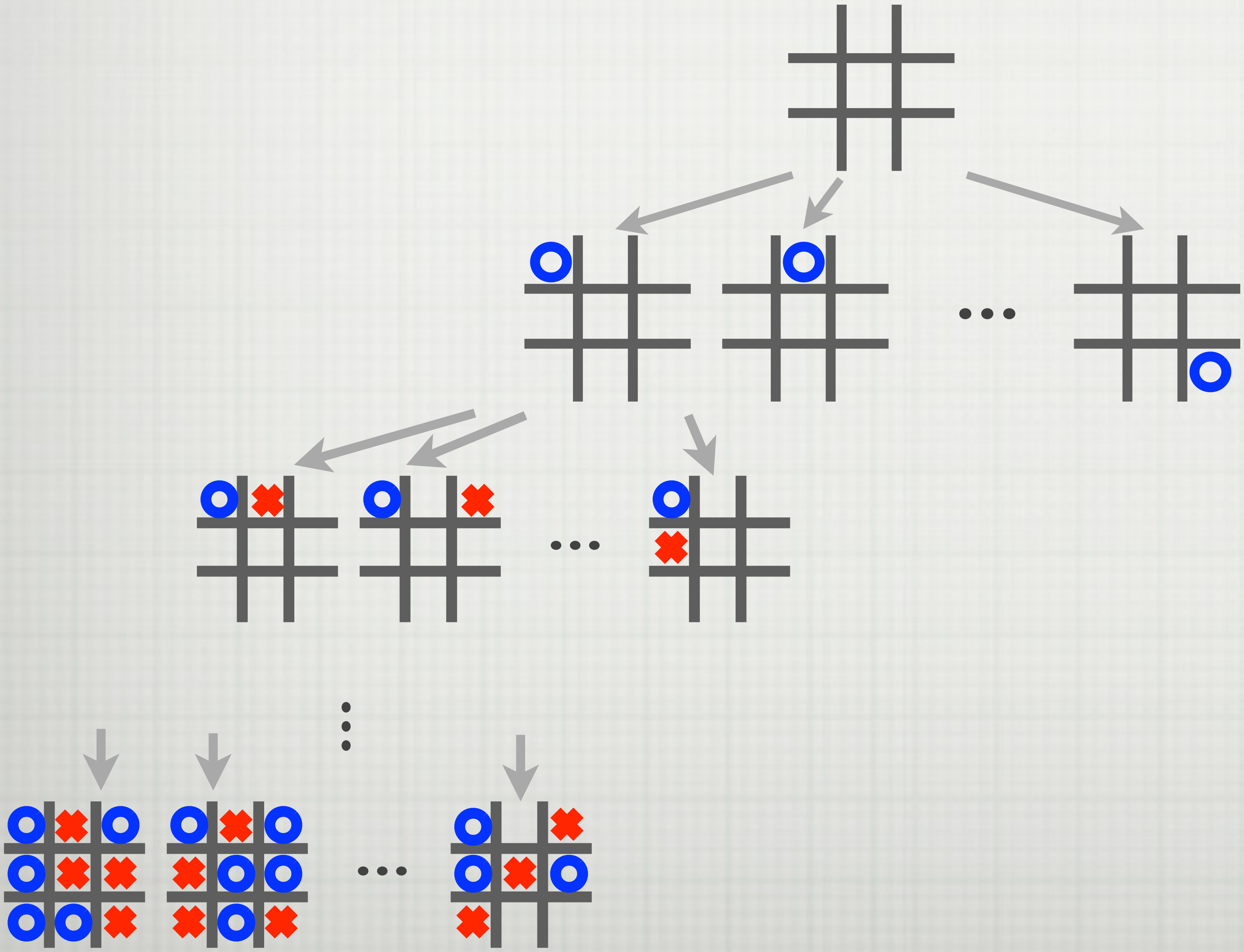
JOHDATUS TEKOÄLYYN

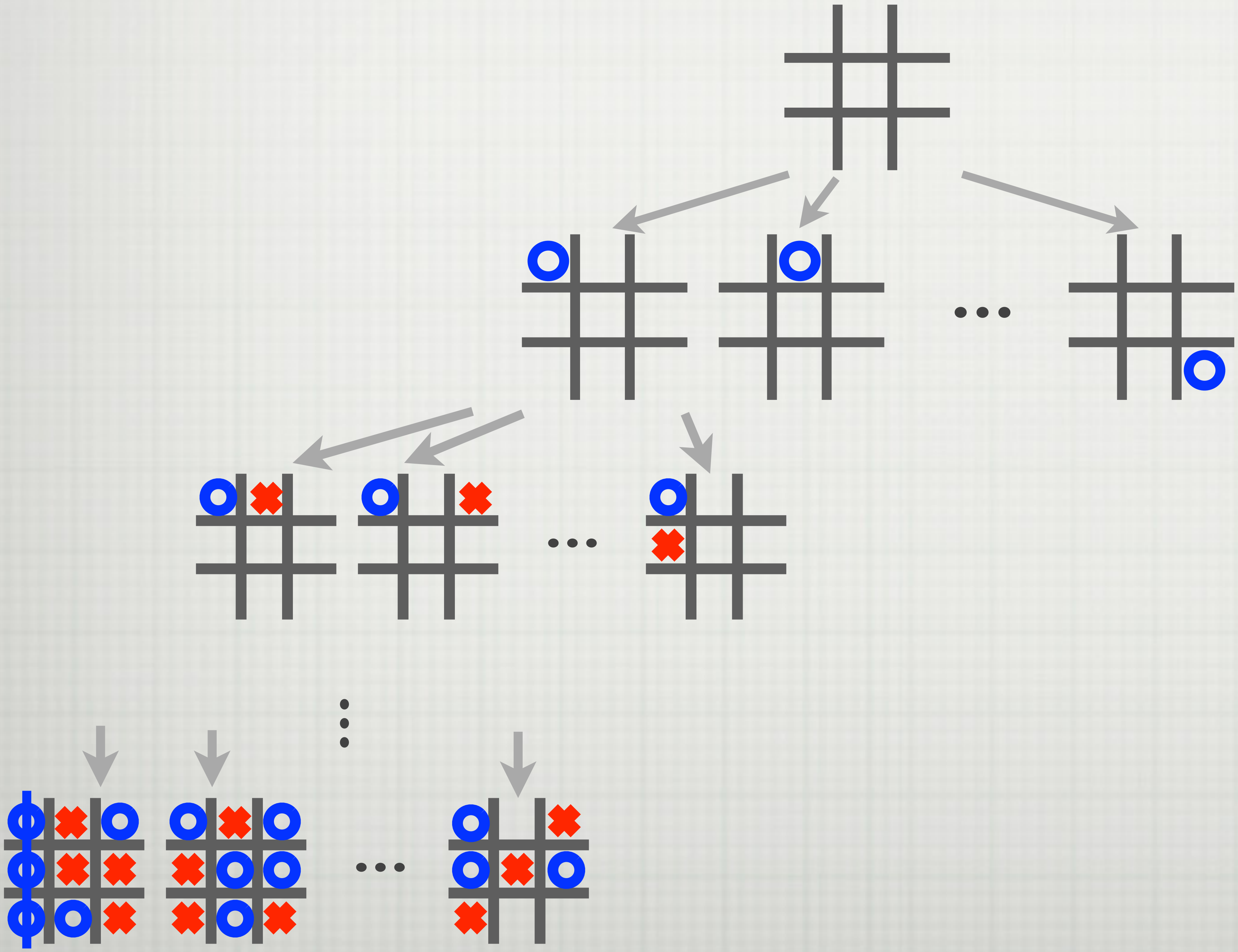
TEEMU ROOS

HELSINGIN YLIOPISTO

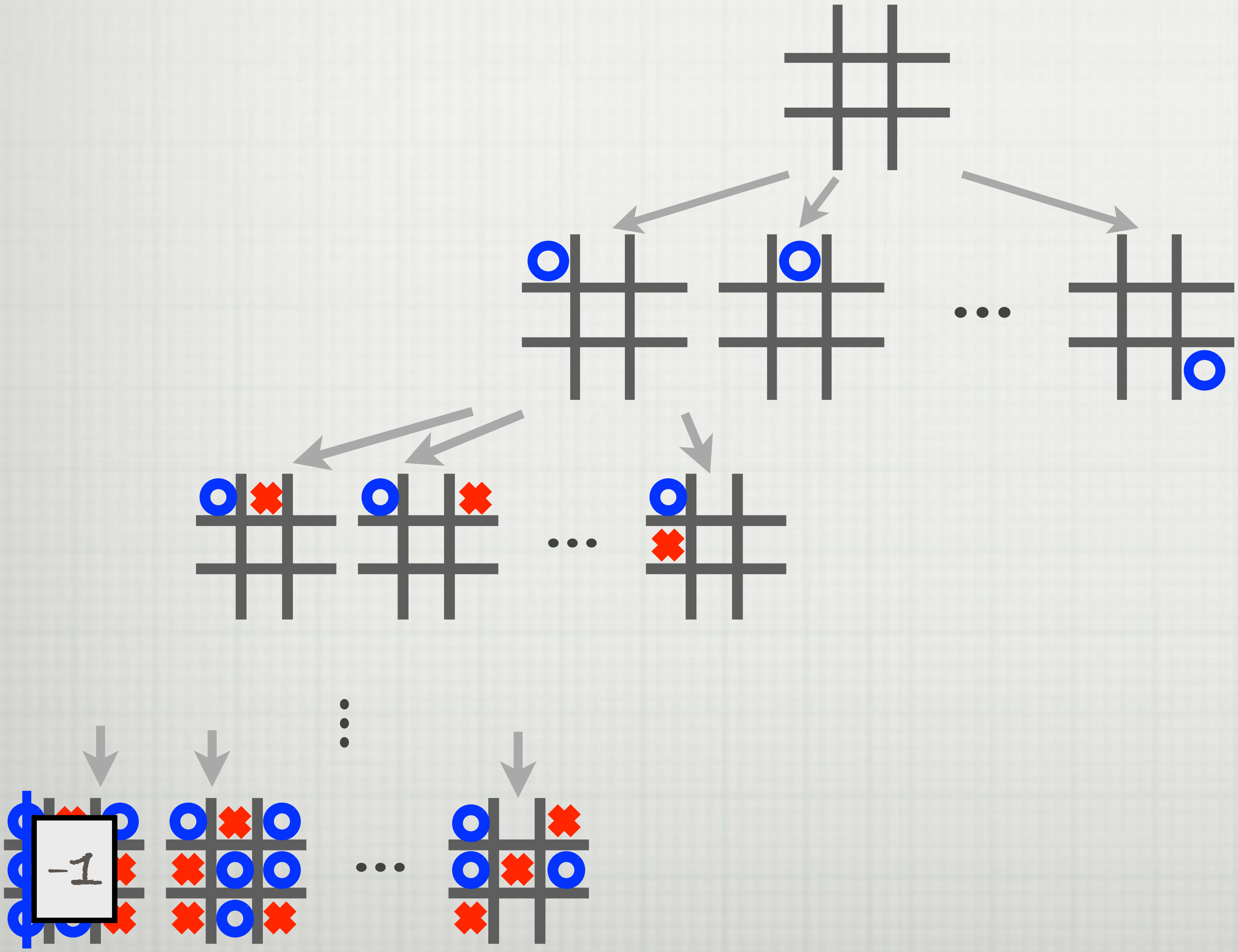
PELIPUU



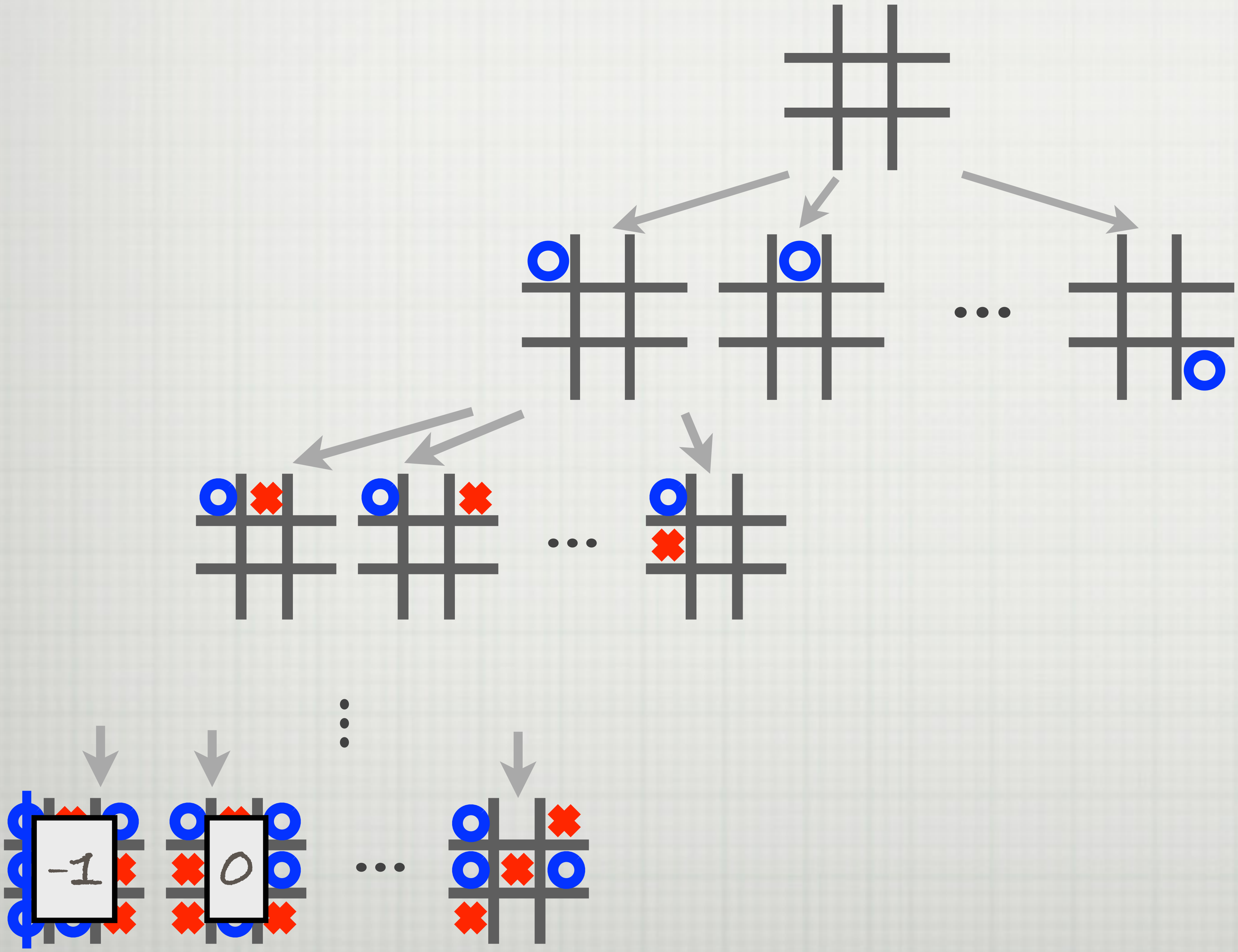
PELIPUU



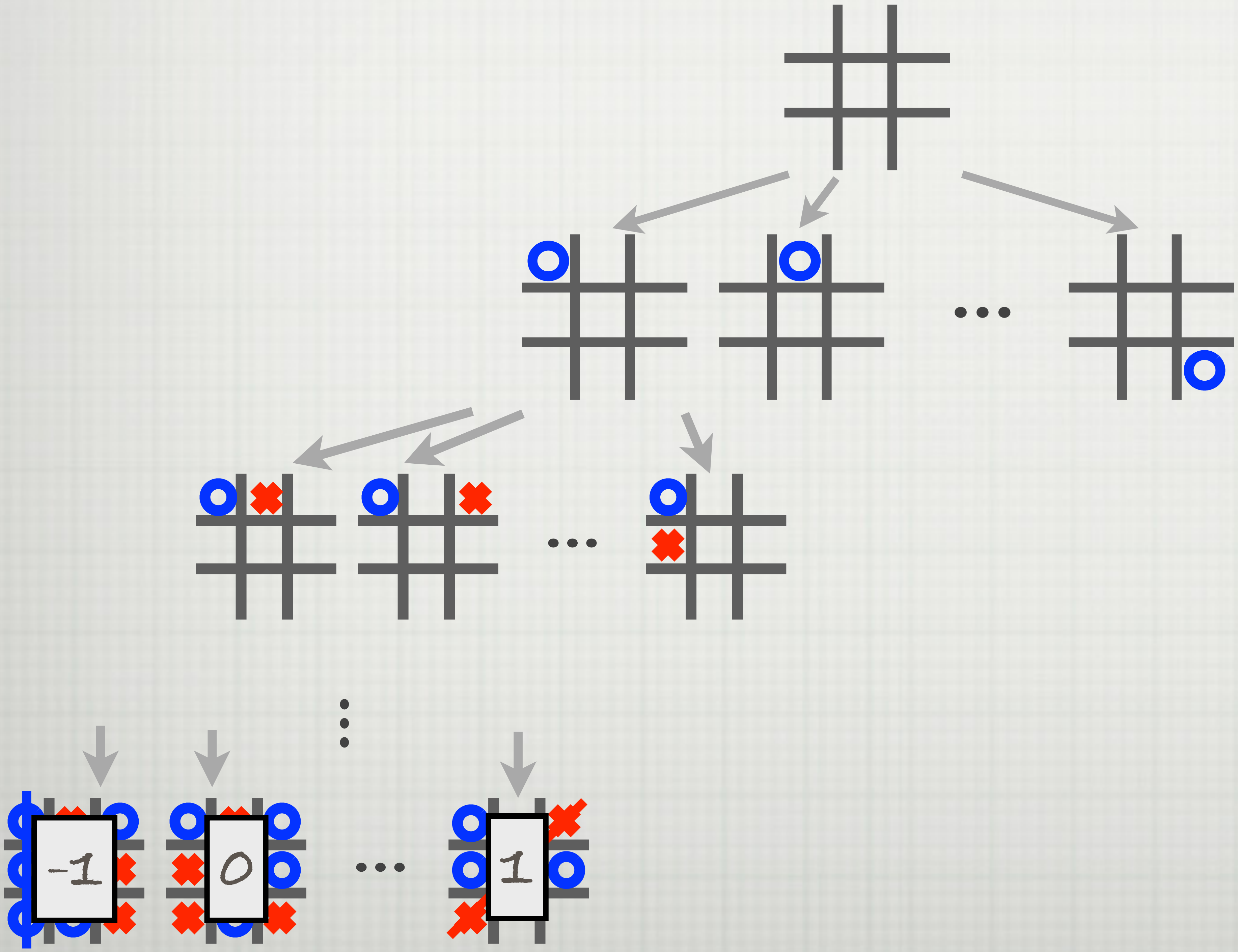
PELIPUU



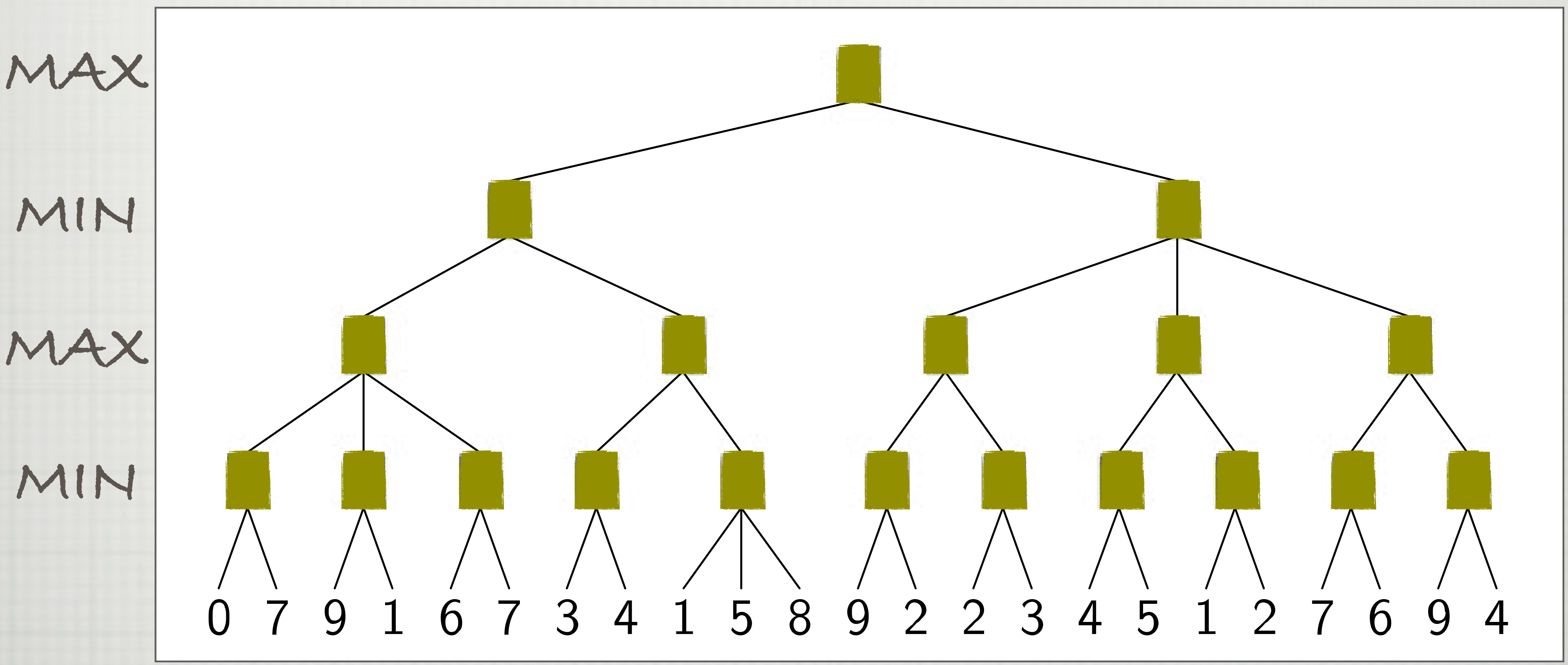
PELIPUU



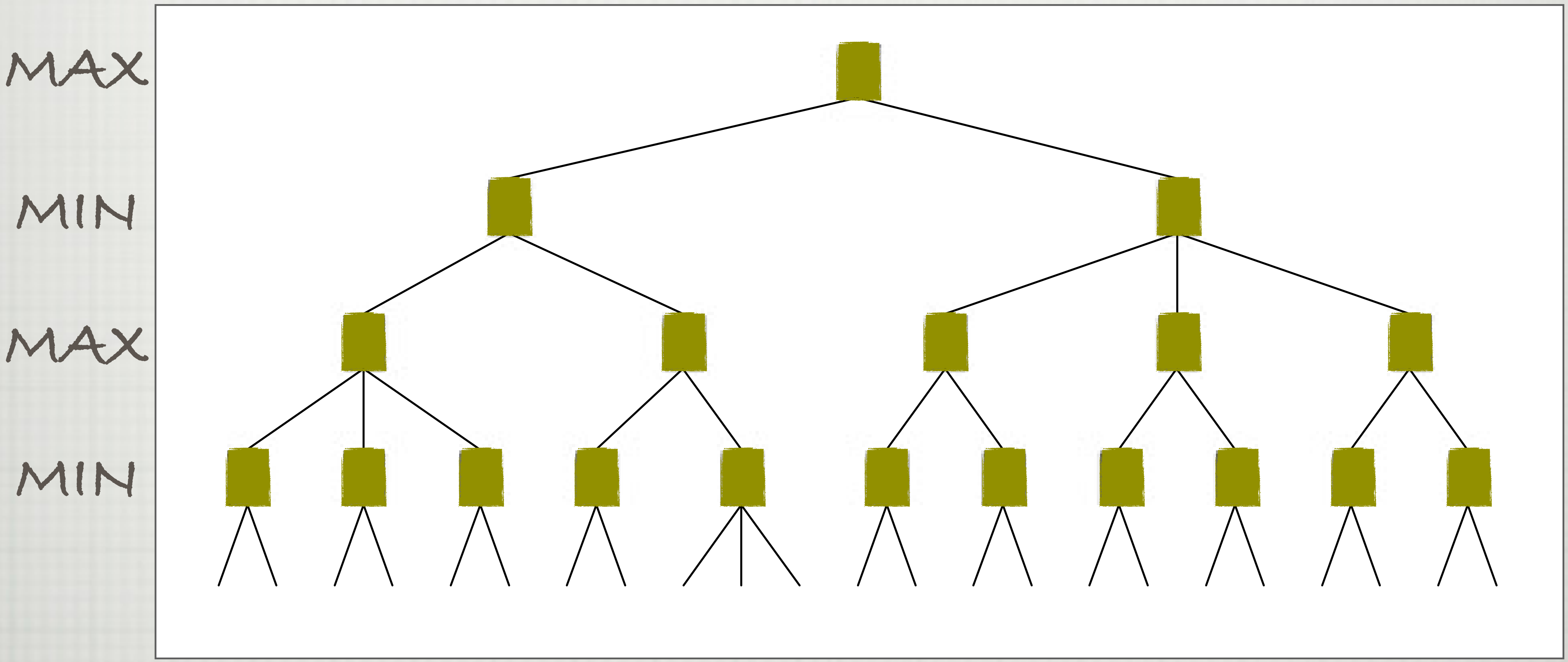
PELIPUU



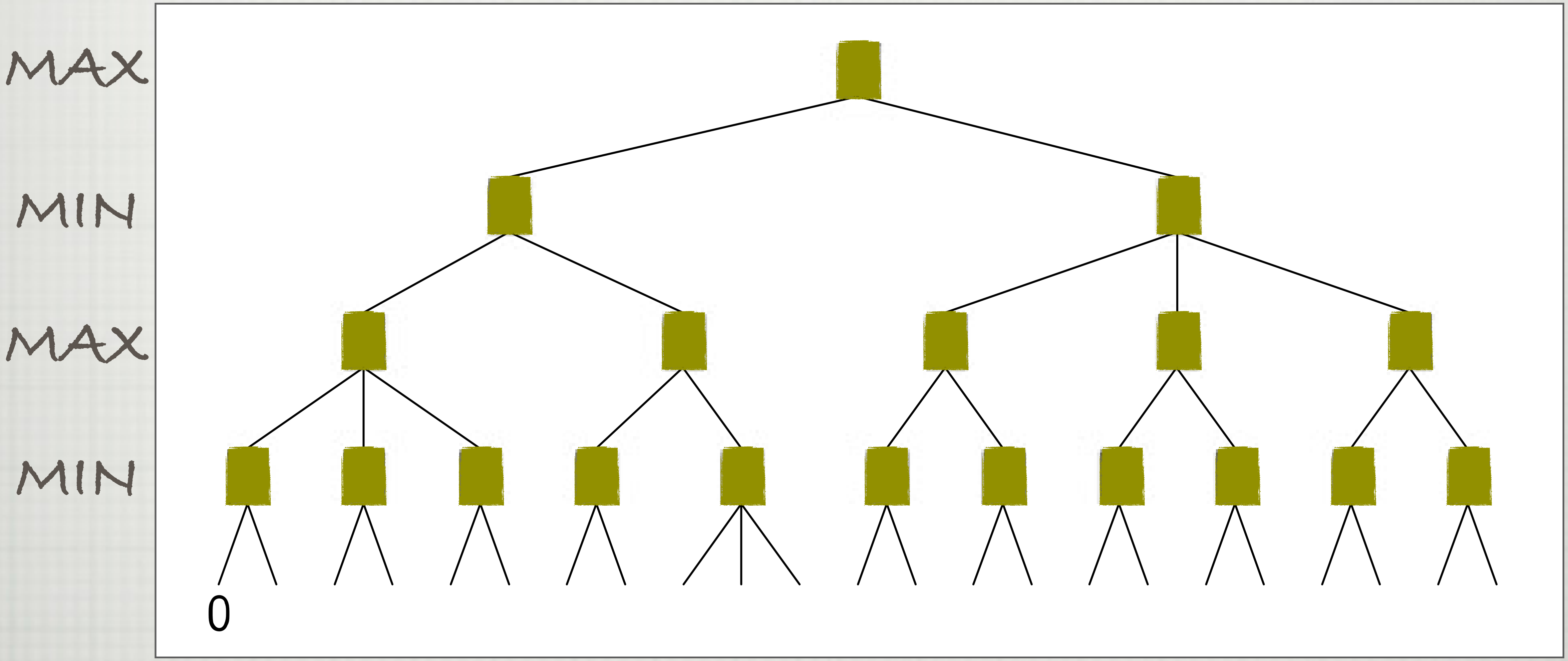
PELIPUU



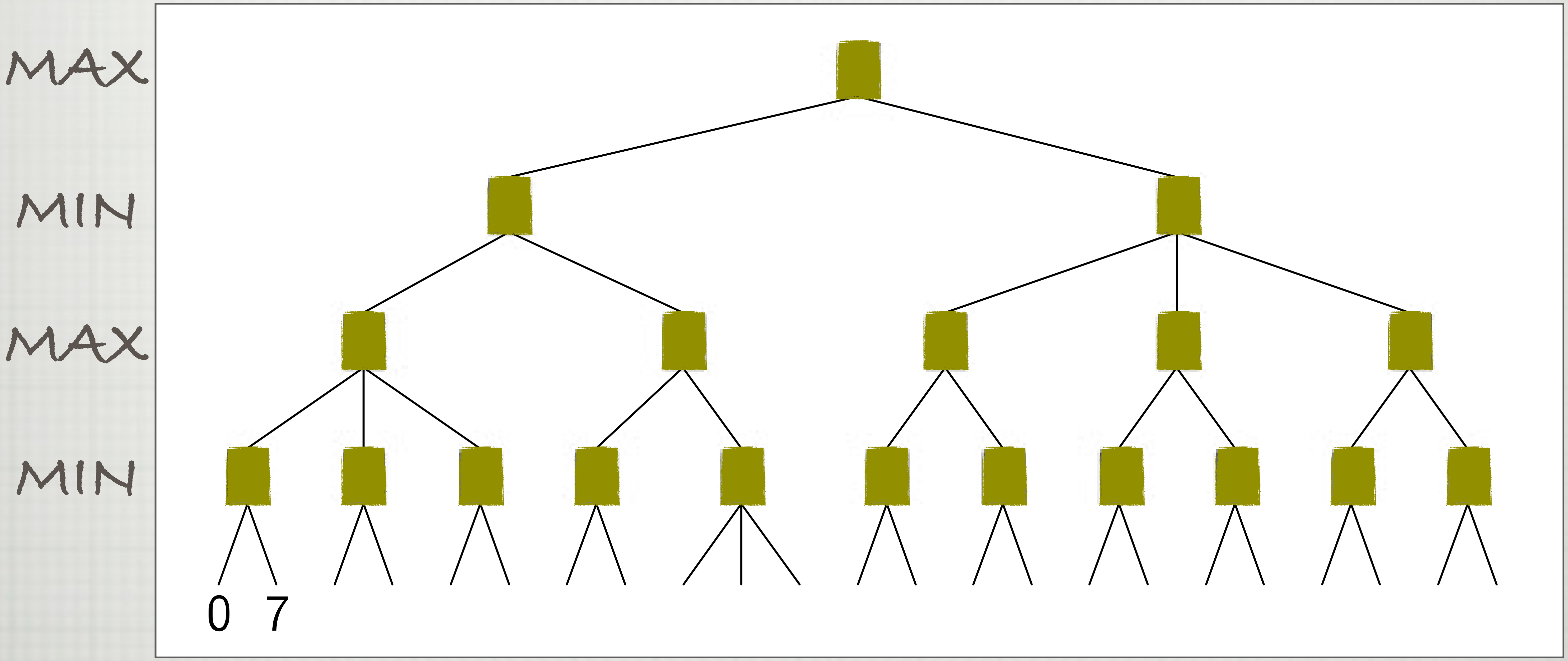
PELIPUU



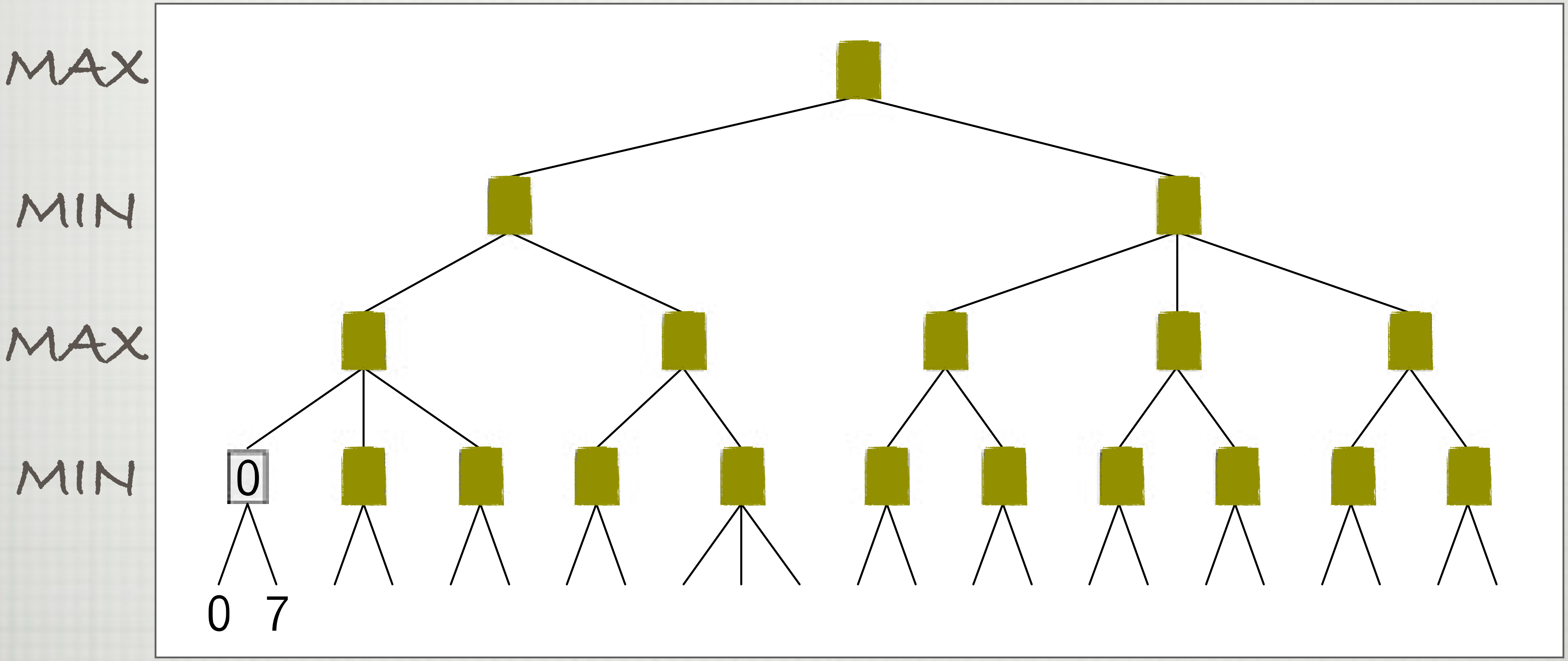
PELIPUU



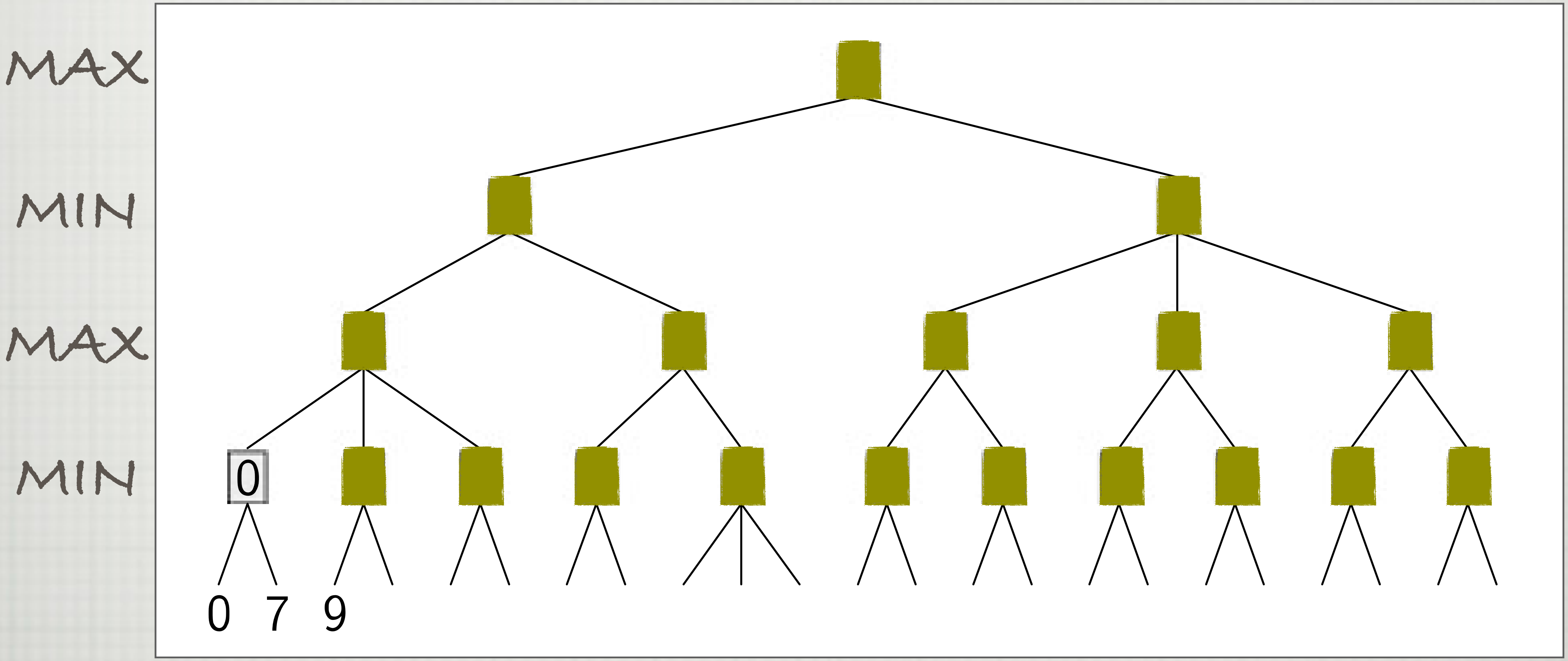
PELIPUU



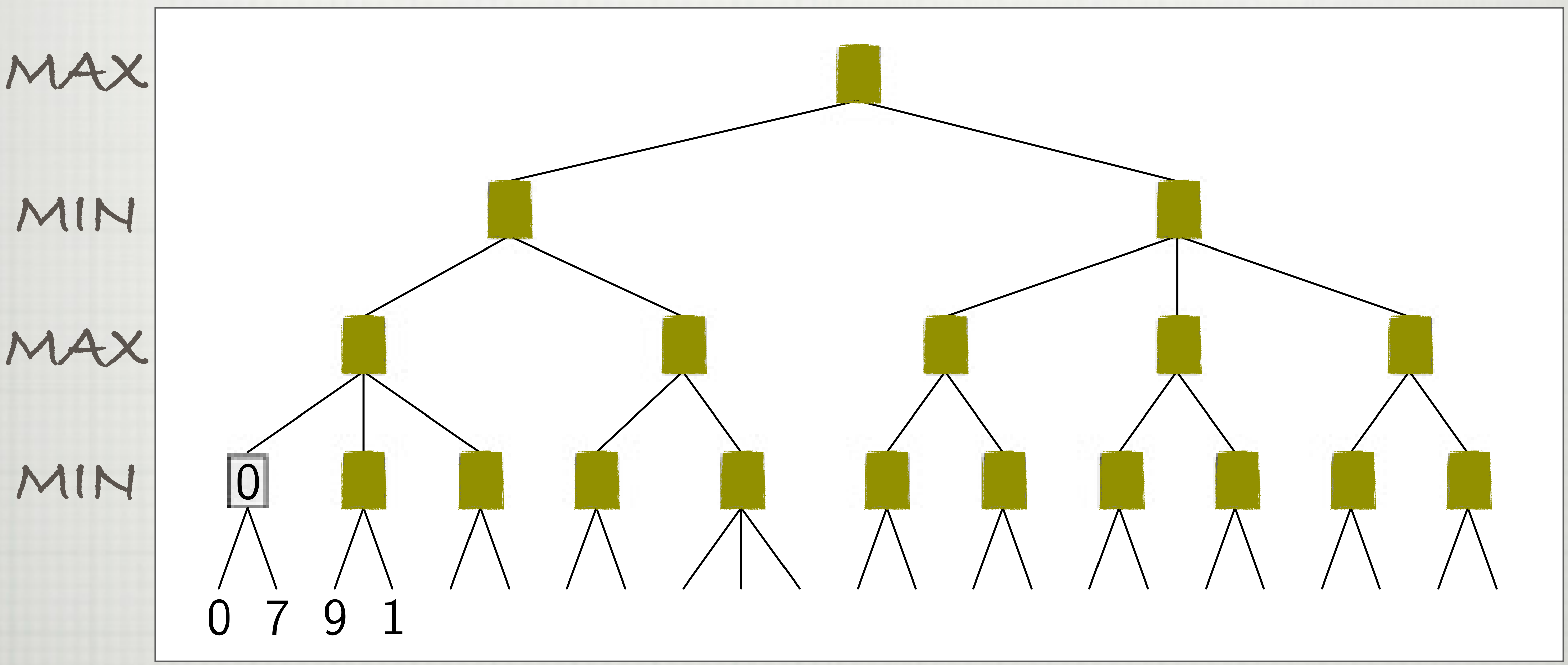
PELIPUU



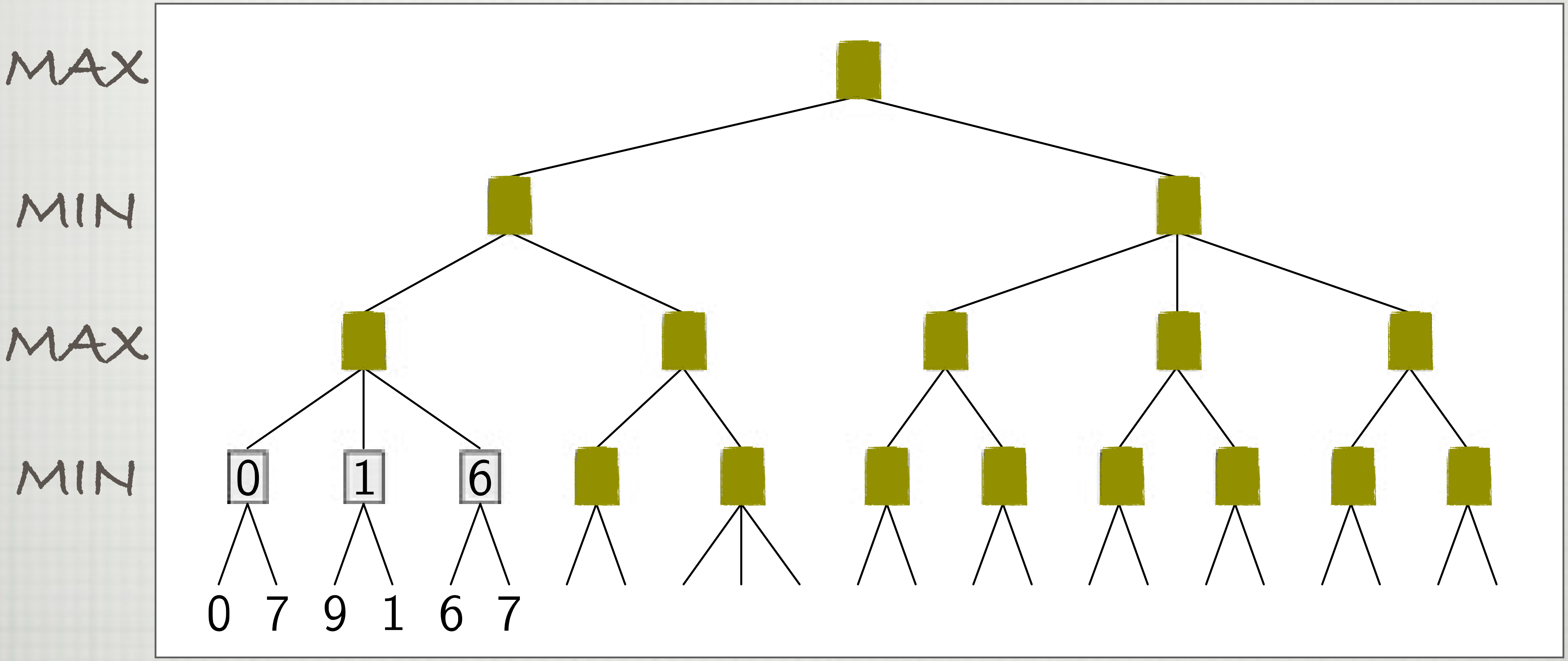
PELIPUU



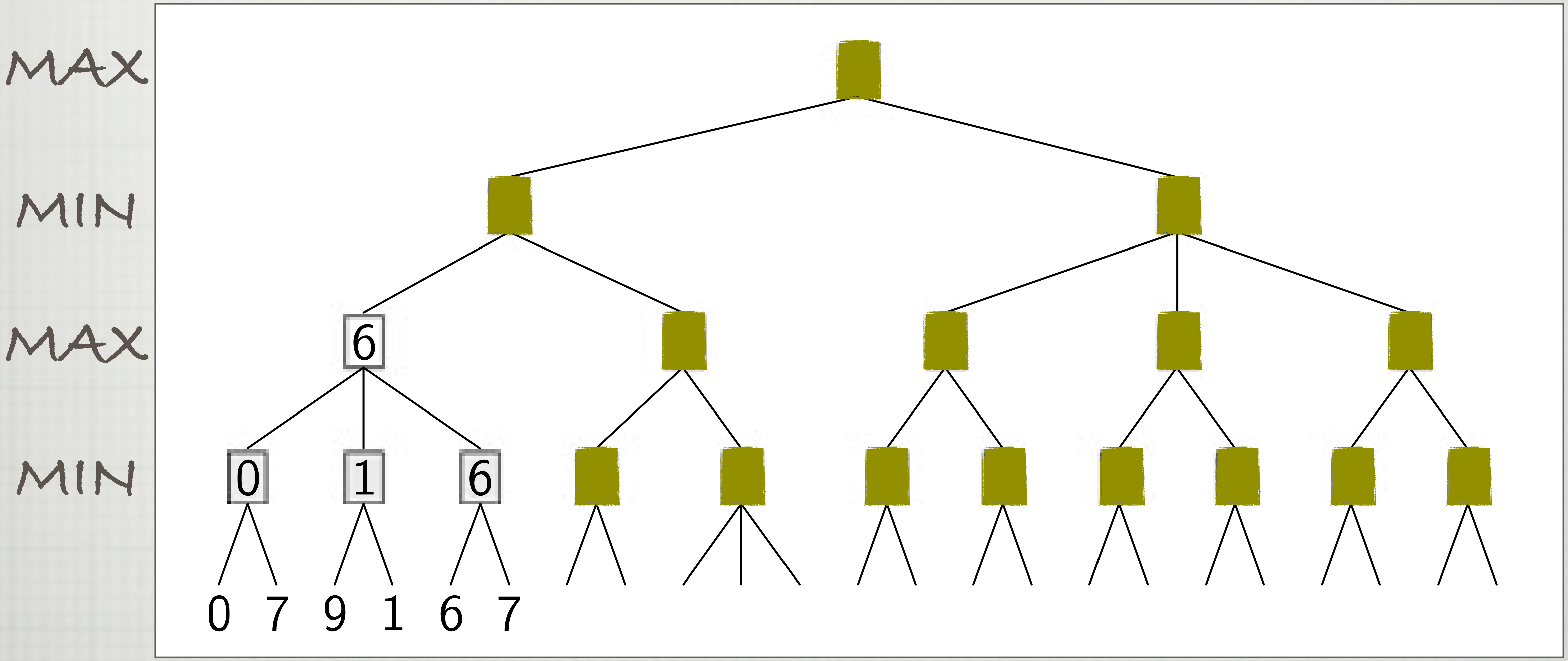
PELIPUU



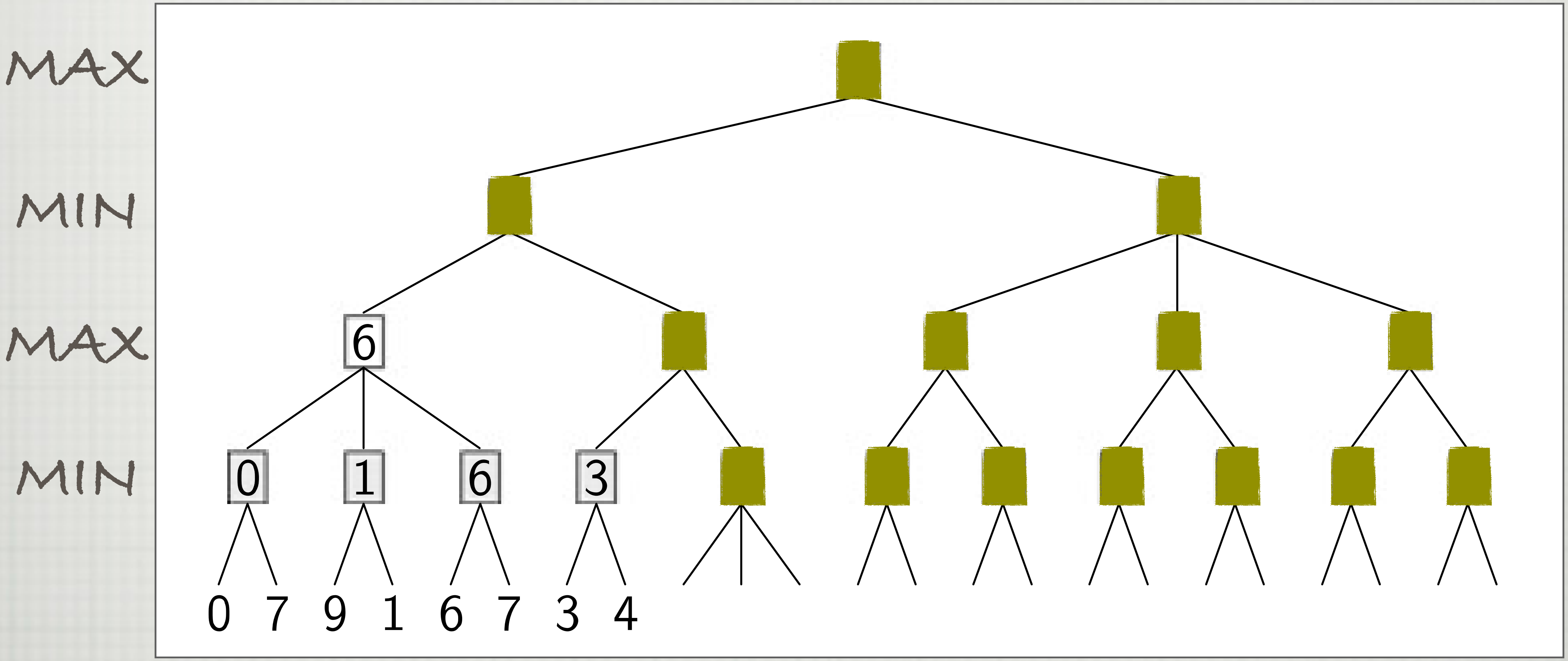
PELIPUU



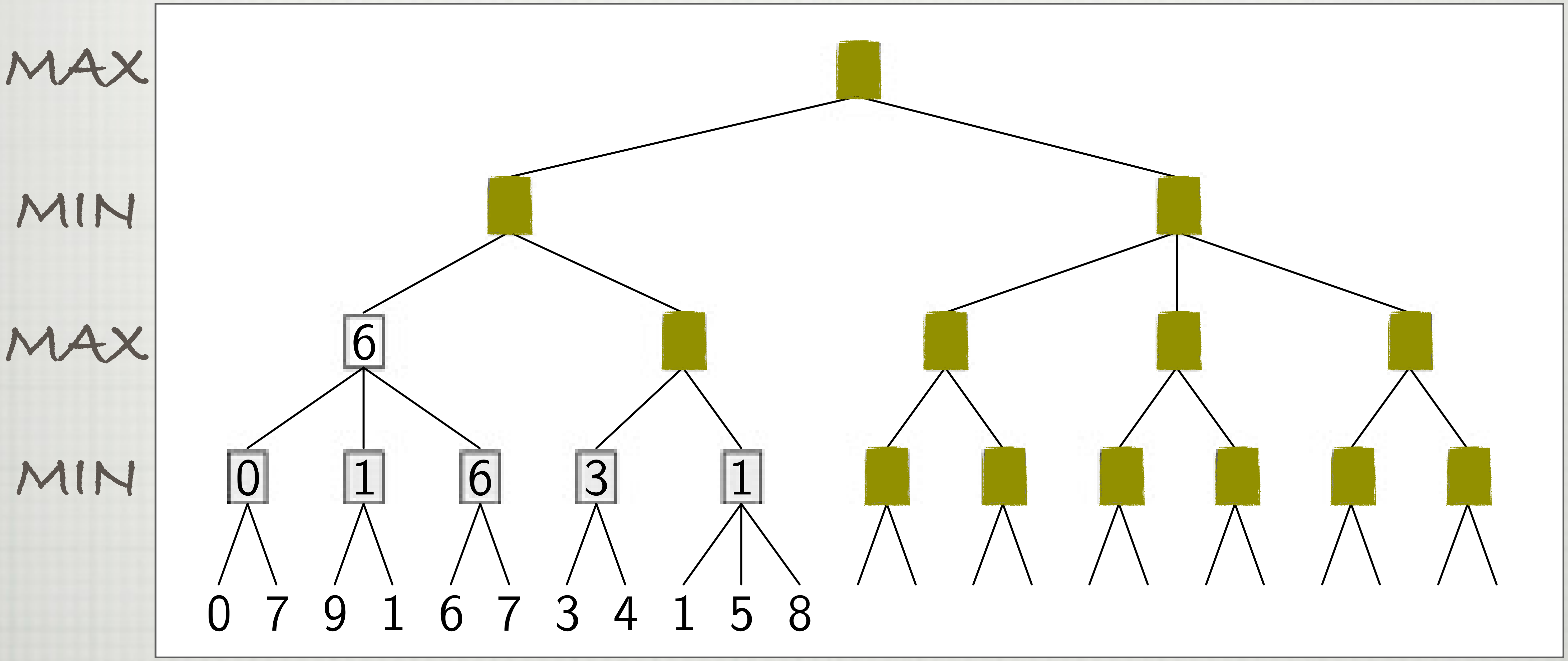
PELIPUU



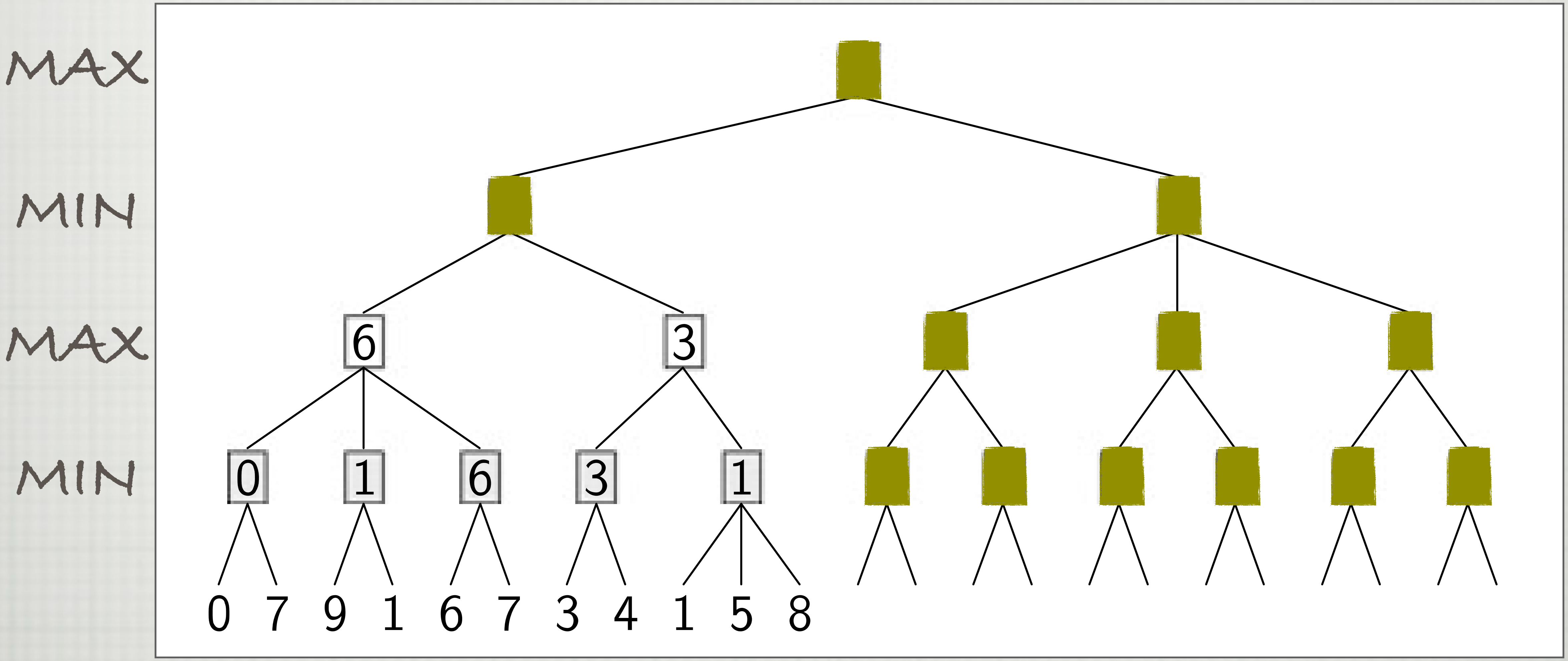
PELIPUU



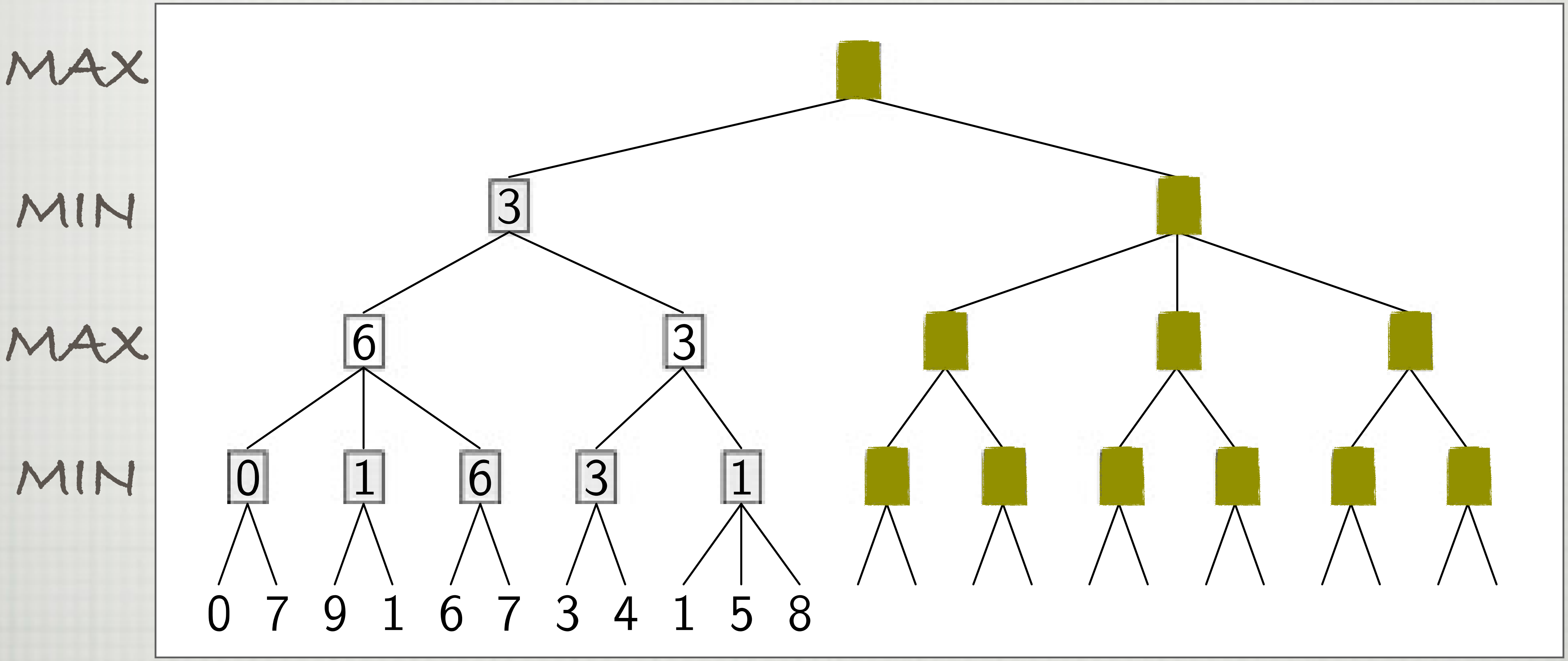
PELIPUU



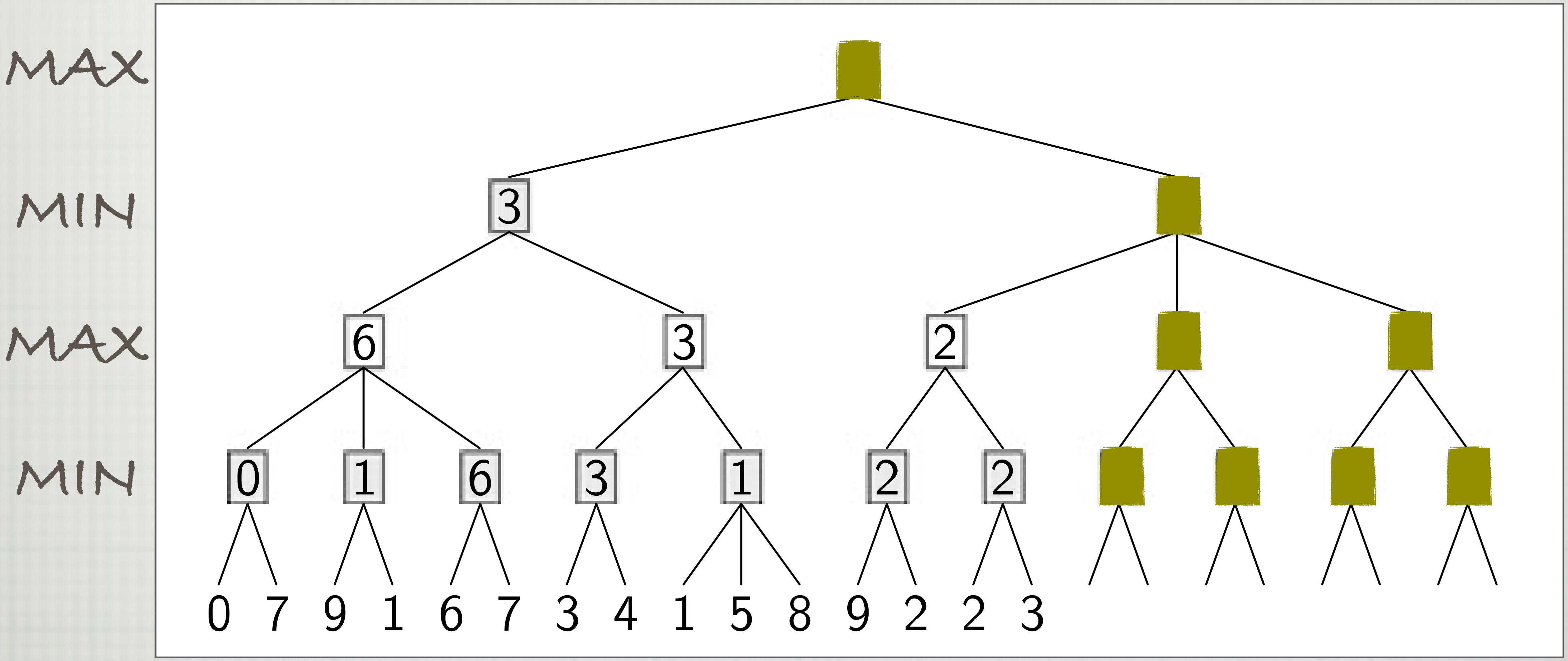
PELIPUU



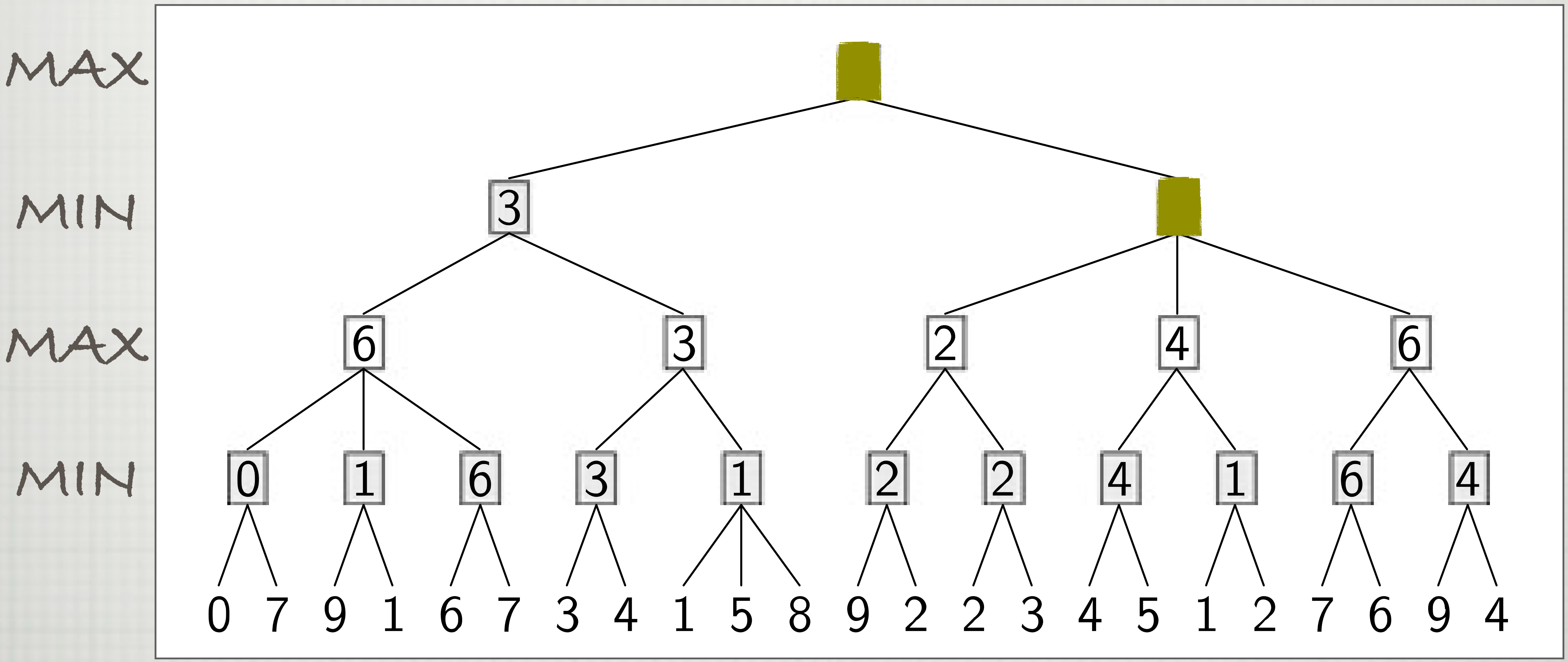
PELIPUU



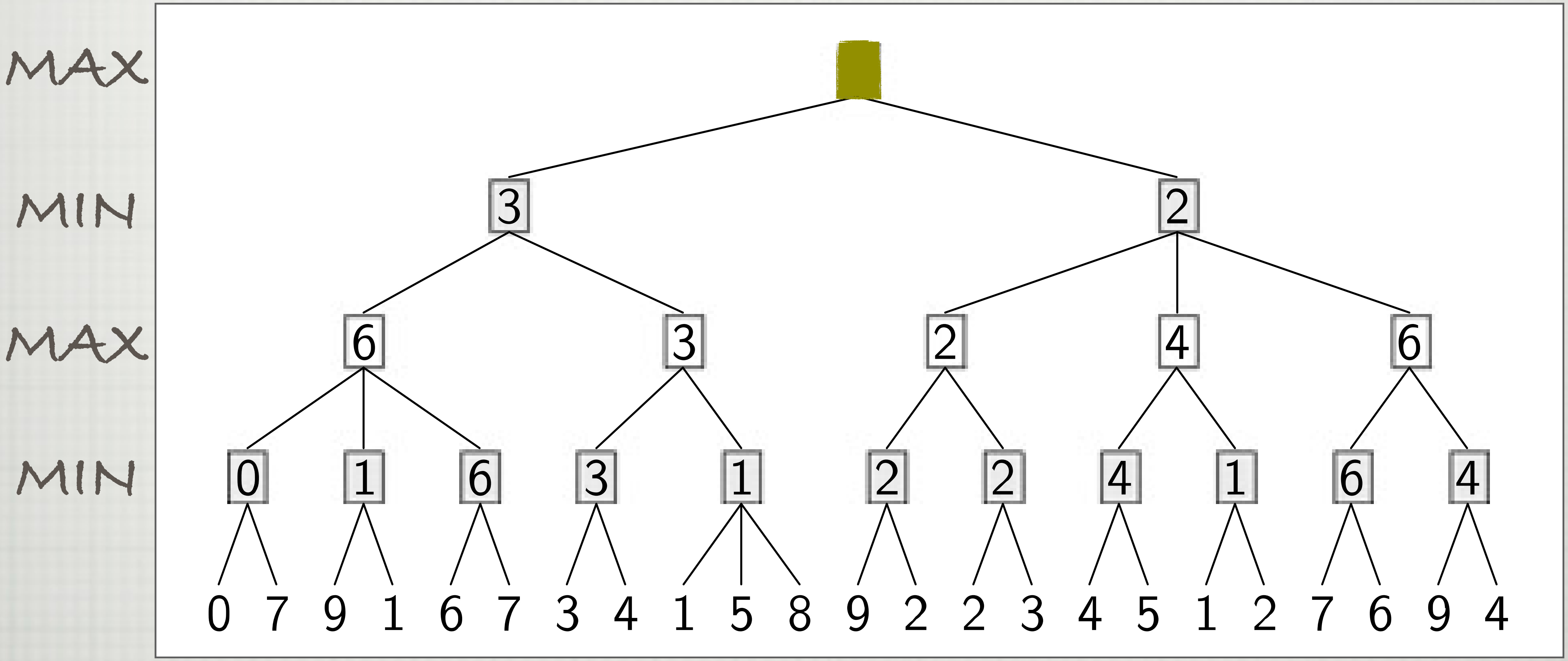
PELIPUU



PELIPUU

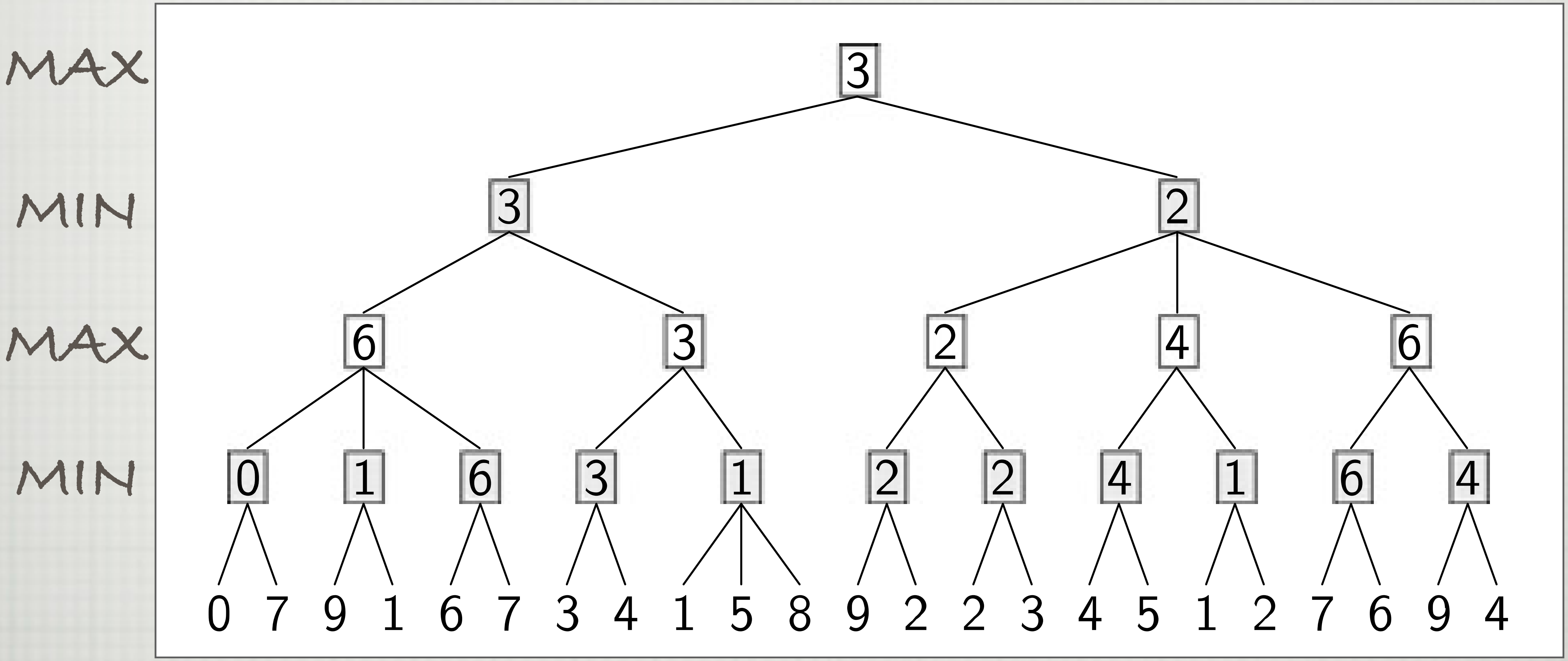

MAX
MIN
MAX
MIN
3
6 3 2 4 6
0 1 6 3 1 2 2 4 1 6 4
0 7 9 1 6 7 3 4 1 5 8 9 2 2 3 4 5 1 2 7 6 9 4

PELIPUU



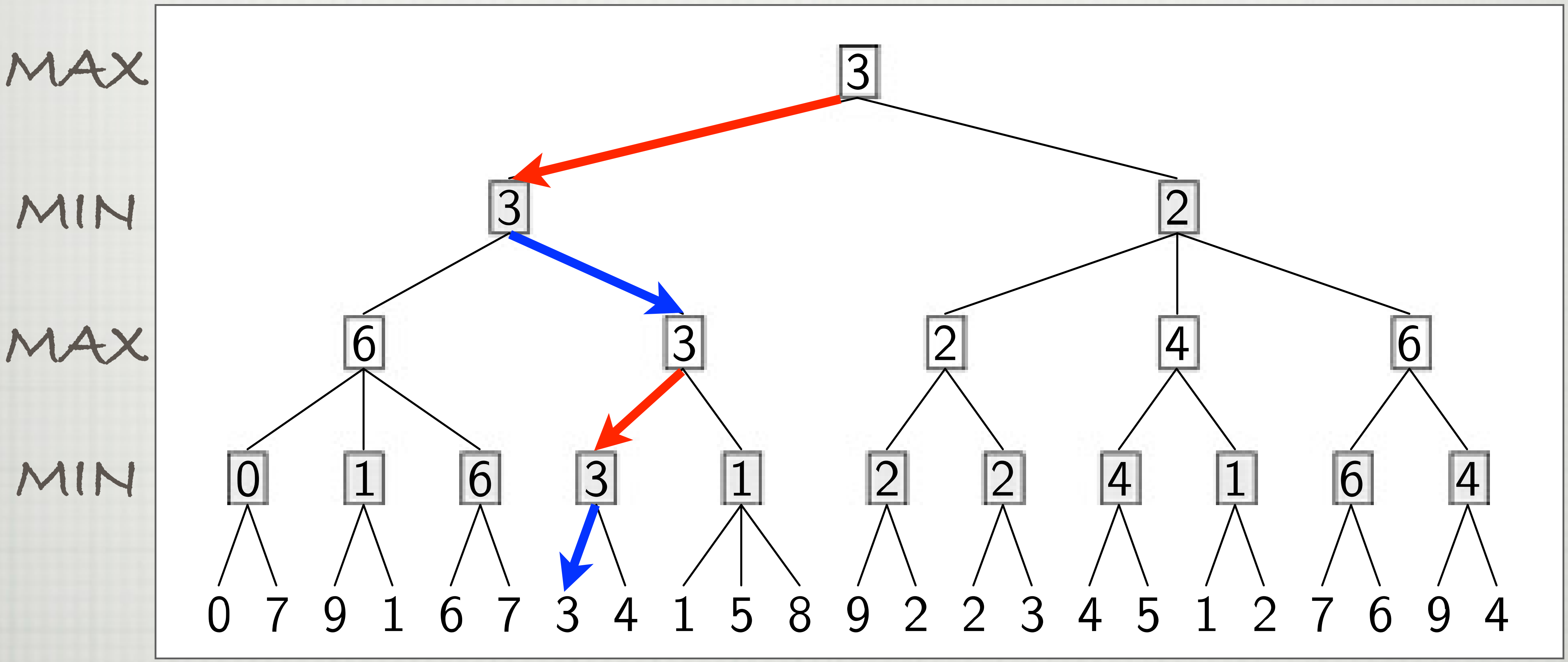
MAX
MIN
MAX
MIN
3 2
6 3 2 4 6
0 1 6 3 1 2 2 4 1 6 4
0 7 9 1 6 7 3 4 1 5 8 9 2 2 3 4 5 1 2 7 6 9 4

PELIPUU



MAX
MIN
MAX
MIN
3
3
2
6
3
2
4
6
0
1
6
3
1
2
2
4
1
6
4
0 7 9 1 6 7 3 4 1 5 8 9 2 2 3 4 5 1 2 7 6 9 4

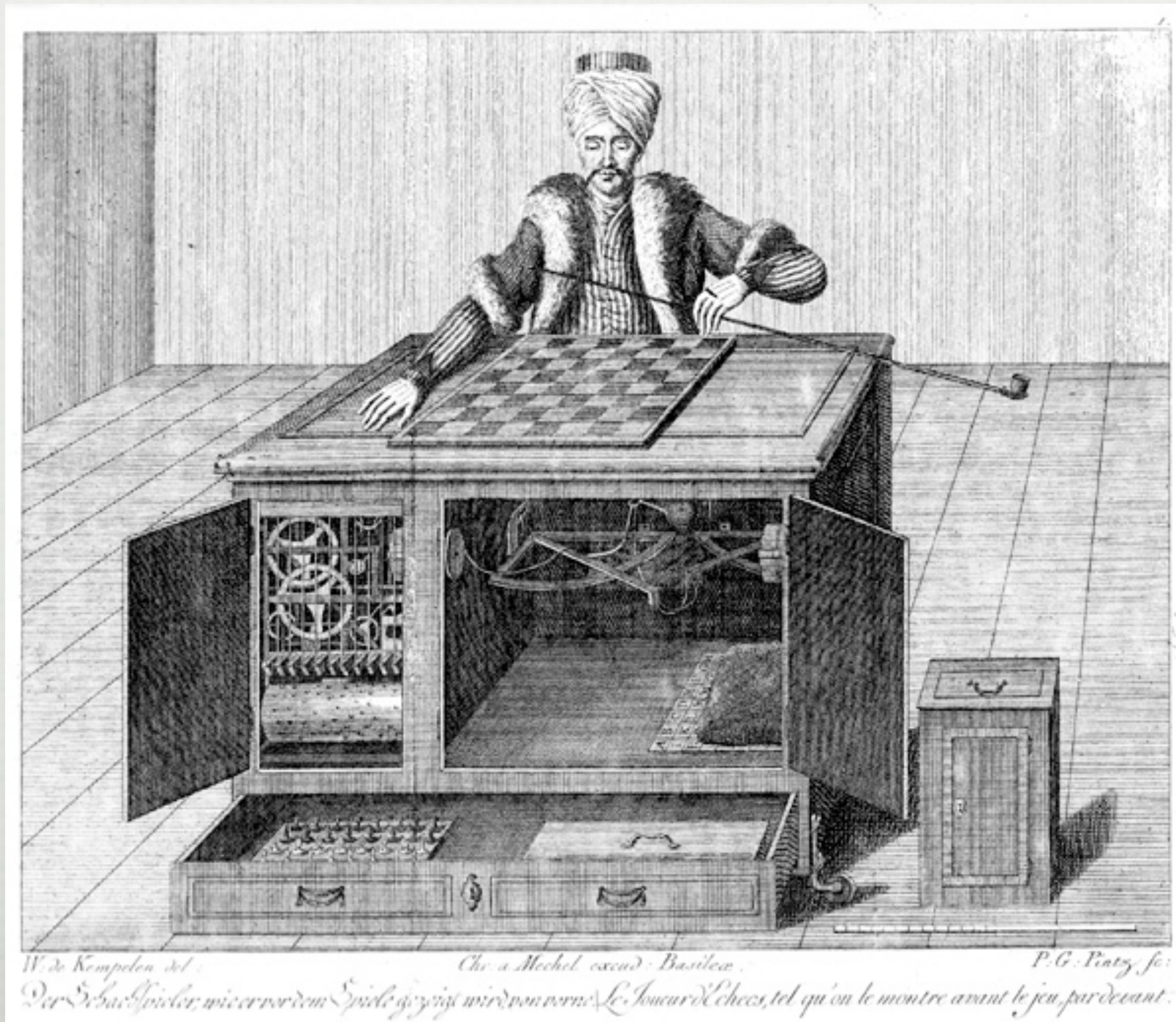
PELIPUU


MAX
MIN
MAX
MIN
3
3 2
6 3 2 4 6
0 1 6 3 1 2 2 4 1 6 4
0 7 9 1 6 7 3 4 1 5 8 9 2 2 3 4 5 1 2 7 6 9 4

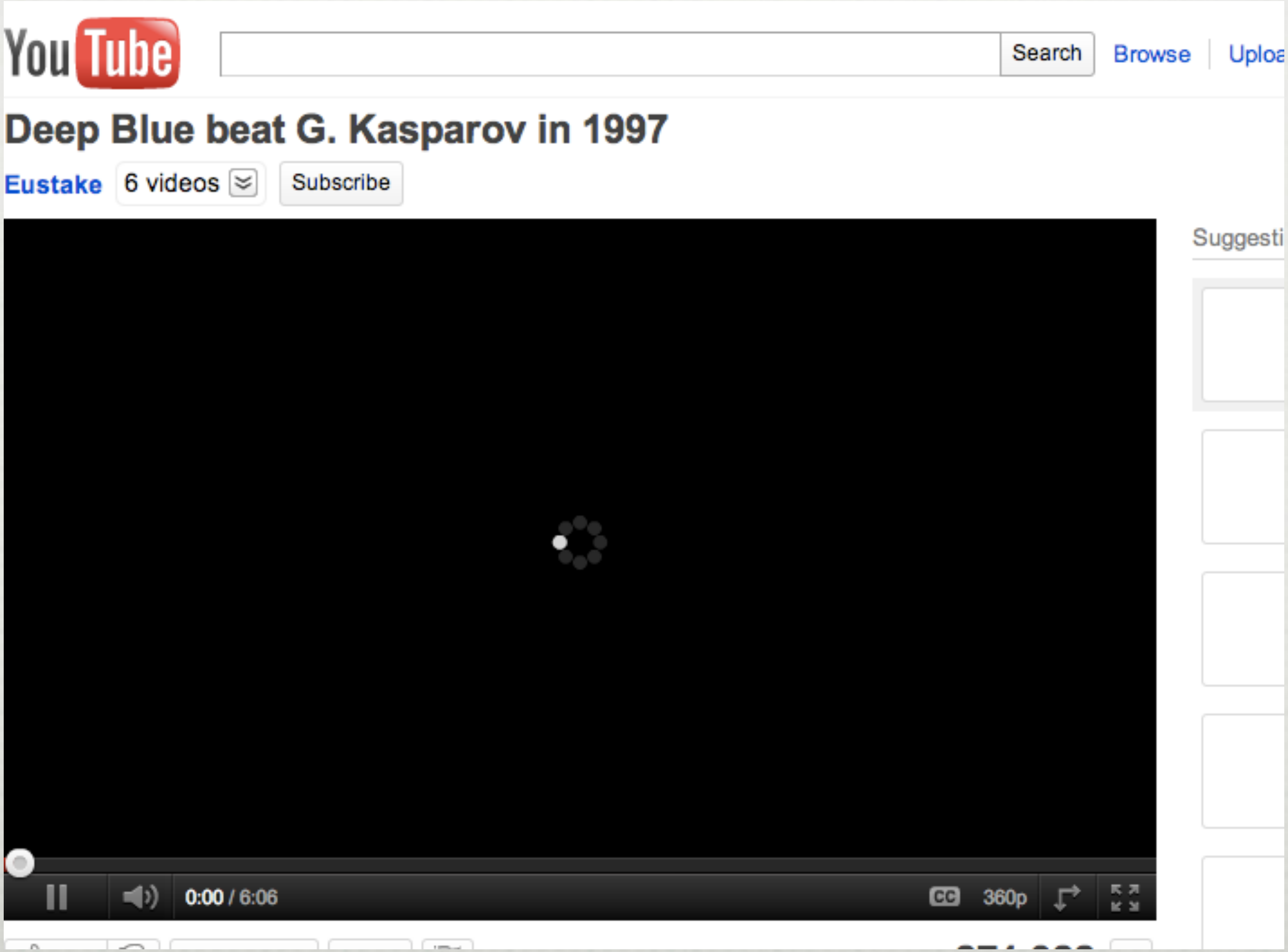
MINIMAX

```
MAX-ARVO(Solmu)

  if LOPPUTILA(Solmu) return(ARVO(Solmu))
  v = −∞
  for each Lapsi in Solmun lapset
      v = MAX(v, MIN-ARVO(Lapsi))
  return(v)

MIN-ARVO(Solmu)

  if LOPPUTILA(Solmu) return(ARVO(Solmu))
  v = +∞
  for each Lapsi in Solmun lapset
      v = MIN(v, MAX-ARVO(Lapsi))
  return(v)
```

SHAKKI

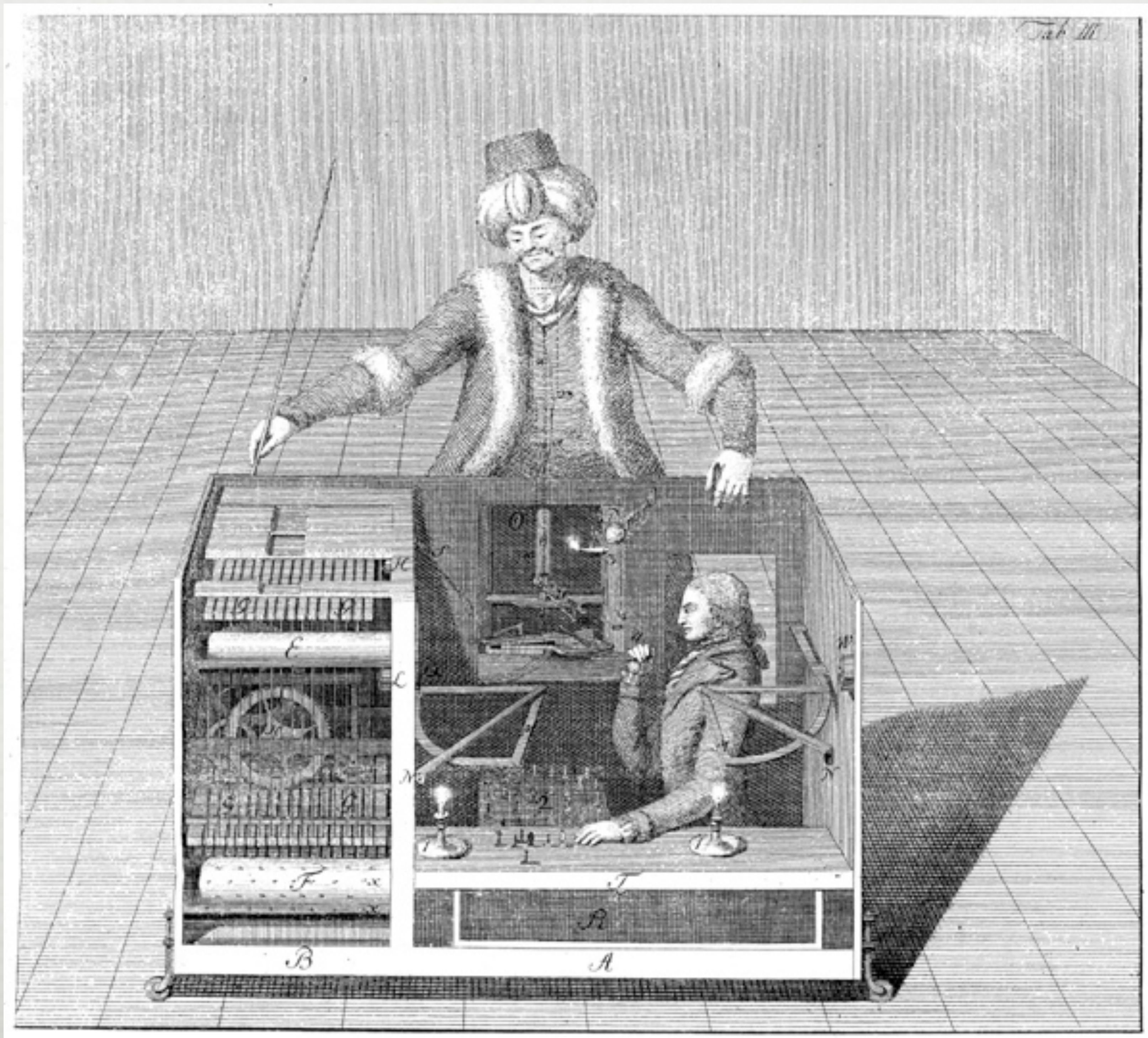

W. de Kempelen del.
Chr. a Mechel excud. Basileæ.
P. G. Pintz sc.
Der Schachspieler, wie er vor dem Spiele gezeigt wird, von vorne. Le Joueur d'Echecs, tel qu'on le montre avant le jeu, par devant.

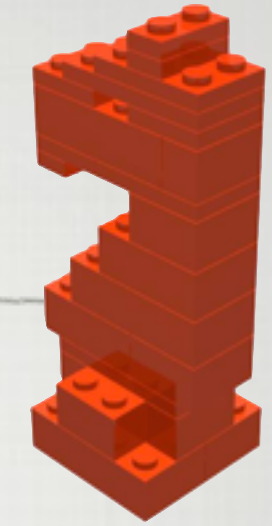
SHAKKI



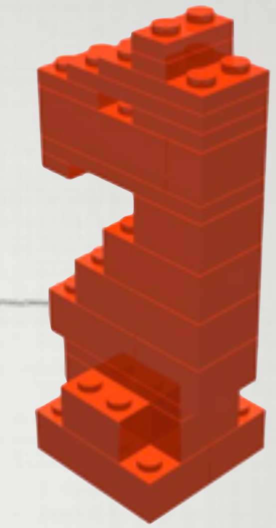
Search
Browse
Deep Blue beat G. Kasparov in 1997
Eustake
6 videos
Subscribe
0:00 / 6:06
360p

SHAKKI





SHAKKI



(NICE TO KNOW: EI TARVITSE OPETELLA)

1769 **Wolfgang von Kempelen** rakentaa “Turkin”

1912 **L. Torres y Quevedo** rakentaa koneen kuningas&torni vs kuningas -loppupeleihin

1948 **Norbert Wiener** esittää syvyysrajoitetun minimax-algoritmin heuristisella arviontifunktiolla

1950 **Claude Shannon** julkaisee artikkelin “Programming a Computer for Playing Chess”

1951 **Alan Turing** kehittää ensimmäisen algoritmin, joka pystyy pelaamaan kokonaisen shakkiottelun

1956 Los Alamos chess: ensimmäinen tietokoneohjelma, joka pelaa (yksinkertaistettua) shakkia

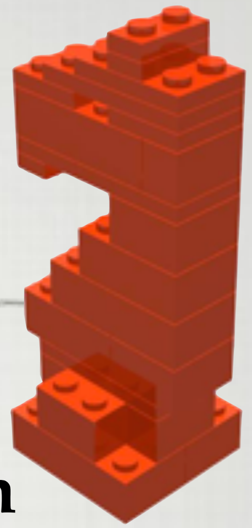
1956 **John McCarthy** keksii alpha-beta-karsinnan

1957 Ensimmäiset oikeaa shakkia pelaavat ohjelmat

1966-67 Ensimmäiset tietokoneohjelmien väliset ottelut (Moskova voittaa.)

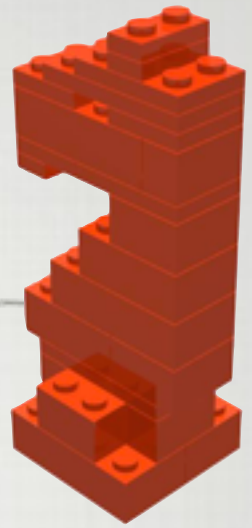
SHAKKI

(NICE TO KNOW: EI TARVITSE OPETELLA)



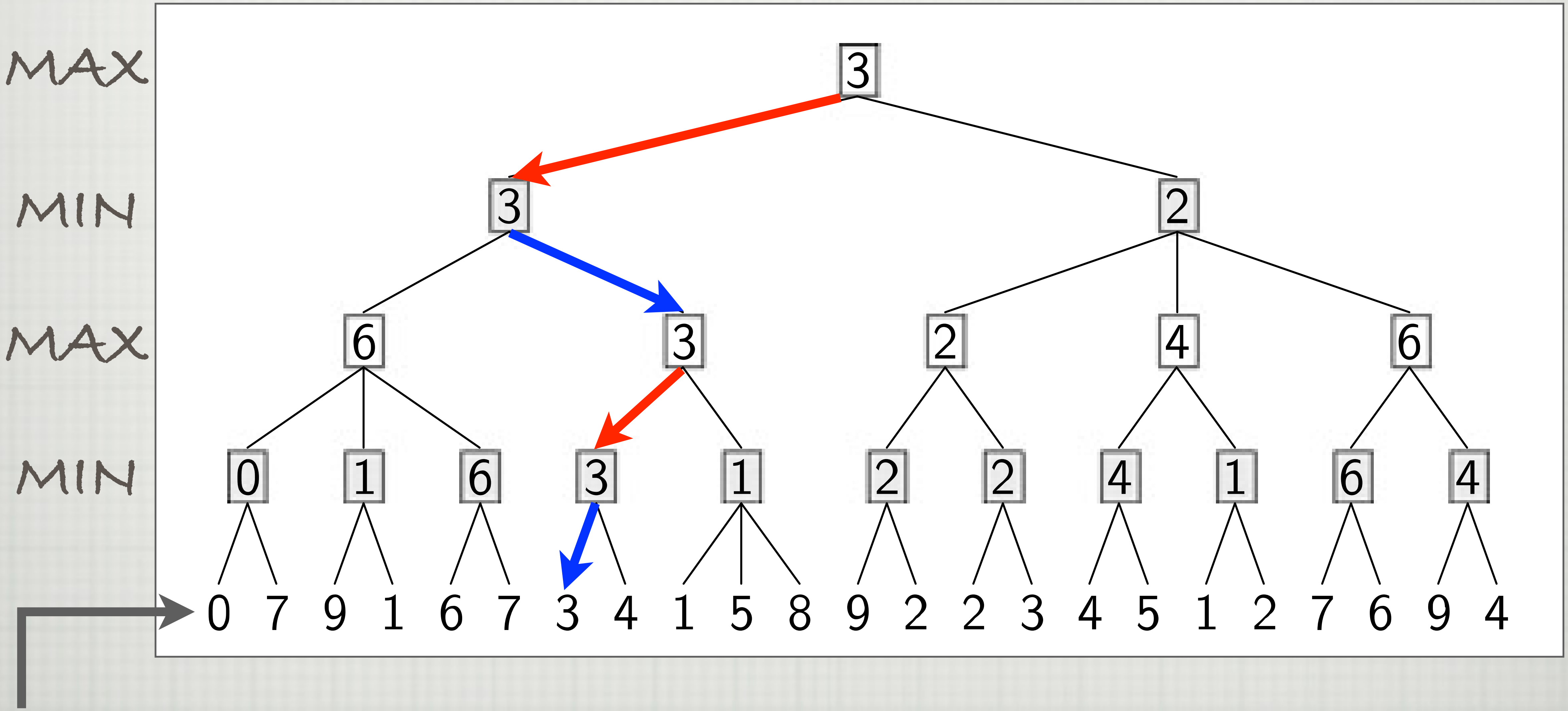
1967 **Ensimmäinen tietokoneohjelman voitto turnauksessa.**
1981 **Cray Blitz voittaa Mississippin osavaltion mestaruuden ja saa ensimmäisenä tietokoneena mestarin statuksen.**
1988 **Deep Thought voittaa ensimmäistä kertaa suurmestarin turnauksessa.**
1989 **Garry Kasparov voittaa kaksi näytösottelua Deep Thoughtia vastaan.**
1996 **Garry Kasparov voittaa Deep Bluen kuuden pelin ottelussa.**
1997 **Deep Blue voittaa Garry Kasparovin kuuden pelin ottelussa.**
2006 **Deep Fritz voittaa maailmanmestari Vladimir Kramnikin.**

SHAKKI



- TILA: (LAUDAN TILANNE)
- SIIRTYMÄT: (SALLITUT SIIRROT)
- MENETELMÄ: SYVYYSRAJOITETTU ALPHA-BETA-KARSINTA

PELIPUU



ARVIOITA TILANTEEN HYVYYDESTÄ

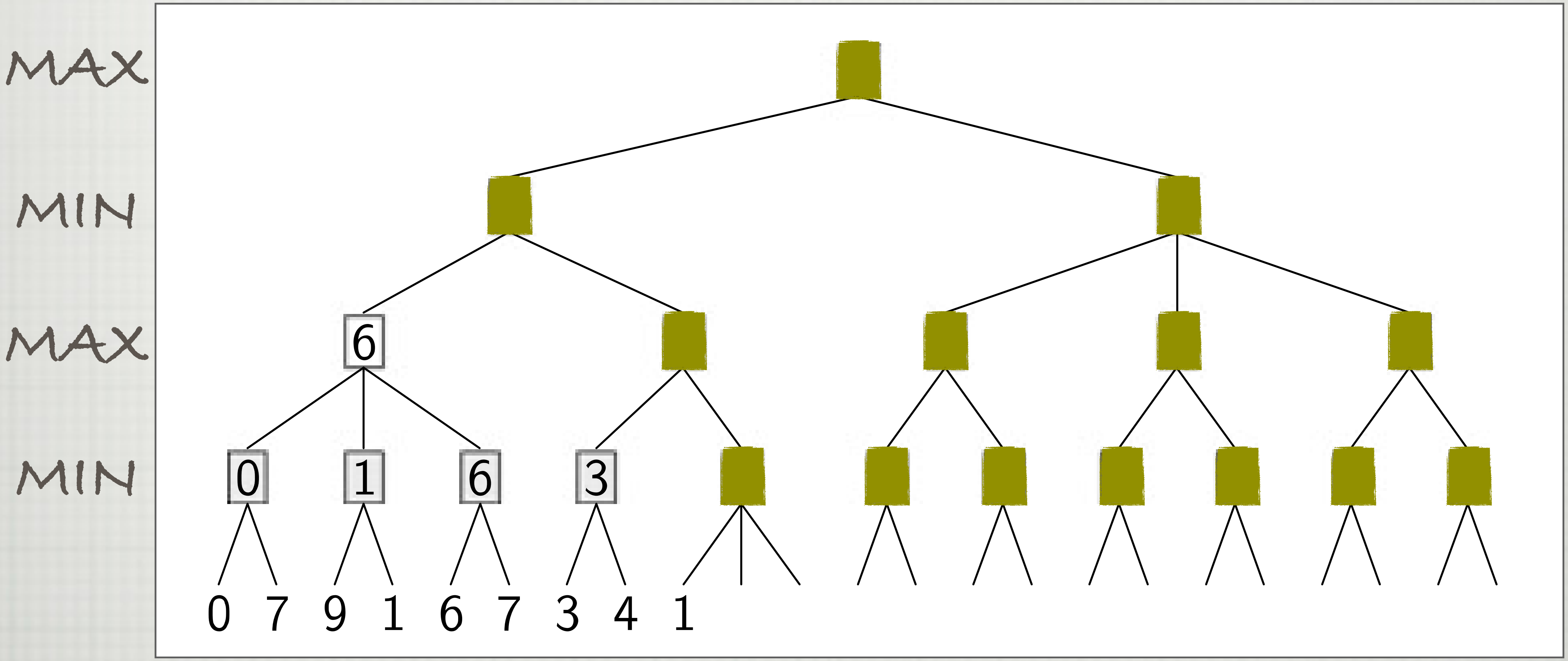
SHAKKI

- TILA: (LAUDAN TILANNE)
- SIIRTYMÄT: (SALLITUT SIIRROT)
- MENETELMÄ: SYVYYSRAJOITETTU ALPHA-BETA-KARSINTA
- TEHTÄVÄ: SUUNNITTELE HEURISTINEN ARVIOINTIFUNKTIO

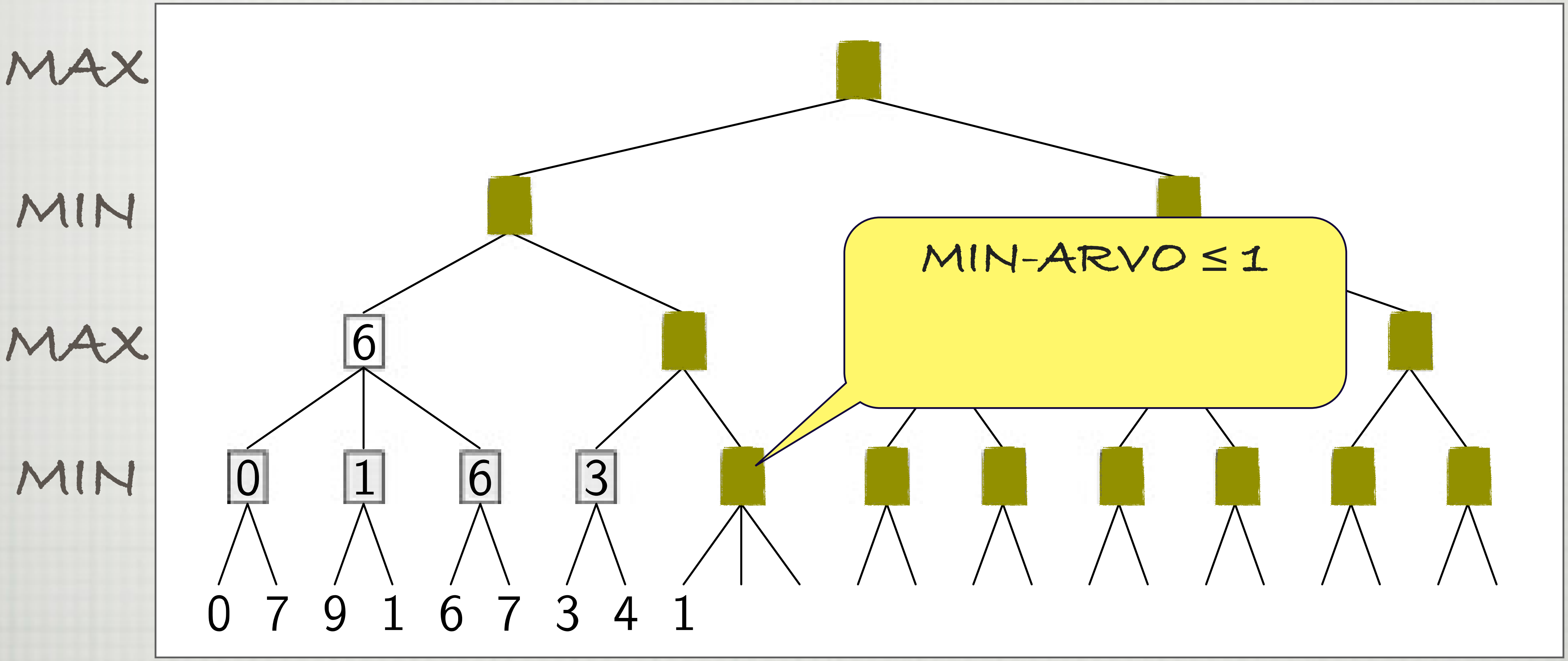
HEURISTIIKKOJEN VALINNASTA

- HEURISTIIKAN HYVYYS VAIKUTTAA PELIN TULOKSEEN: HYVÄ HEURISTIIKKA -> HYVÄ TULOS
- VASTAAVASTI HEURISTIIKAN HYVYYTTÄ VOI MITATA TARKKAILEMALLA PELIEN TULOKSIA: HYVÄ TULOS -> HYVÄ HEURISTIIKKA
- JOSKUS HYVÄKIN PELAAJA VOI SILTI HÄVITÄ HUONOMMALLEEN JA TOISINPÄIN, JOTEN ARVIOINTI EI OLE HELPPOA
- YLEINEN MENETELMÄ HYVYYDEN ARVIONTIIN: ELO-RATING

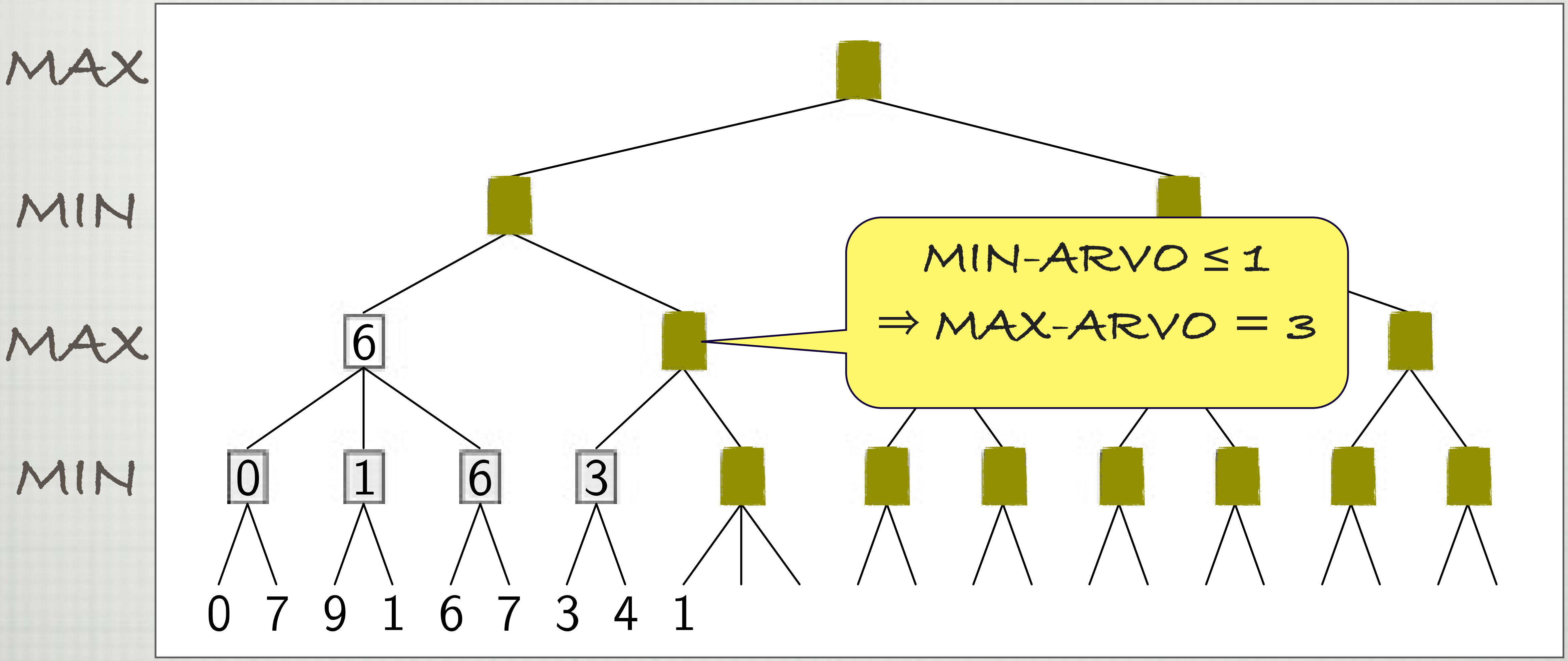
ALPHA-BETA-KARSINTA



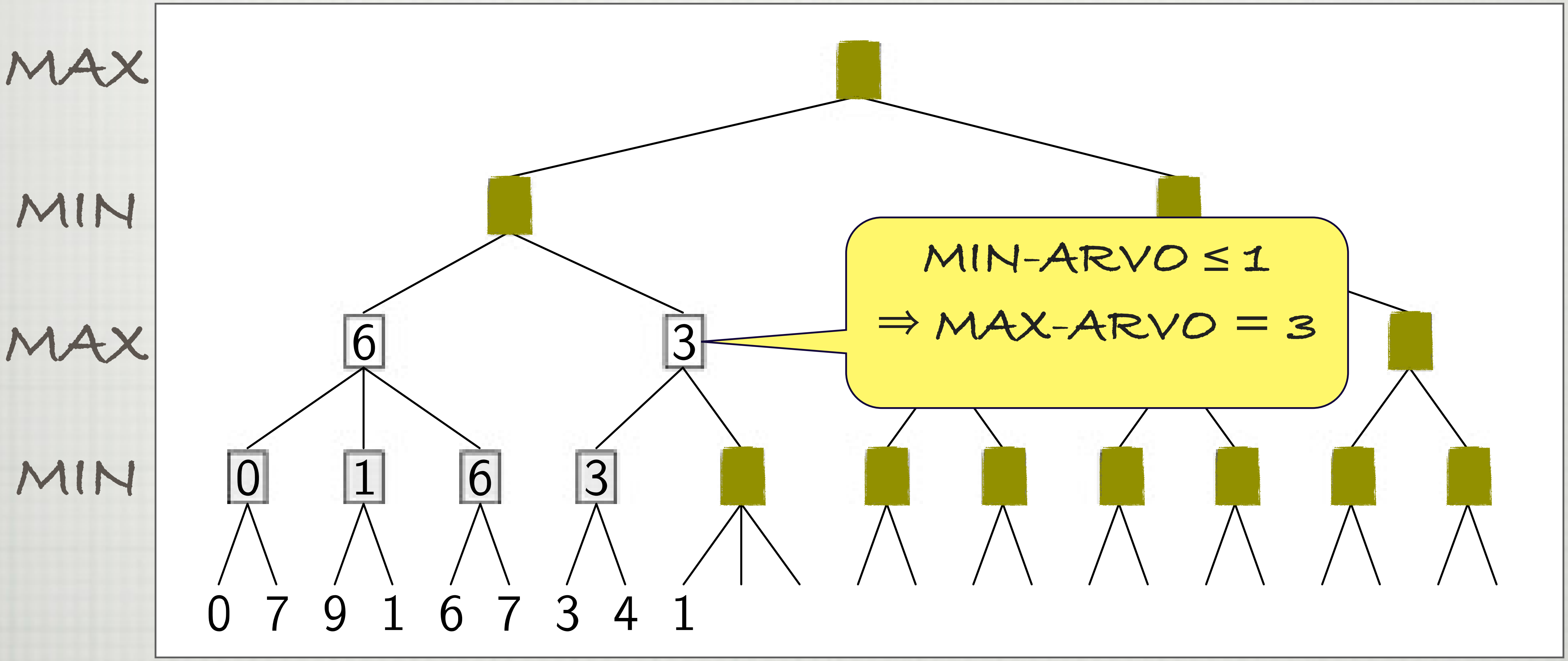
ALPHA-BETA-KARSINTA



ALPHA-BETA-KARSINTA



ALPHA-BETA-KARSINTA



ALPHA-BETA-KARSINTA

MAX-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = $-\infty$
for each Lapsi in LAPSET(Solmu)
    v = MAX(v, MIN-ARVO(Lapsi, α, β))
    if v $\geq \beta$ **return**(v)
    α = MAX(α, v)
return(v)

MIN-PELAAJAN TOISTAISEKSI PARAS ARVO

MAX-PELAAJAN TOISTAISEKSI PARAS ARVO

ALPHA-BETA-KARSINTA

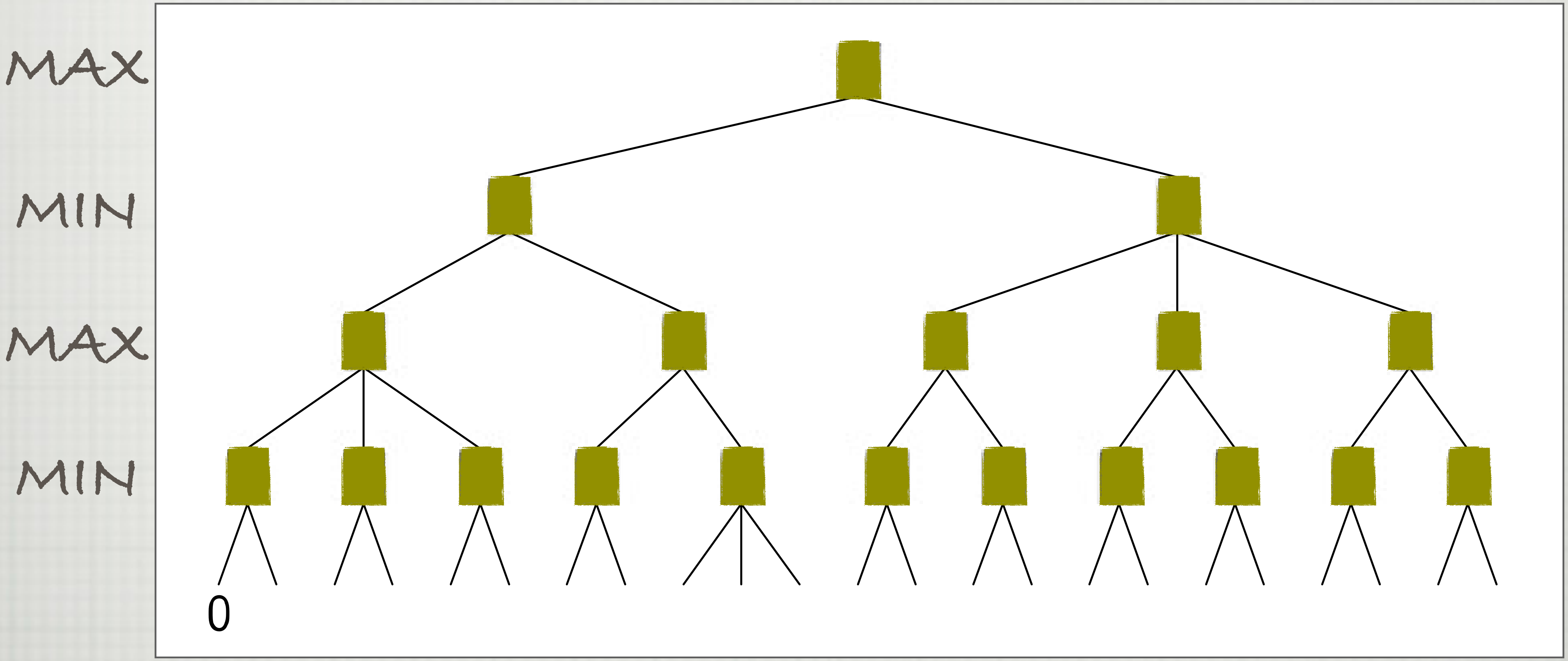
MAX-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = $-\infty$
for each Lapsi in LAPSET(Solmu)
  v = MAX(v, MIN-ARVO(Lapsi, α, β))
  if v $\geq \beta$ **return**(v)
  α = MAX(α, v)
return(v)

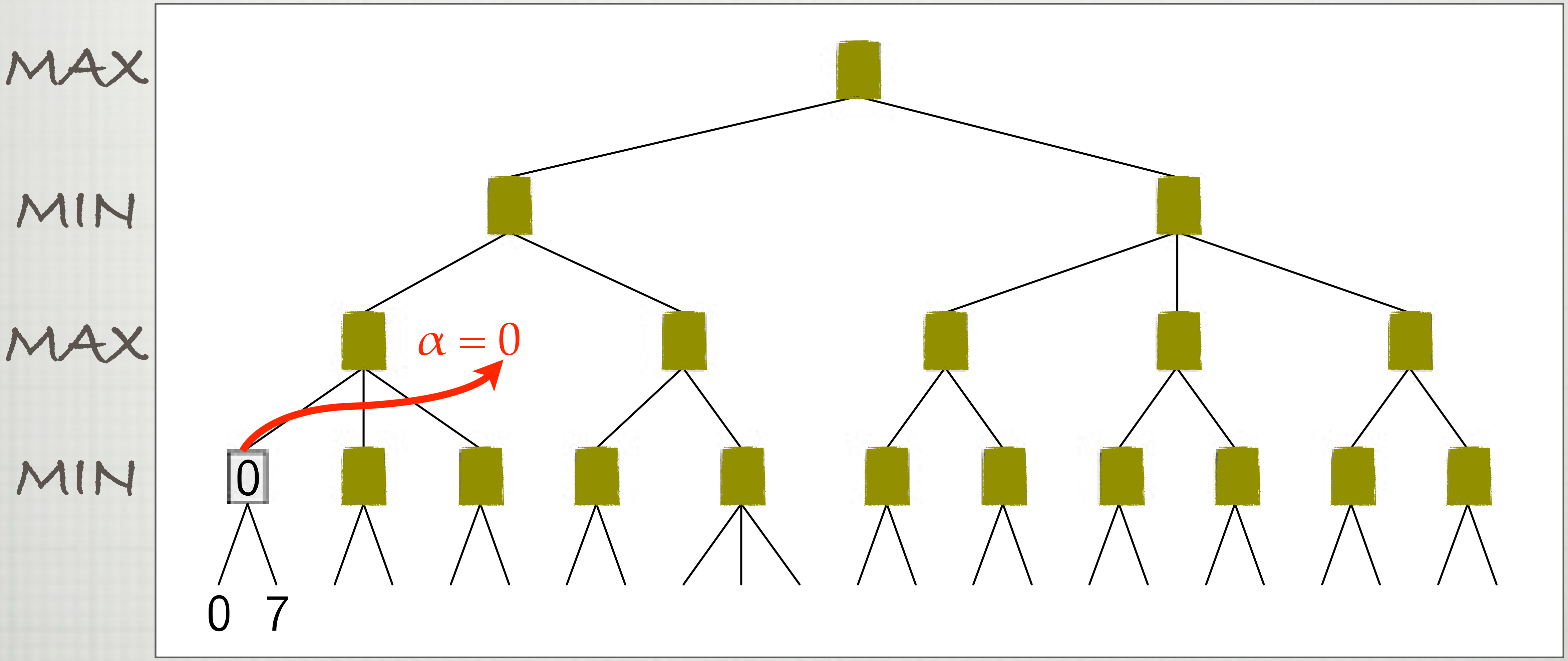
MIN-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = $+\infty$
for each Lapsi in LAPSET(Solmu)
  v = MIN(v, MAX-ARVO(Lapsi, α, β))
  if v $\leq \alpha$ **return**(v)
  β = MIN(β, v)
return(v)

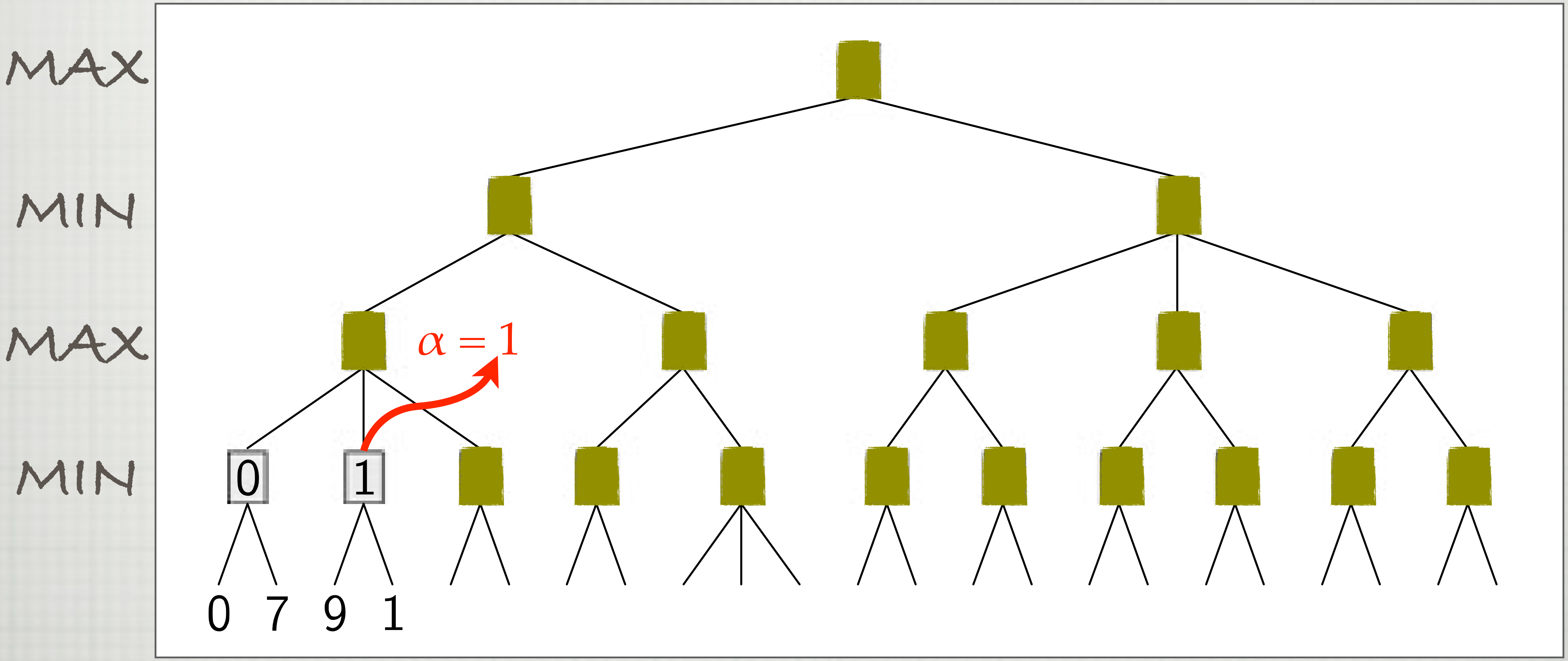
ALPHA-BETA-KARSINTA



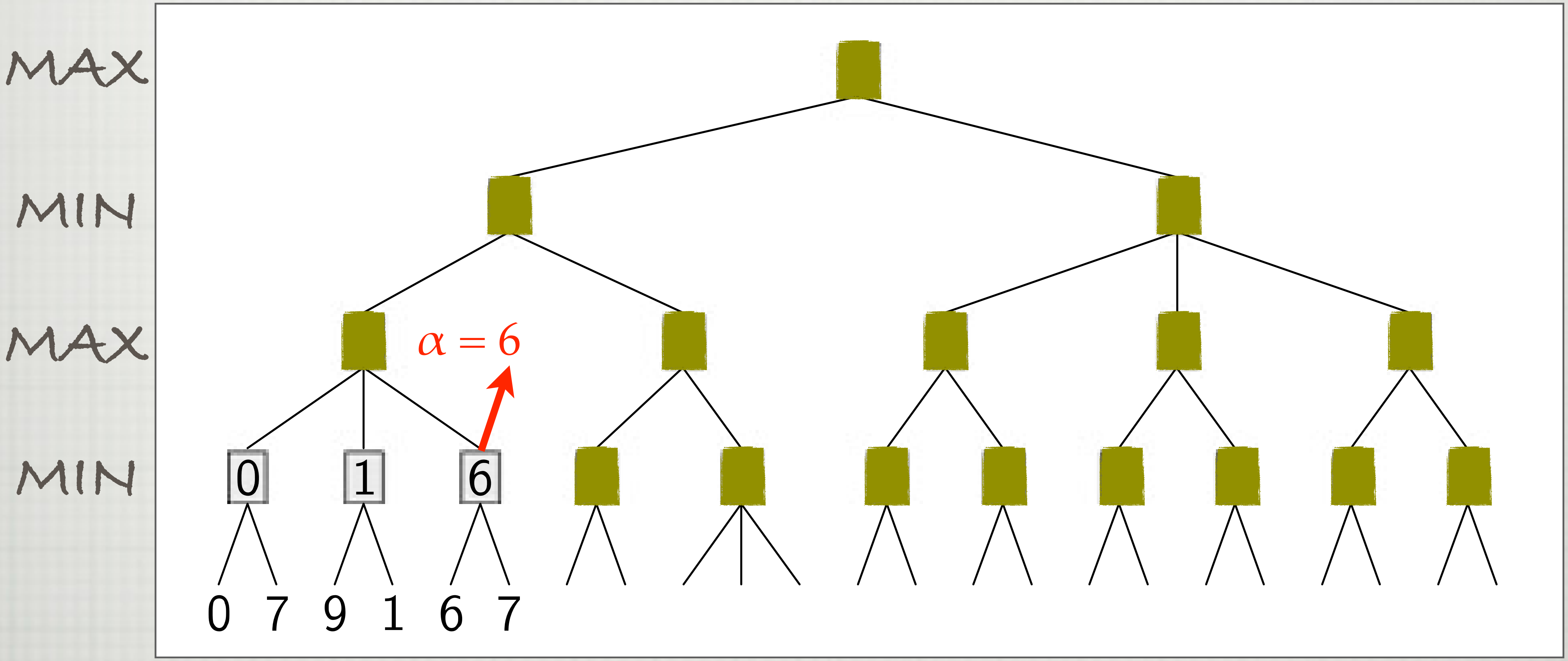
ALPHA-BETA-KARSINTA



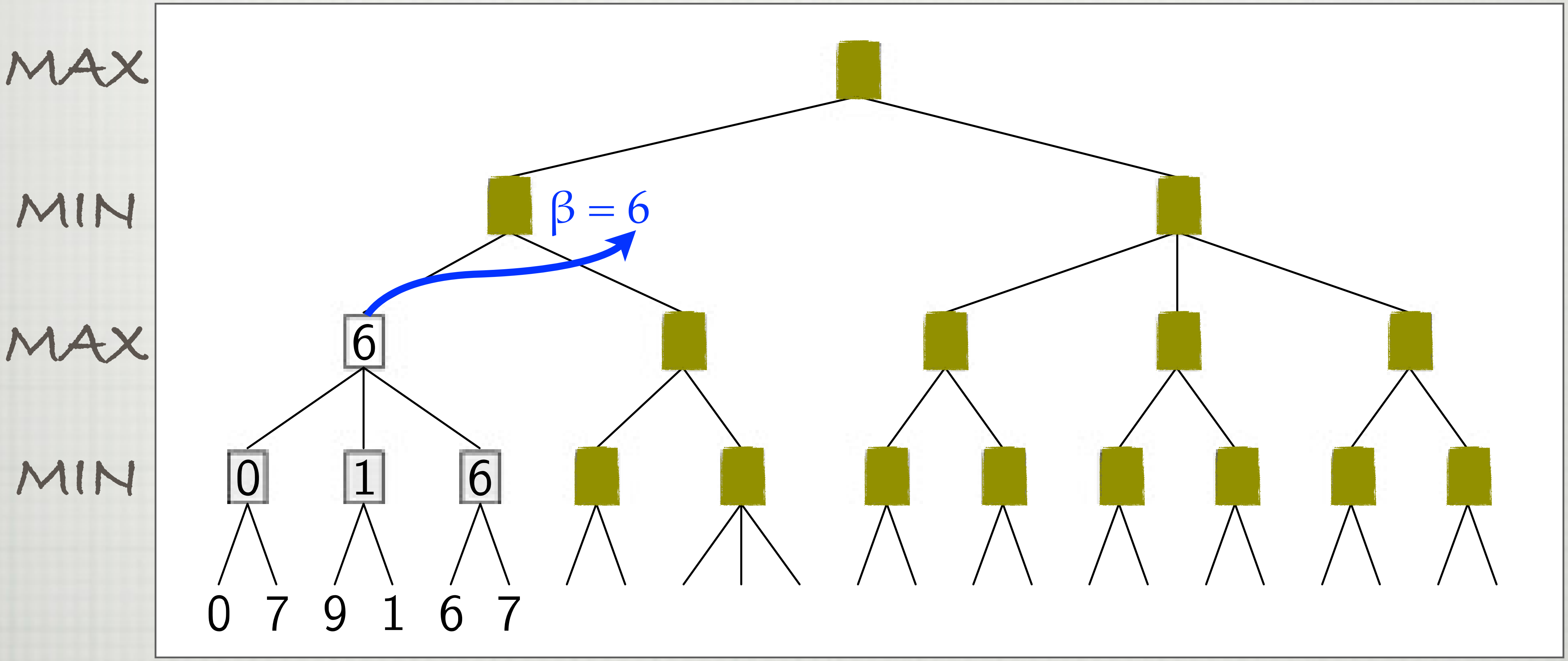
ALPHA-BETA-KARSINTA



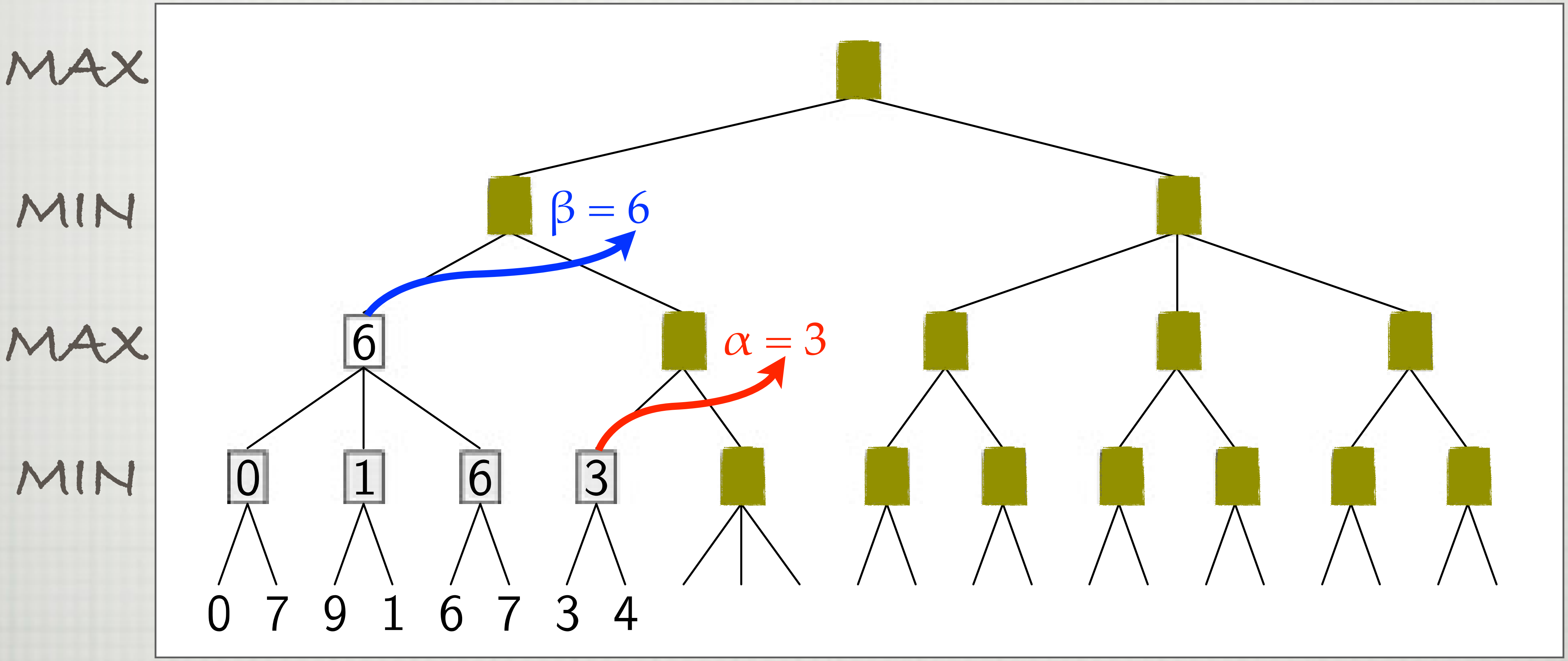
ALPHA-BETA-KARSINTA



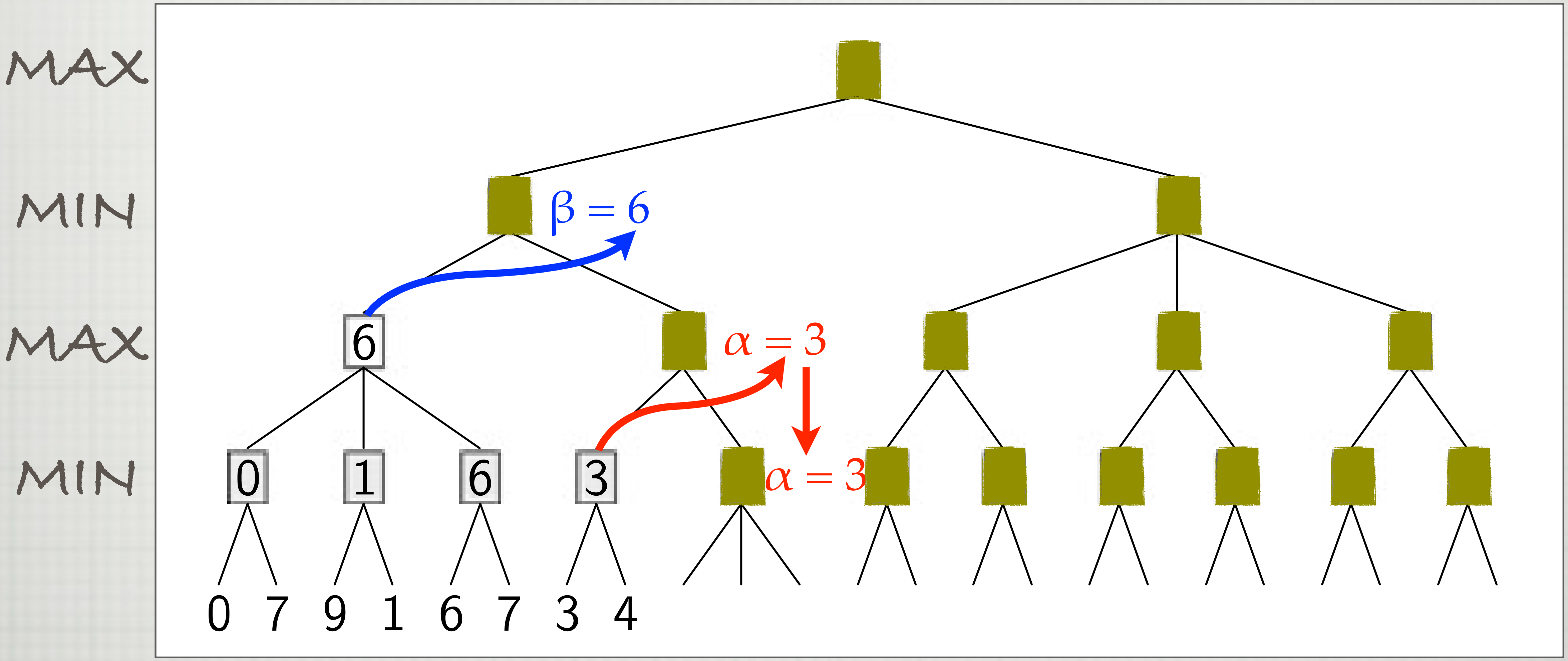
ALPHA-BETA-KARSINTA



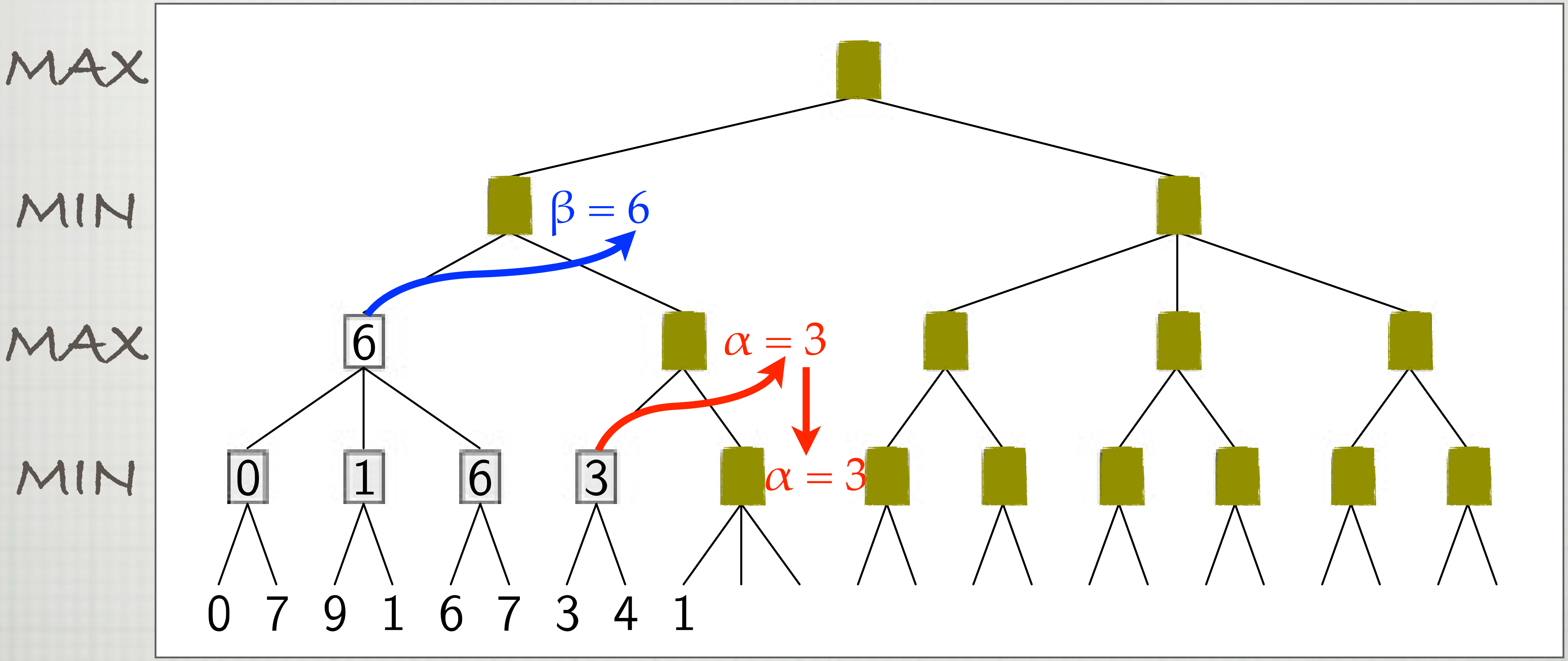
ALPHA-BETA-KARSINTA



ALPHA-BETA-KARSINTA

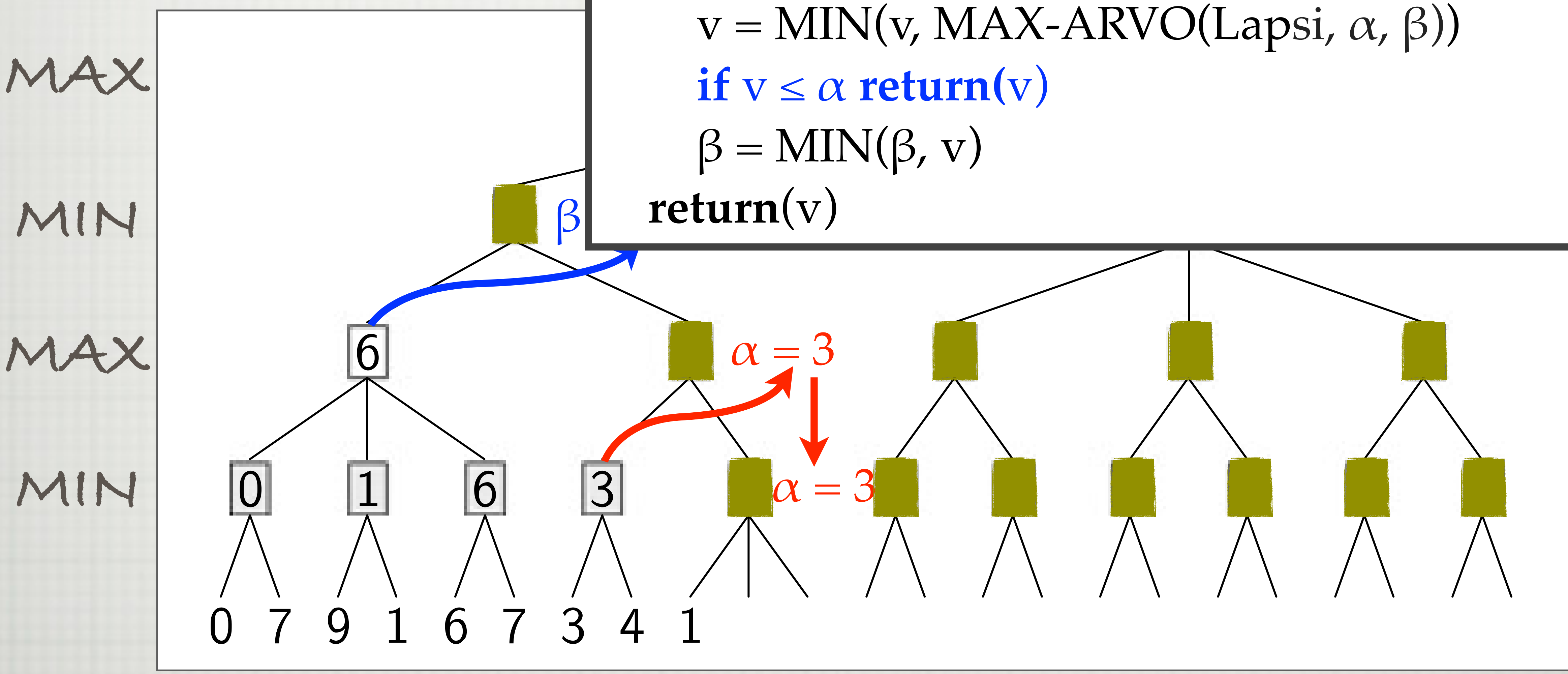


ALPHA-BETA-KARSINTA



ALPHA

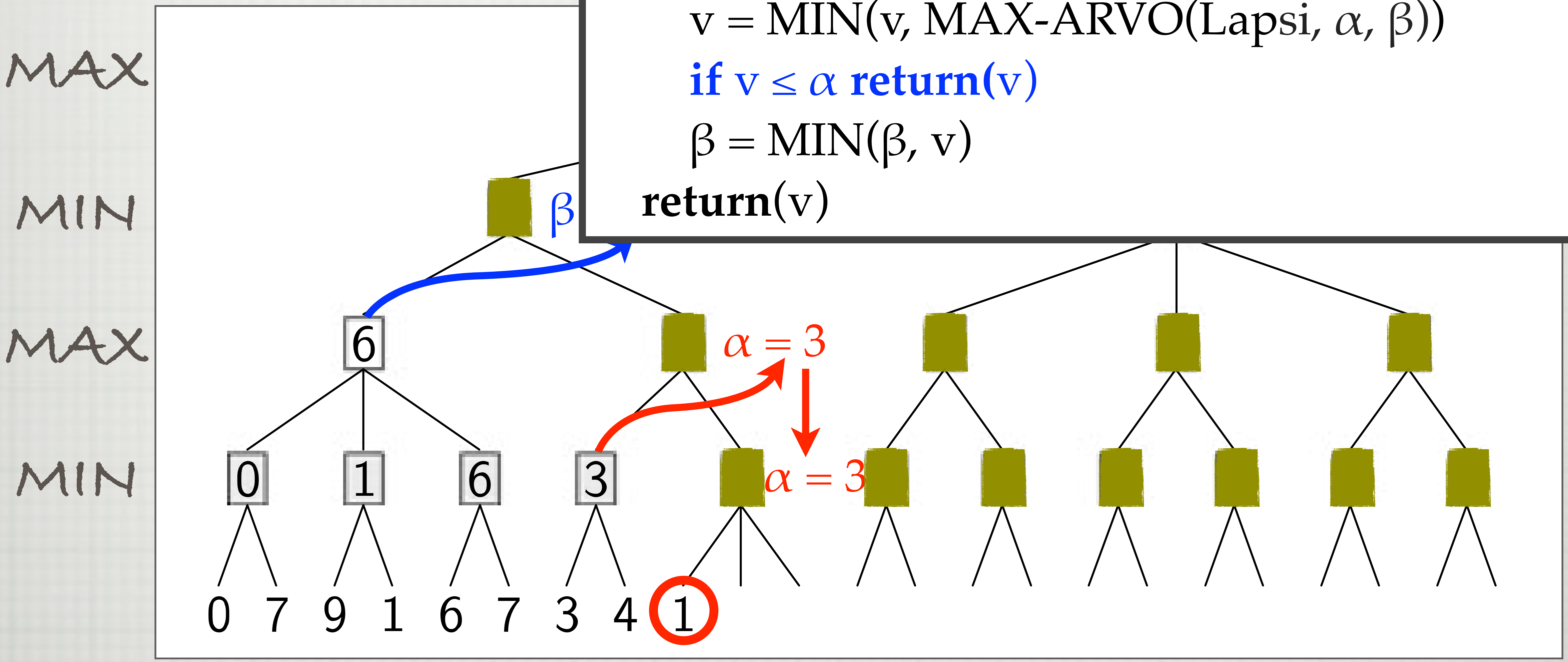
```
MIN-ARVO(Solmu, α, β)

    if LOPPUTILA(Solmu) return(ARVO(Solmu))
    v = +∞
    for each Lapsi in LAPSET(Solmu)
        v = MIN(v, MAX-ARVO(Lapsi, α, β))
        if v ≤ α return(v)
        β = MIN(β, v)
    return(v)
```



ALPHA

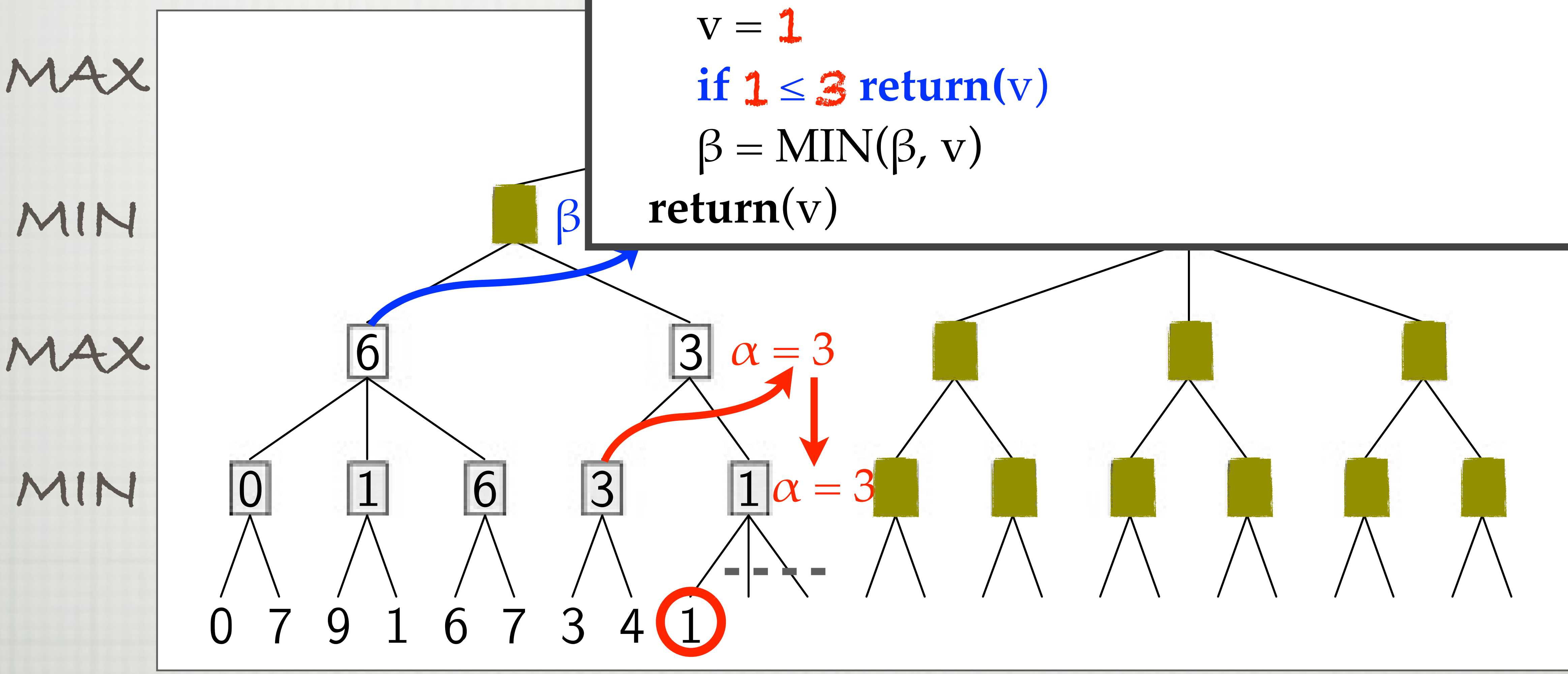
MIN-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = +∞
for each Lapsi in LAPSET(Solmu)
  v = MIN(v, MAX-ARVO(Lapsi, α, β))
  if v ≤ α **return**(v)
  β = MIN(β, v)
return(v)

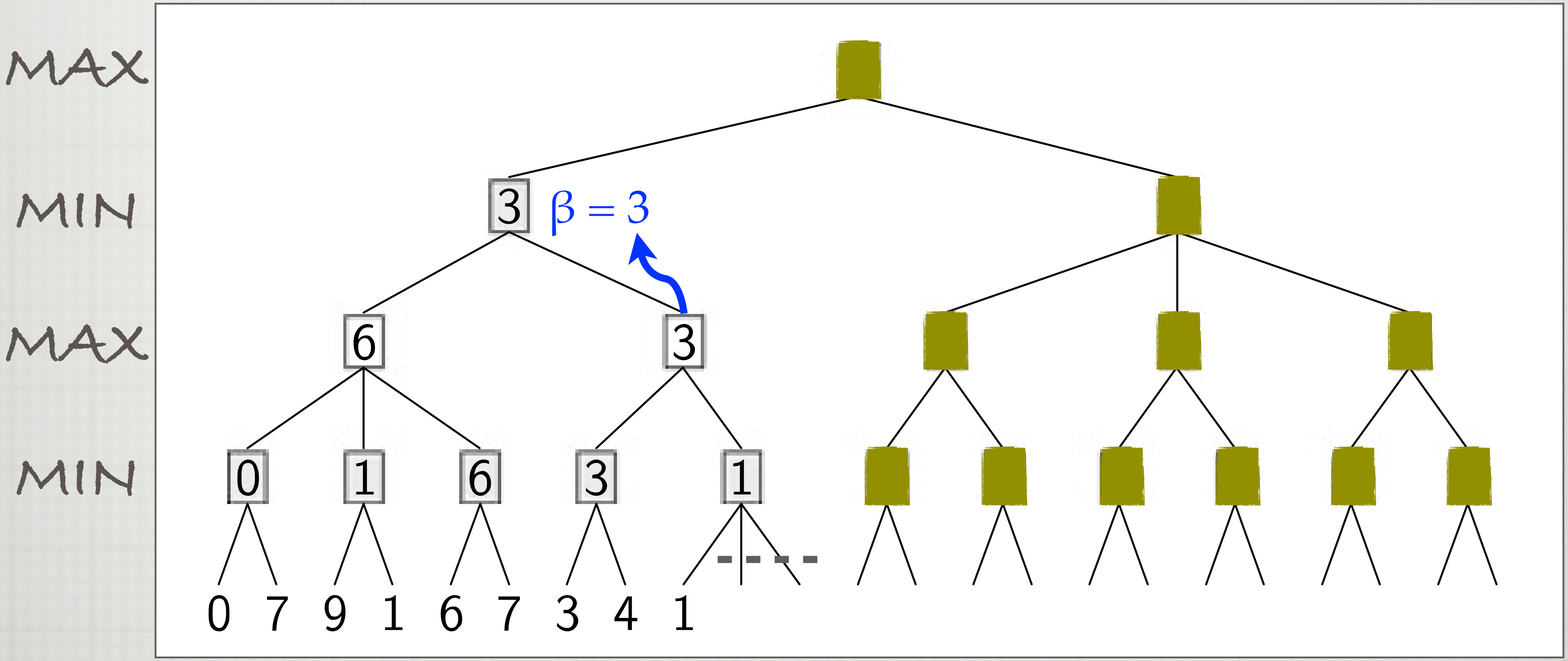


ALPHA

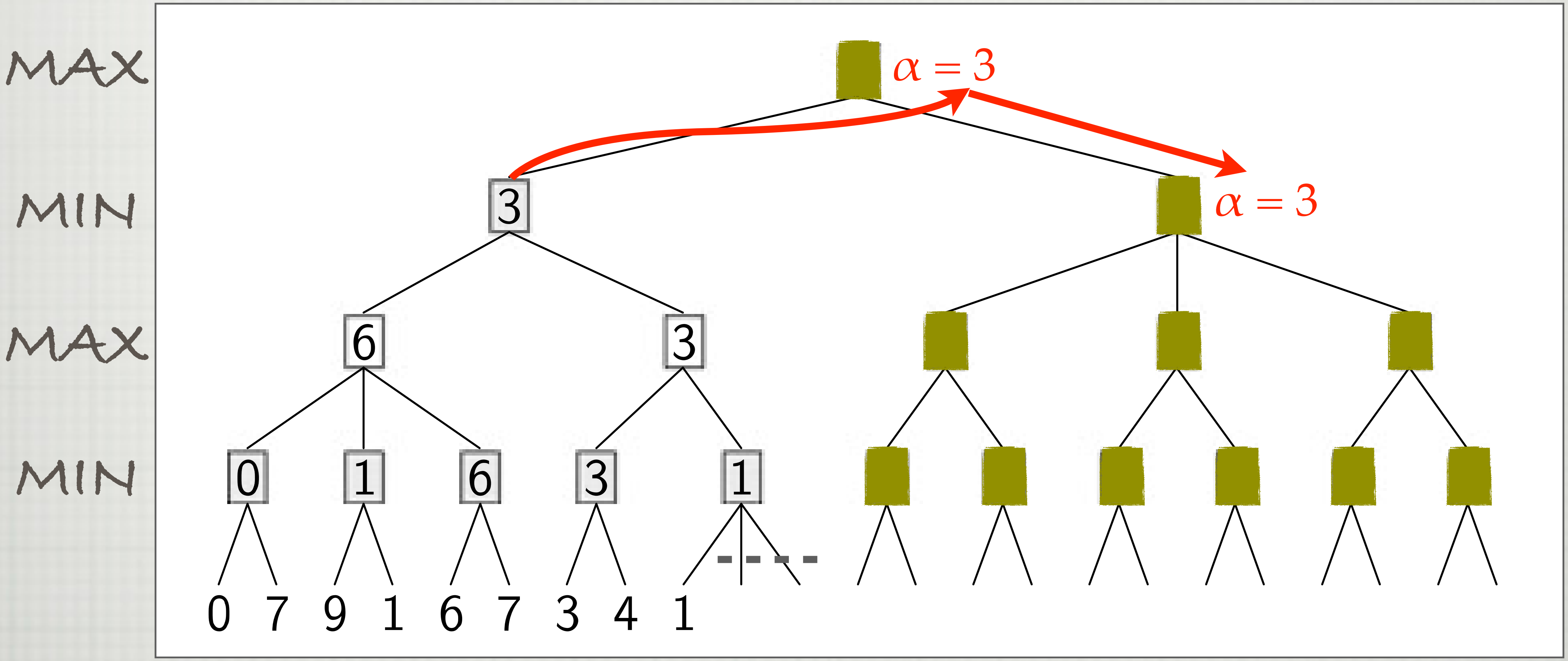
```
MIN-ARVO(Solmu, α, β)

    if LOPPUTILA(Solmu) return(ARVO(Solmu))
    v = +∞
    for each Lapsi in LAPSET(Solmu)
        v = 1
        if 1 ≤ 3 return(v)
        β = MIN(β, v)
    return(v)
```



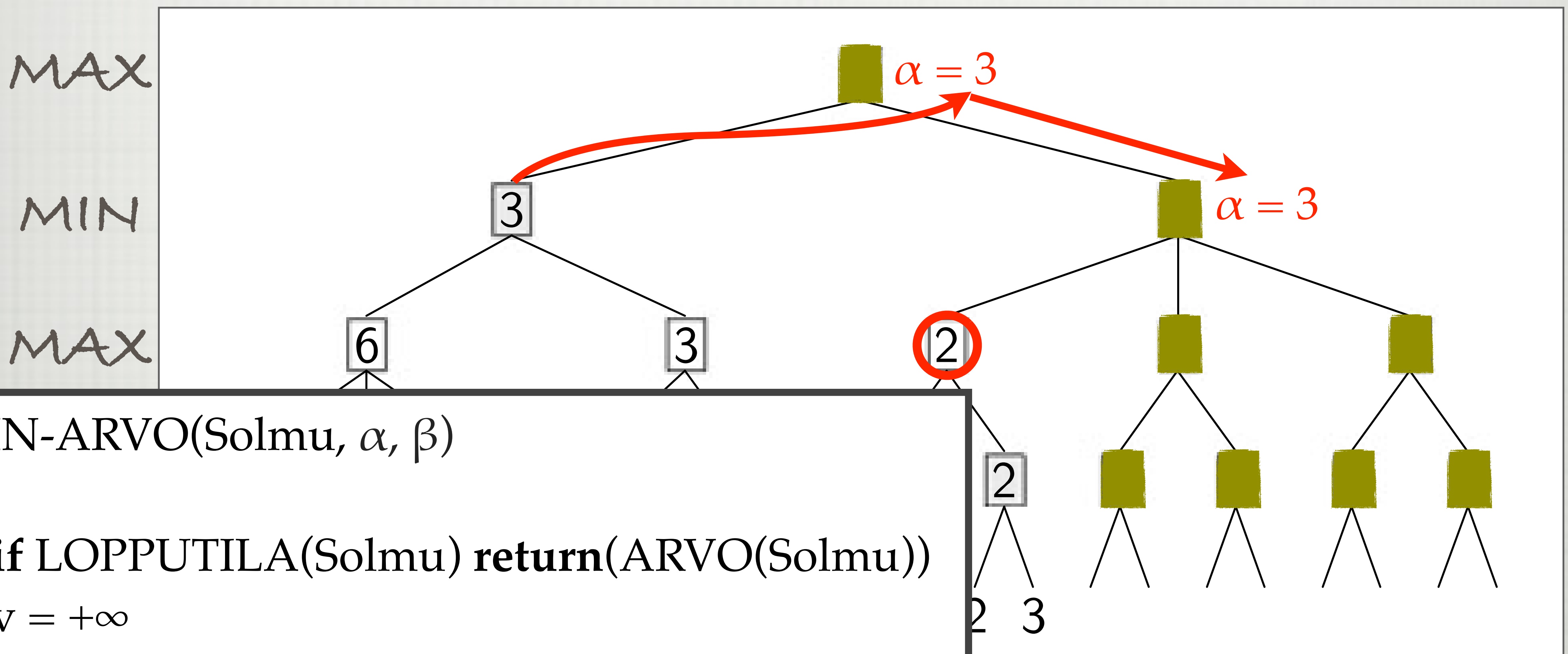
ALPHA-BETA-KARSINTA



ALPHA-BETA-KARSINTA



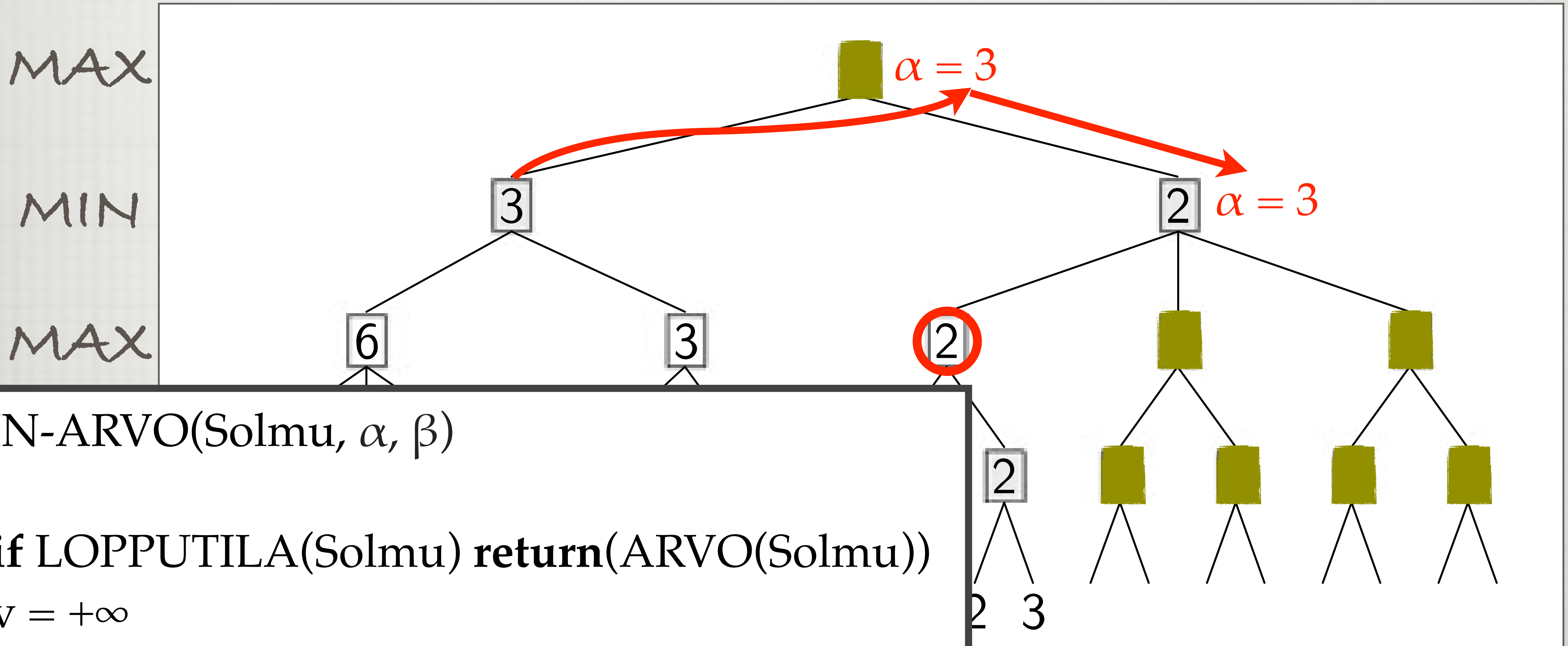
ALPHA-BETA-KARSINTA



MIN-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = +∞
for each Lapsi in LAPSET(Solmu)
  v = 2
  if v ≤ α **return**(v)
  β = MIN(β, v)
return(v)

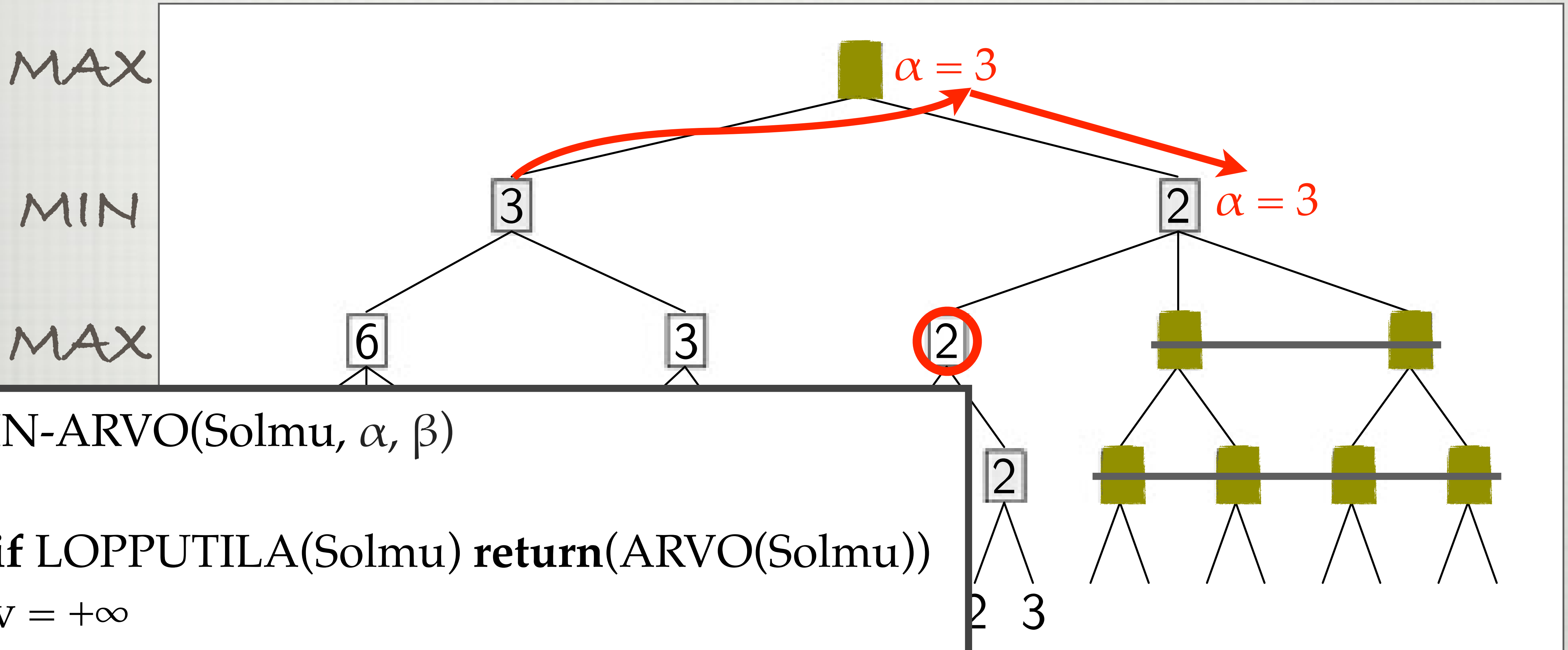
ALPHA-BETA-KARSINTA



MIN-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = +∞
for each Lapsi in LAPSET(Solmu)
v = 2
if 2 ≤ 3 **return**(v)
β = MIN(β, v)
return(v)

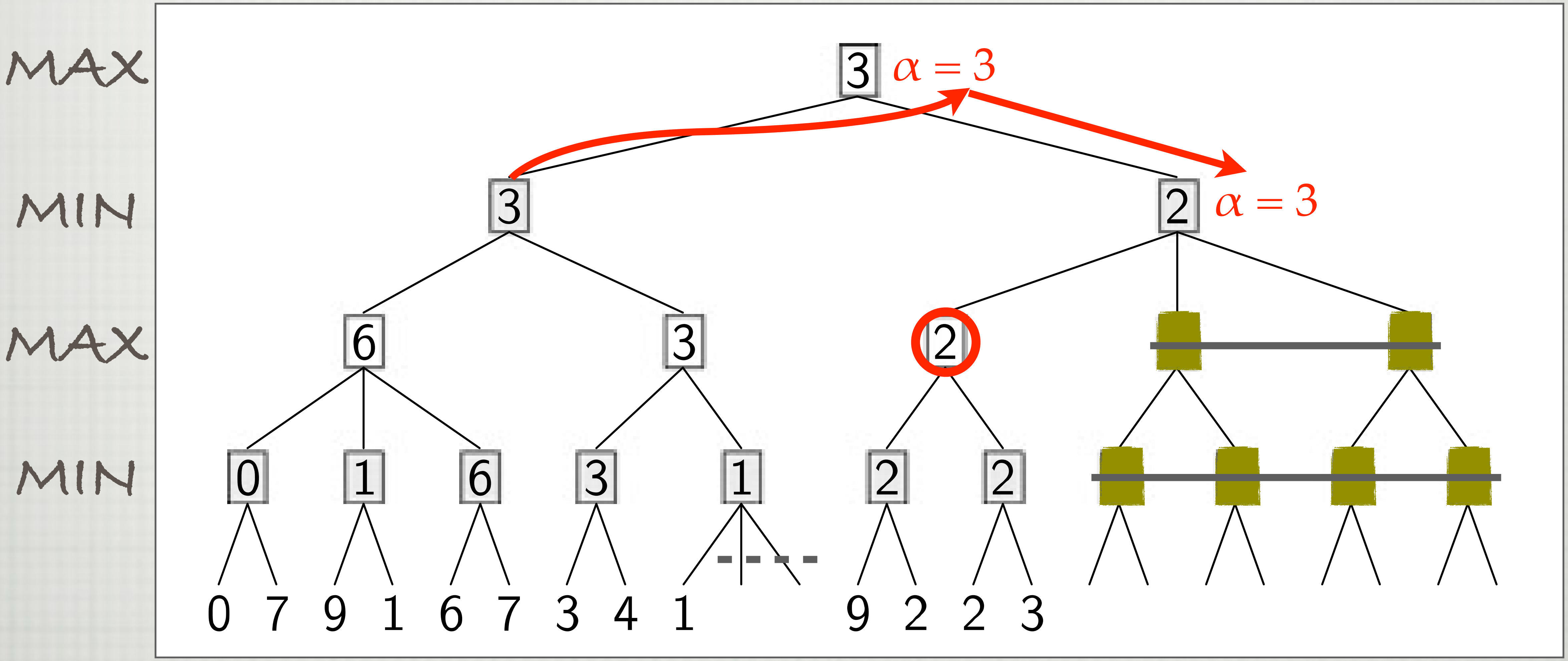
ALPHA-BETA-KARSINTA



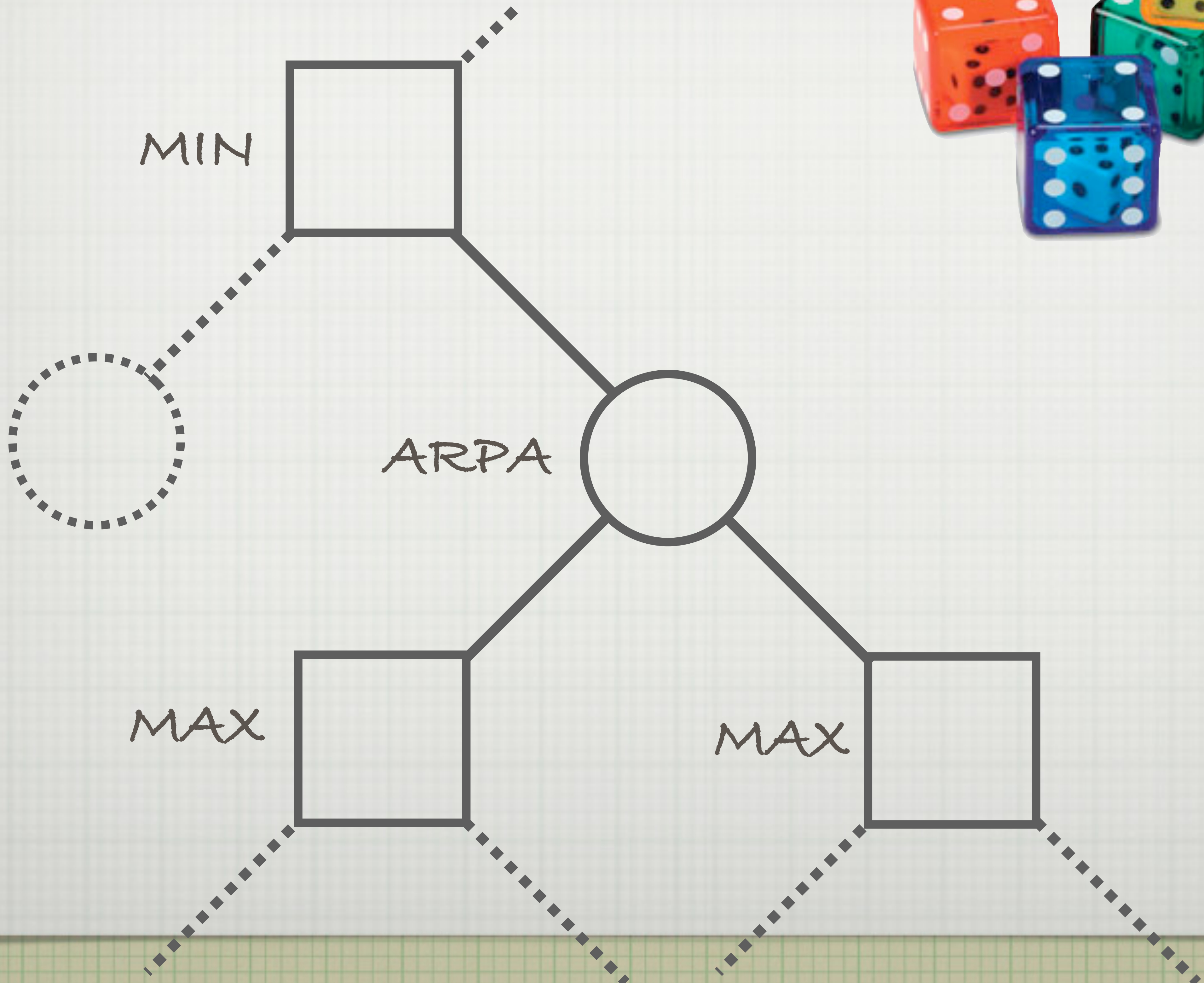
MIN-ARVO(Solmu, α, β)

if LOPPUTILA(Solmu) **return**(ARVO(Solmu))
v = +∞
for each Lapsi in LAPSET(Solmu)
v = 2
if 2 ≤ 3 **return**(v)
β = MIN(β, v)
return(v)

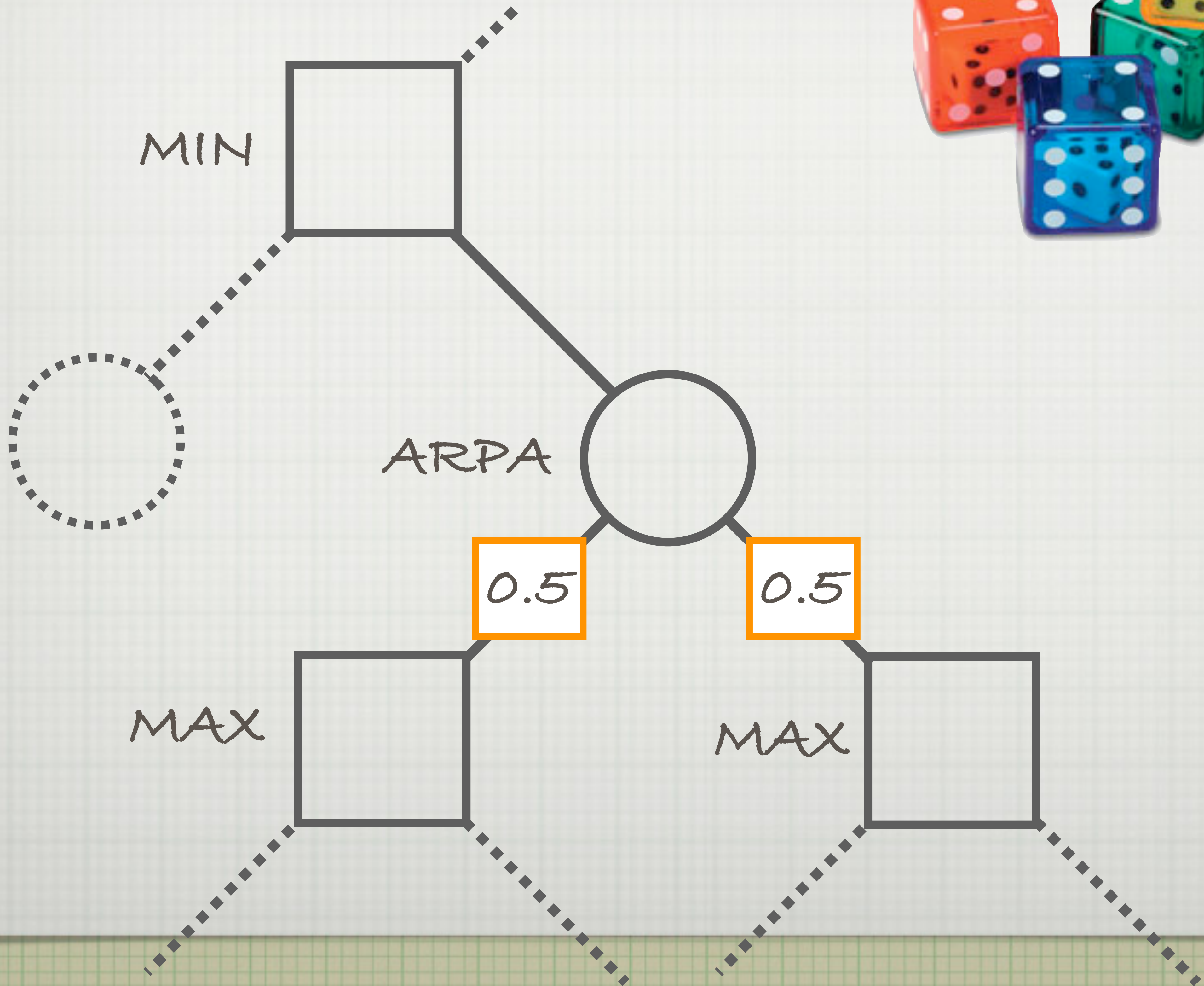
ALPHA-BETA-KARSINTA



ARPAPELIT



ARPAPELIT



ENSI VIIKOLLA

* LOGIIKASTA (TEKOÄLYN HISTORIAA)
* TODENNÄKÖISYYS