GAMES

CS221: Section 5

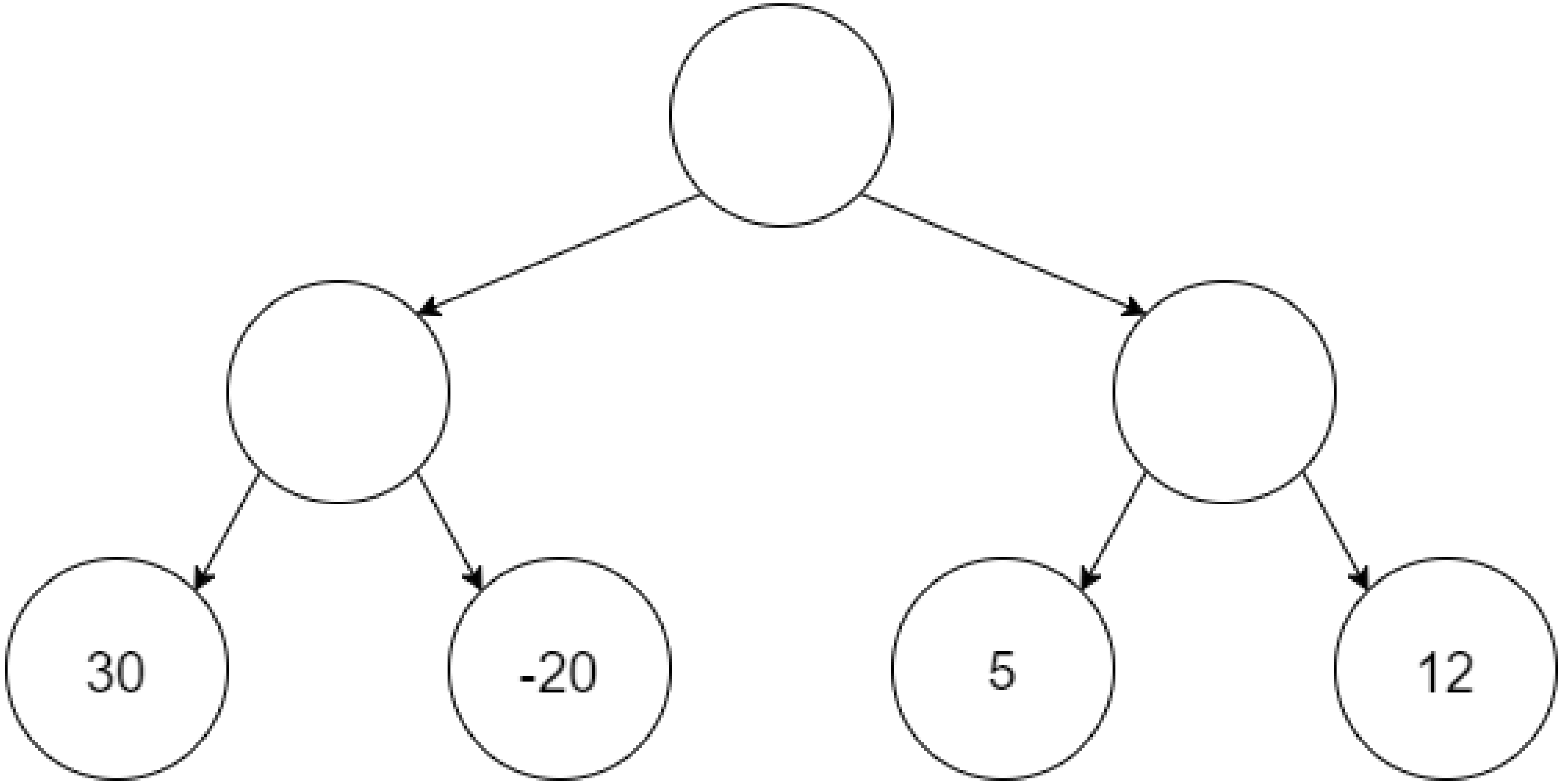
Today's agenda

- Game Trees
- Expectimax
- Minimax
- TD Learning

- Game Trees
- Expectimax
- Minimax
- TD Learning

Key idea: game tree

Each node is a decision point for a player.
Each root-to-leaf path is a possible outcome of the game.



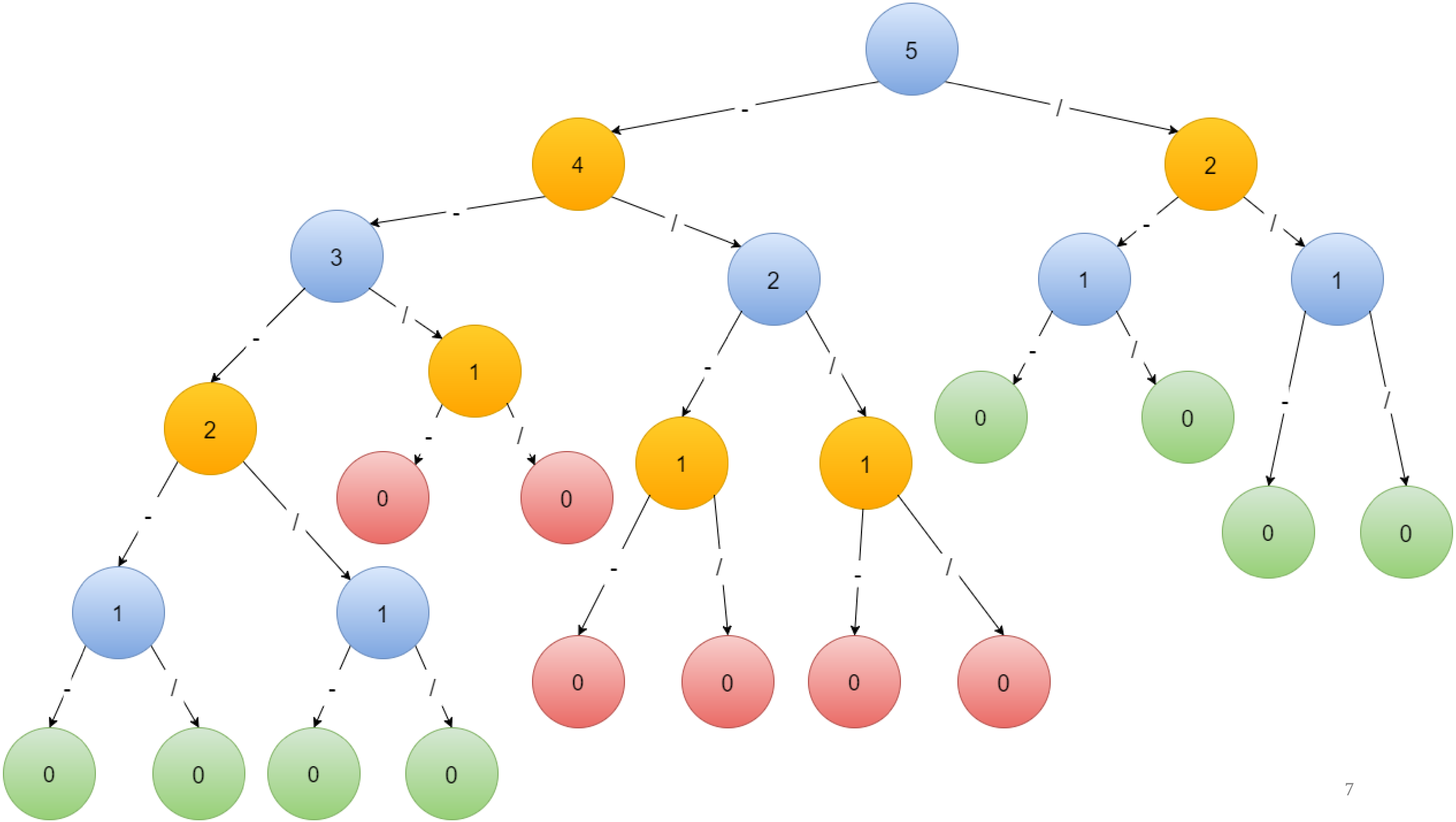
Problem Statement

- Start with a number N
- Players take turns either decrementing N or replacing it with $\left\lfloor \frac{N}{2} \right\rfloor$
- The person to first reach 0 is the winner

Problem Statement

- Start with a number N
- Players take turns either decrementing N or replacing it with $\left\lfloor \frac{N}{2} \right\rfloor$
- The person to first reach 0 is the winner

Lets say N=5


5
4
2
3
2
1
1
2
1
1
1
0
0
0
0
0
0
1
1
0
0
1
1
0
0
0
0
0
0
0
0
0

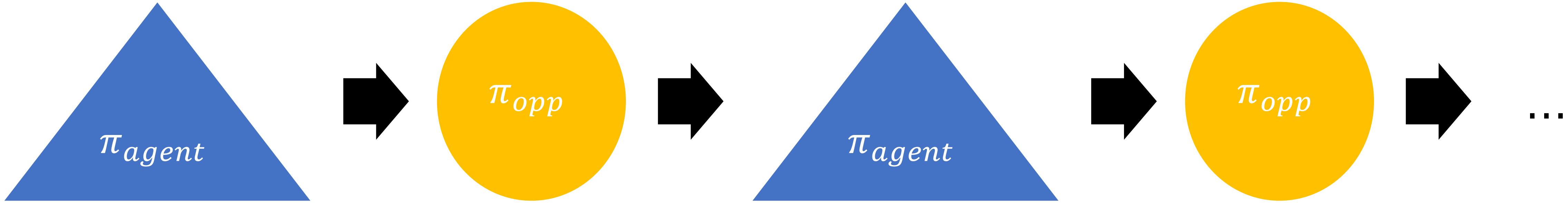
- Game Trees
- Expectimax
- Minimax
- TD Learning

Expectimax

The agent chooses the policy that is

optimal with respect to a fixed known policy

Expectimax Recurrence



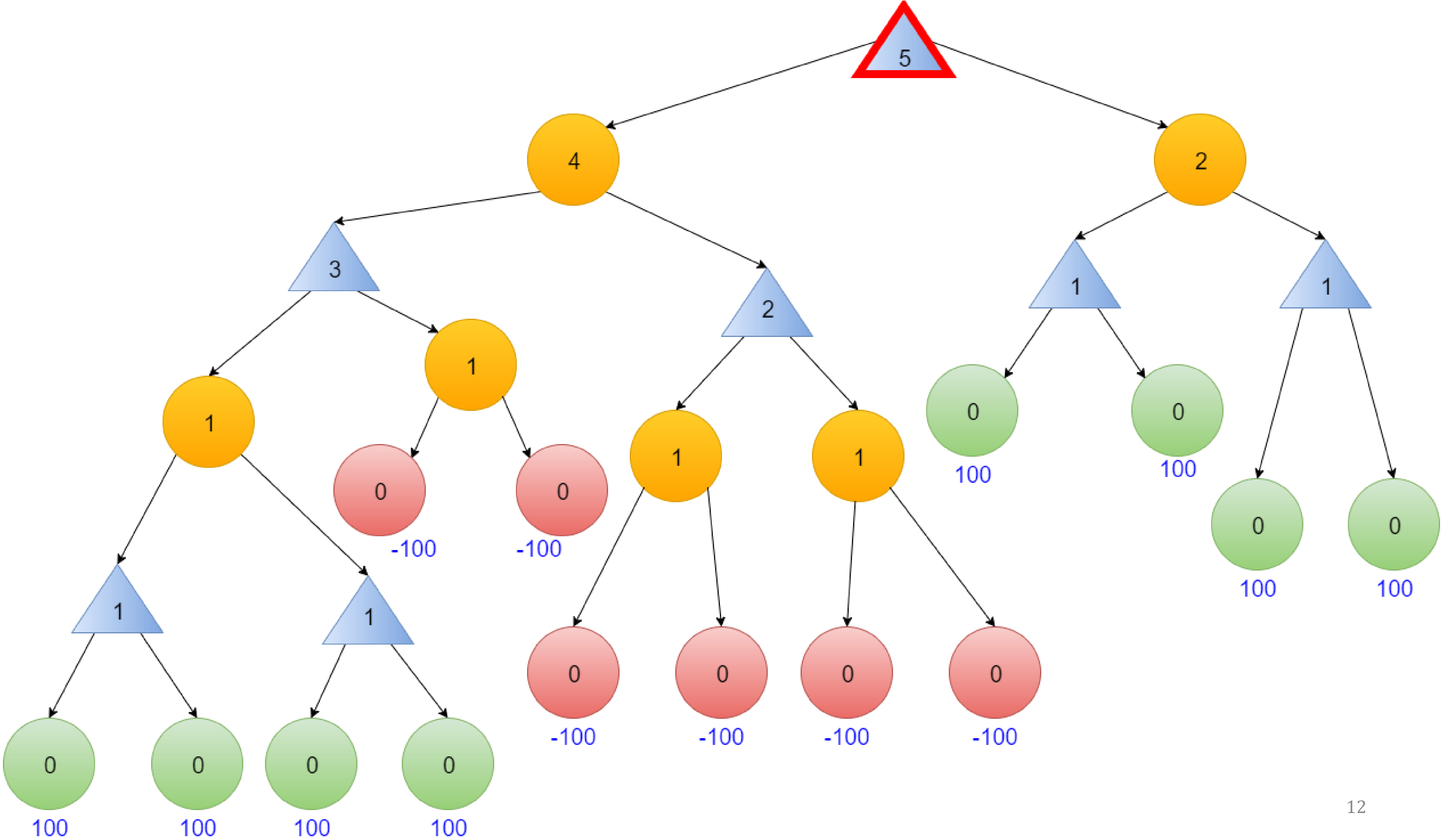
$$V_{opt,\pi}(s) = \begin{cases} \text{Utility}(s) & \text{IsEnd}(s) \\ \max_{a \in \text{Actions}(s)} V_{opt,\pi}(Succ(s,a)) & \text{Player}(s) = \text{agent} \\ \sum_{a \in \text{Actions}(s)} \pi_{opp}(s,a) V_{opt,\pi}(Succ(s,a)) & \text{Player}(s) = \text{opp} \end{cases}$$

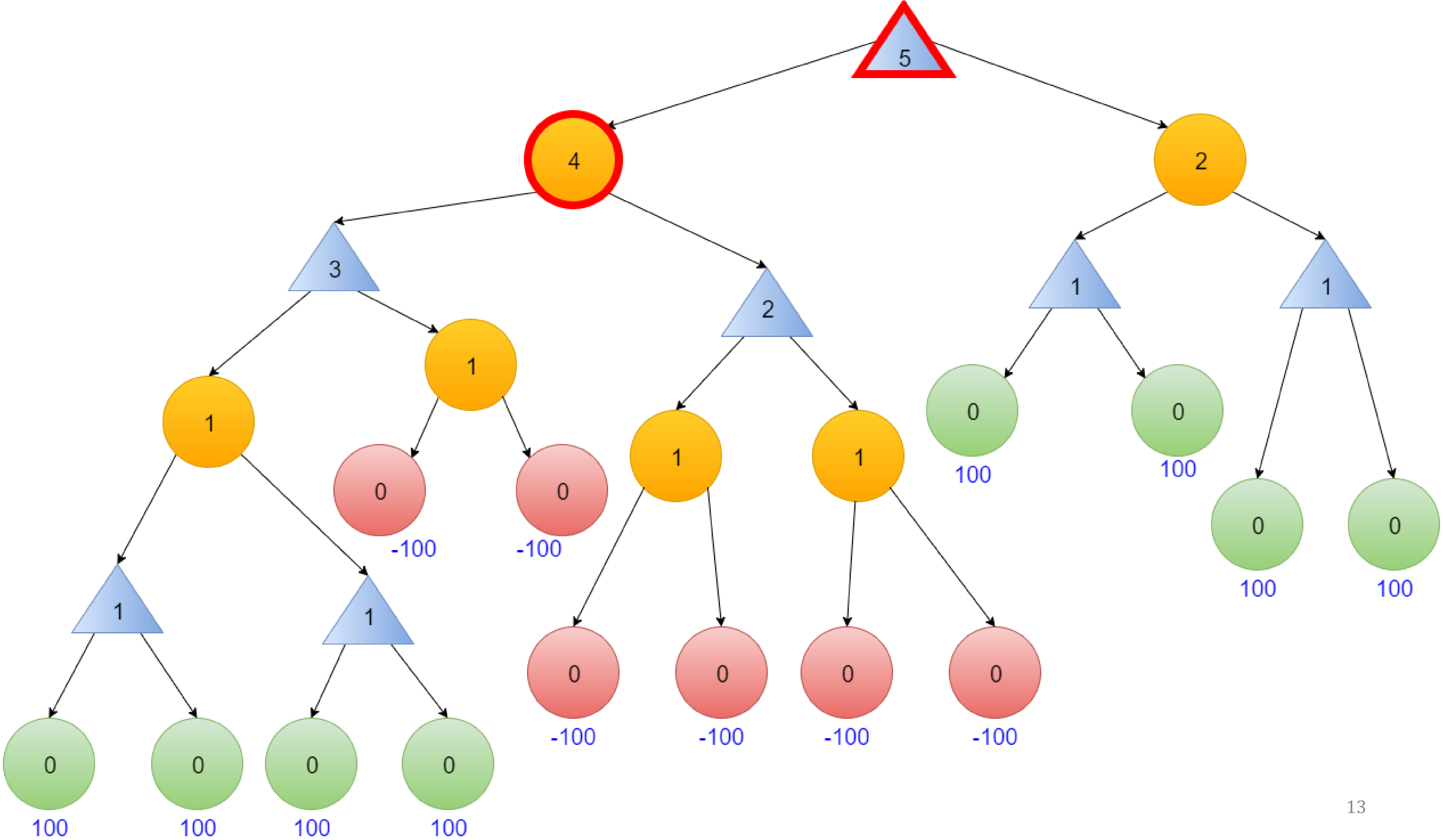
For our example, let's assume that the opponent

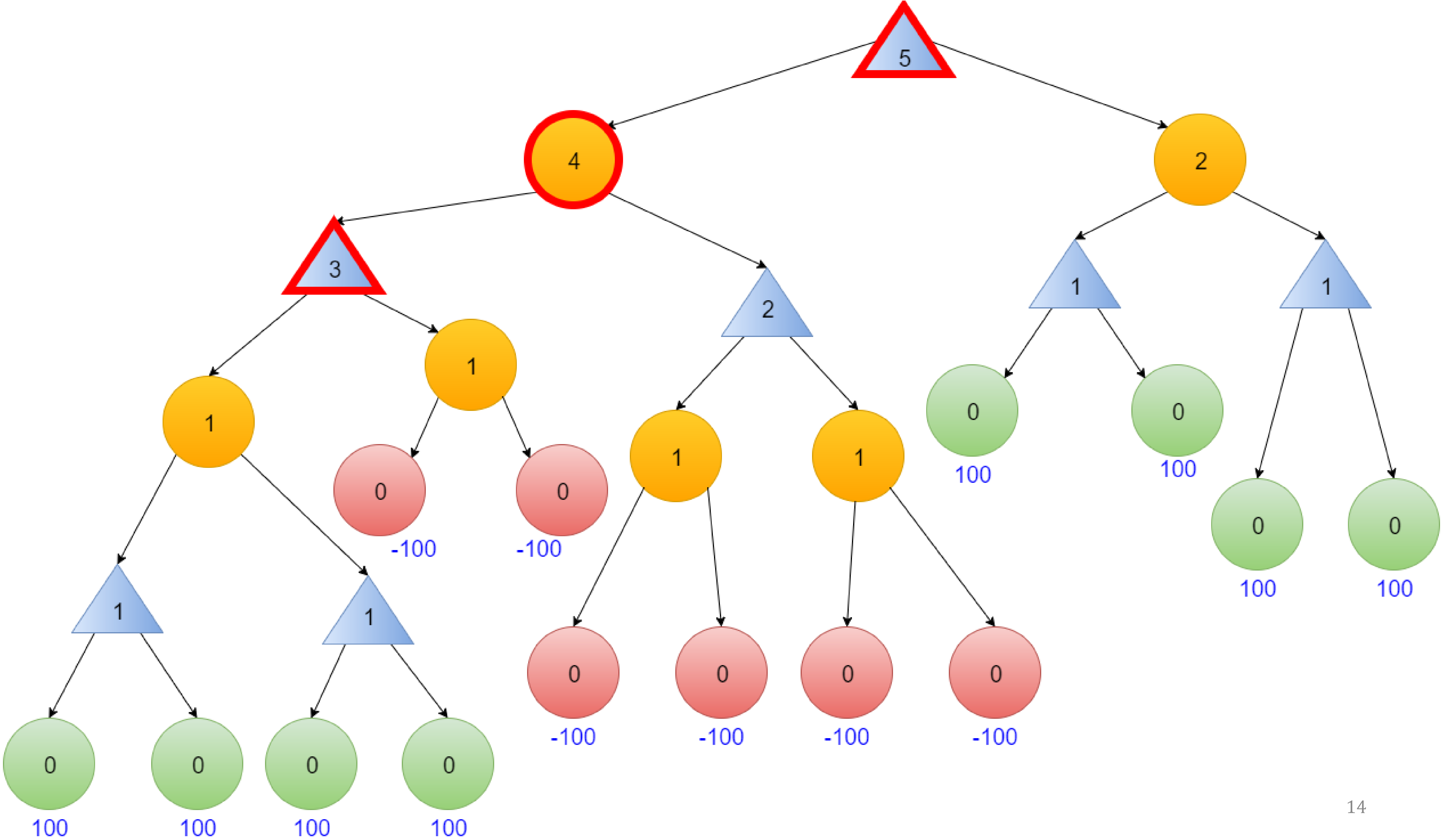
- Decrements if the number is odd
- Uniformly chooses to decrement or halve if the number is even

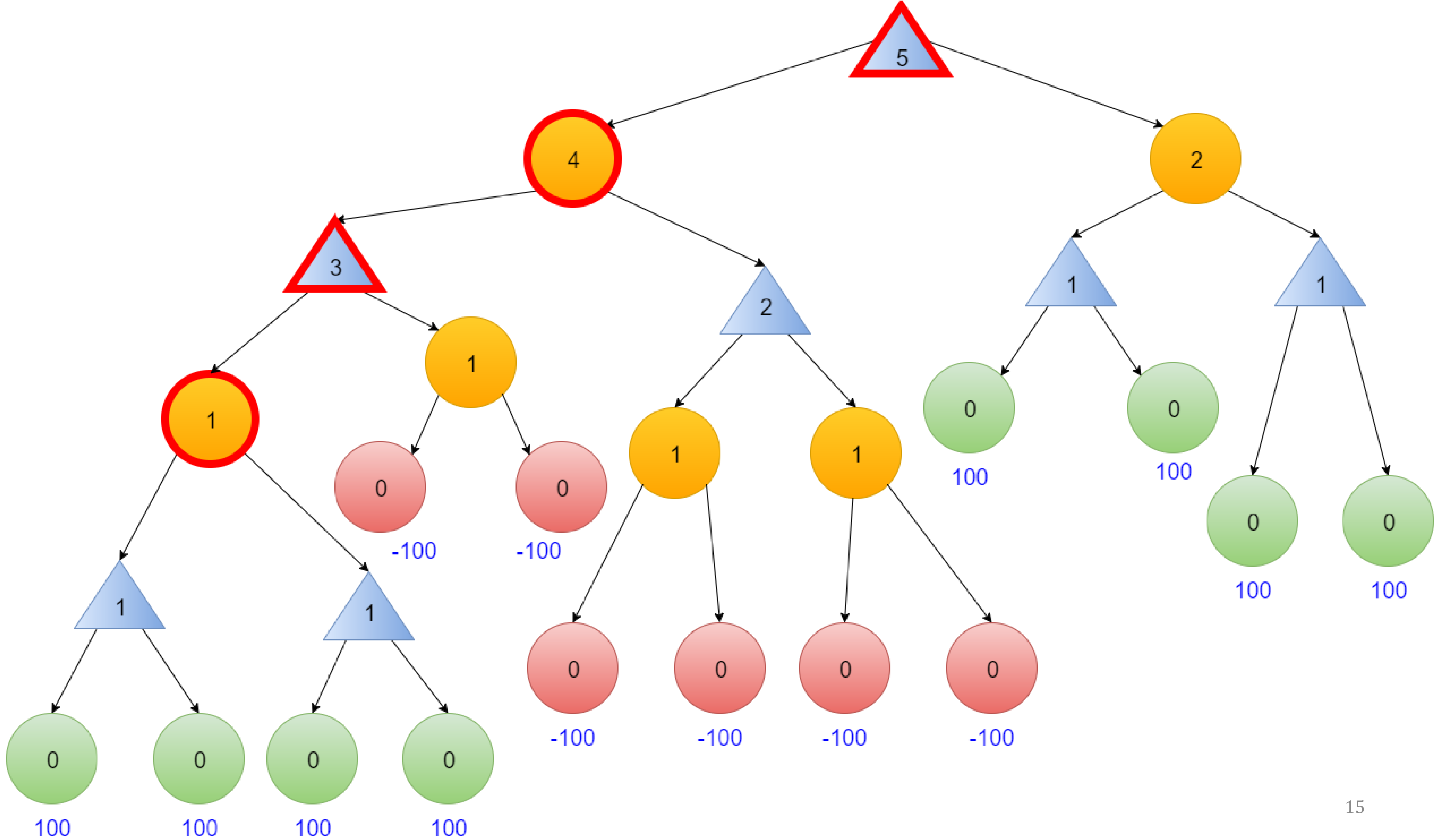
Thus the opponent's policy is given by

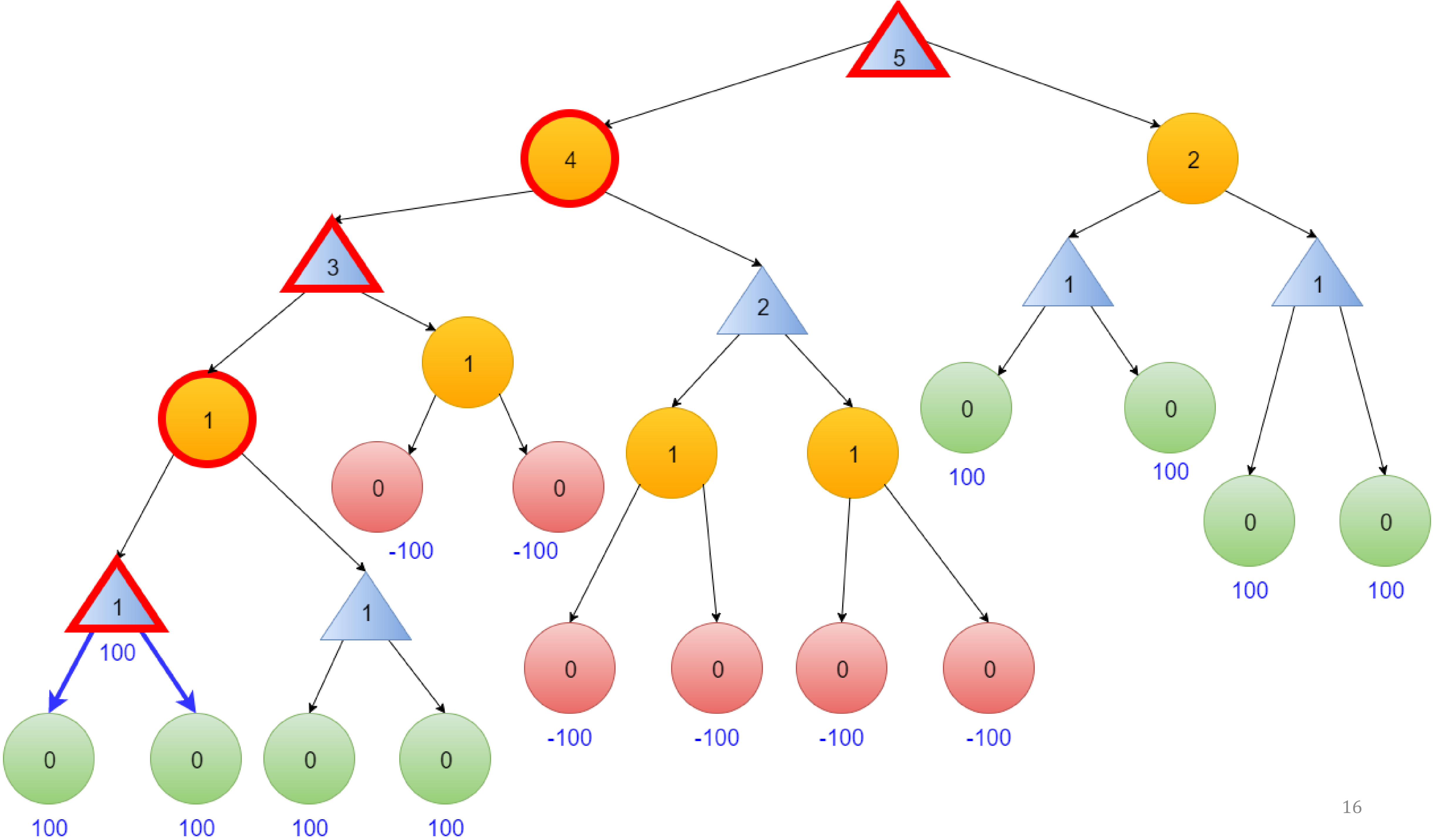
$$\pi_{opp} = \begin{cases} \text{Decrement 1} & \text{Number is odd} \\ \text{Choose uniformly} & \text{Number is even} \end{cases}$$

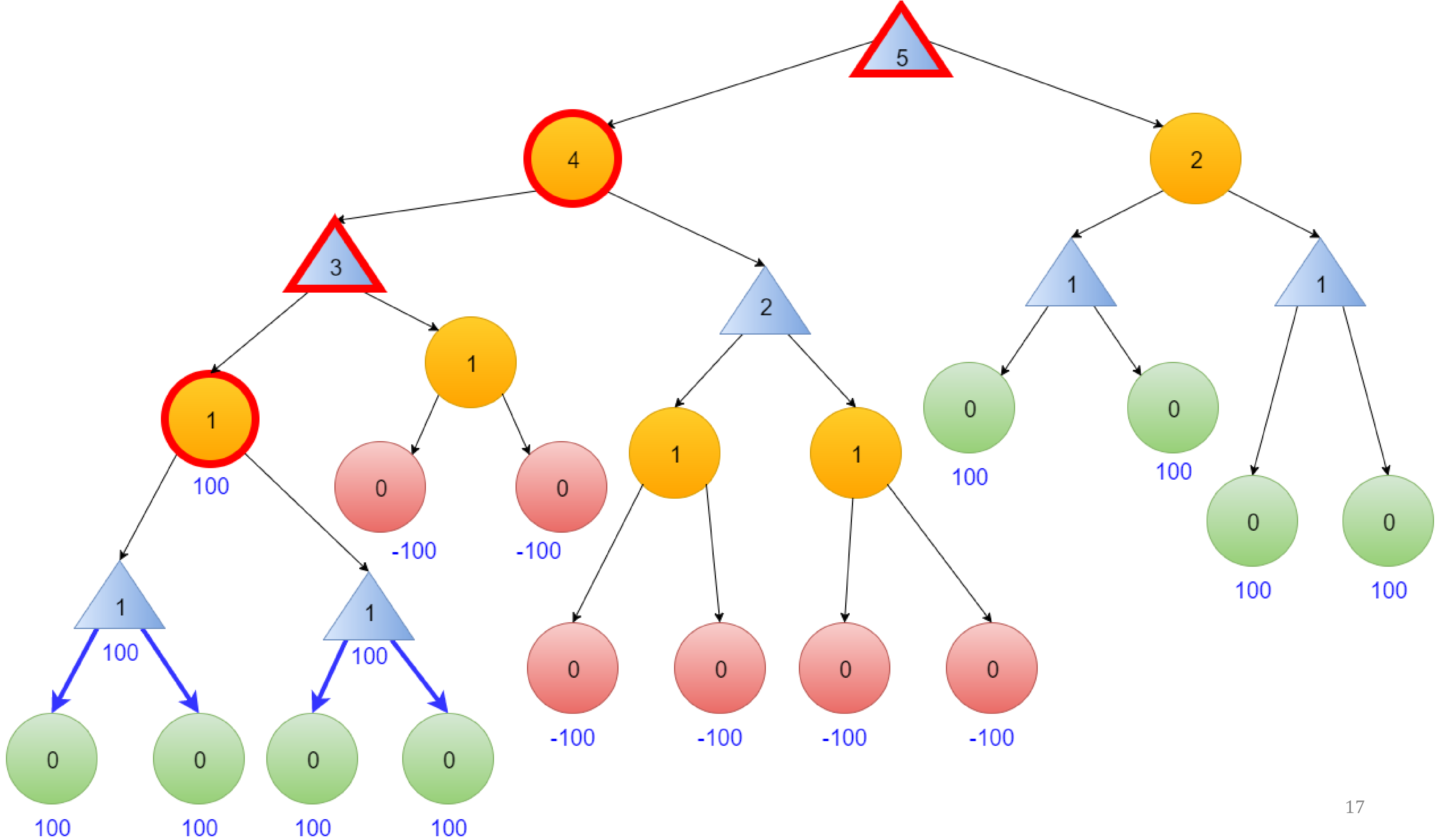

5
4
2
3
2
1
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
100
100
-100
-100
100
100
-100
-100
-100
-100
100
100
100
100


5
4
2
3
2
1
1
1
0
0
-100
-100
1
1
0
0
100
100
0
0
100
100
1
1
0
0
0
0
-100
-100
-100
-100
0
0
0
0
100
100
100
100

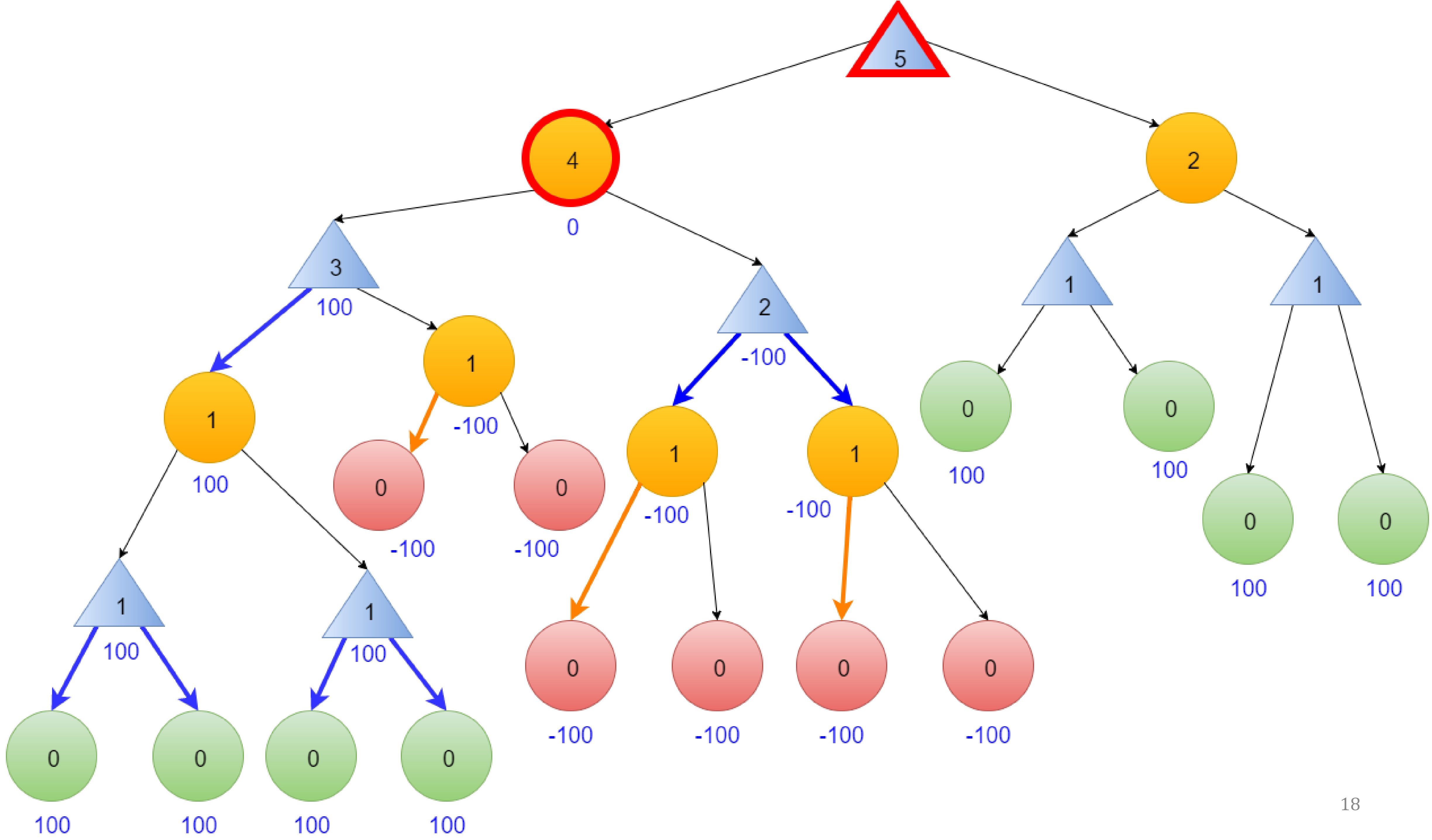

5
4
2
3
2
1
1
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
-100
-100
-100
-100
-100
-100
100
100
100
100
100
100
100
100

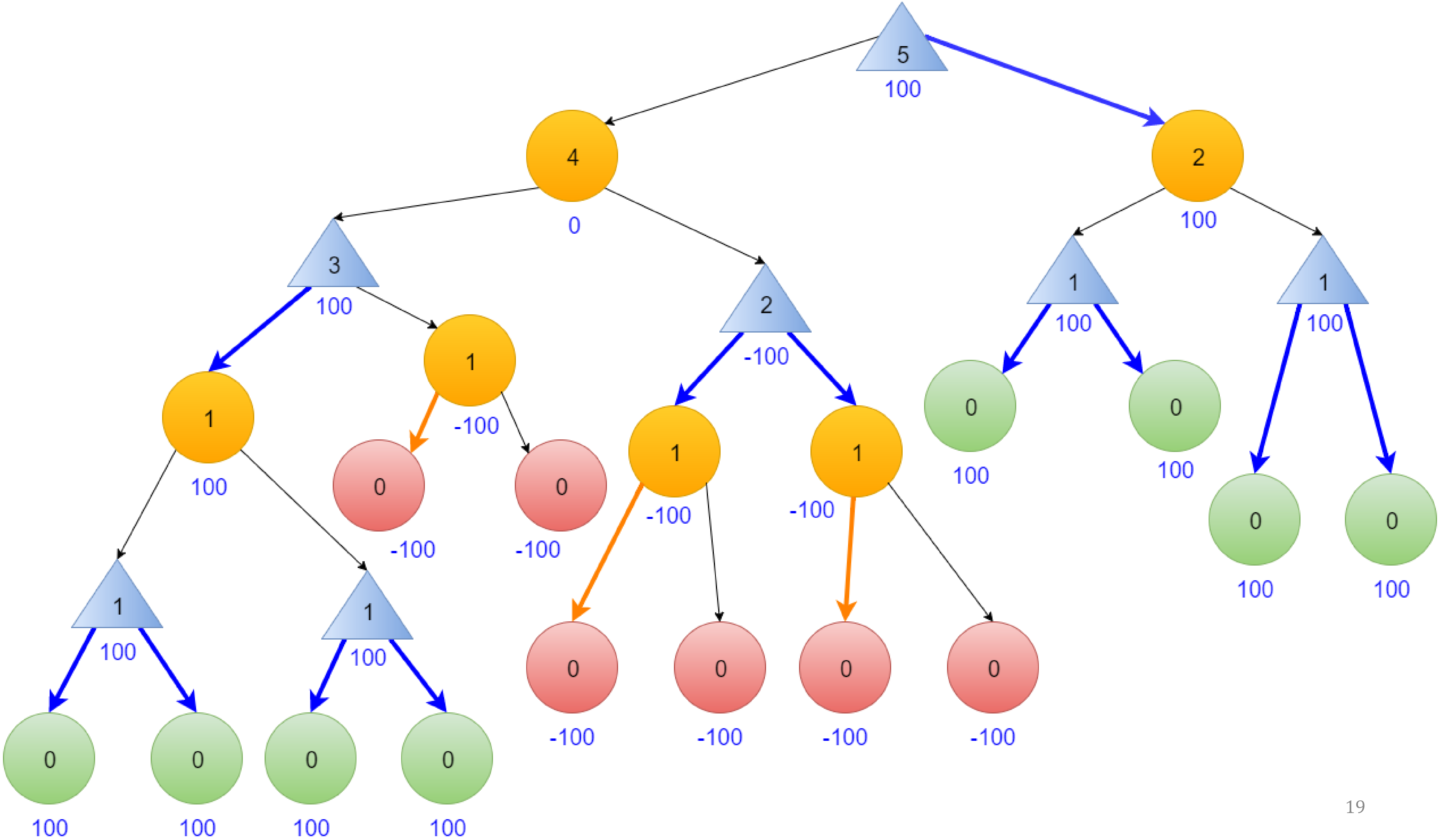

5
4
2
3
2
1
1
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
100
100
100
100
100
100
100
100
-100
-100
-100
-100
-100
-100


5
4
2
3
2
1
1
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
100
-100
-100
-100
-100
-100
-100
100
100
100
100
100
100
100
100
100



5
4
2
3
2
1
1
1
1
1
1
1
100
0
0
100
100
0
0
-100
-100
100
100
1
1
0
0
0
0
100
100
100
100
0
0
0
0
-100
-100
-100
-100
0
0
0
0
100
100
100
100

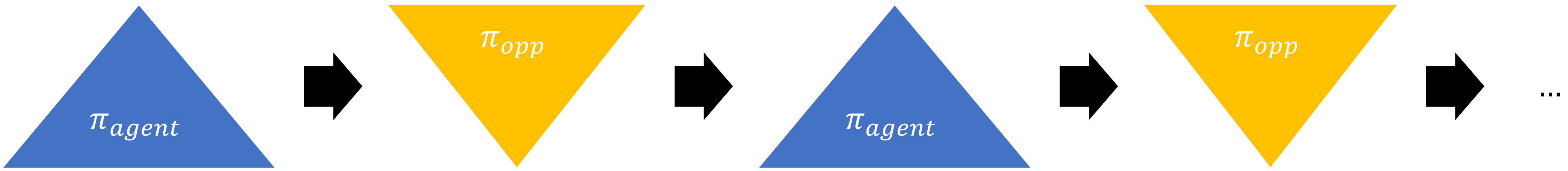

5
4
0
2
3
100
2
-100
1
1
1
1
100
-100
-100
-100
1
1
0
0
0
0
100
100
100
100
0
0
-100
-100
1
1
100
100
0
0
0
0
-100
-100
-100
-100
0
0
100
100
0
0
0
0
100
100
100
100



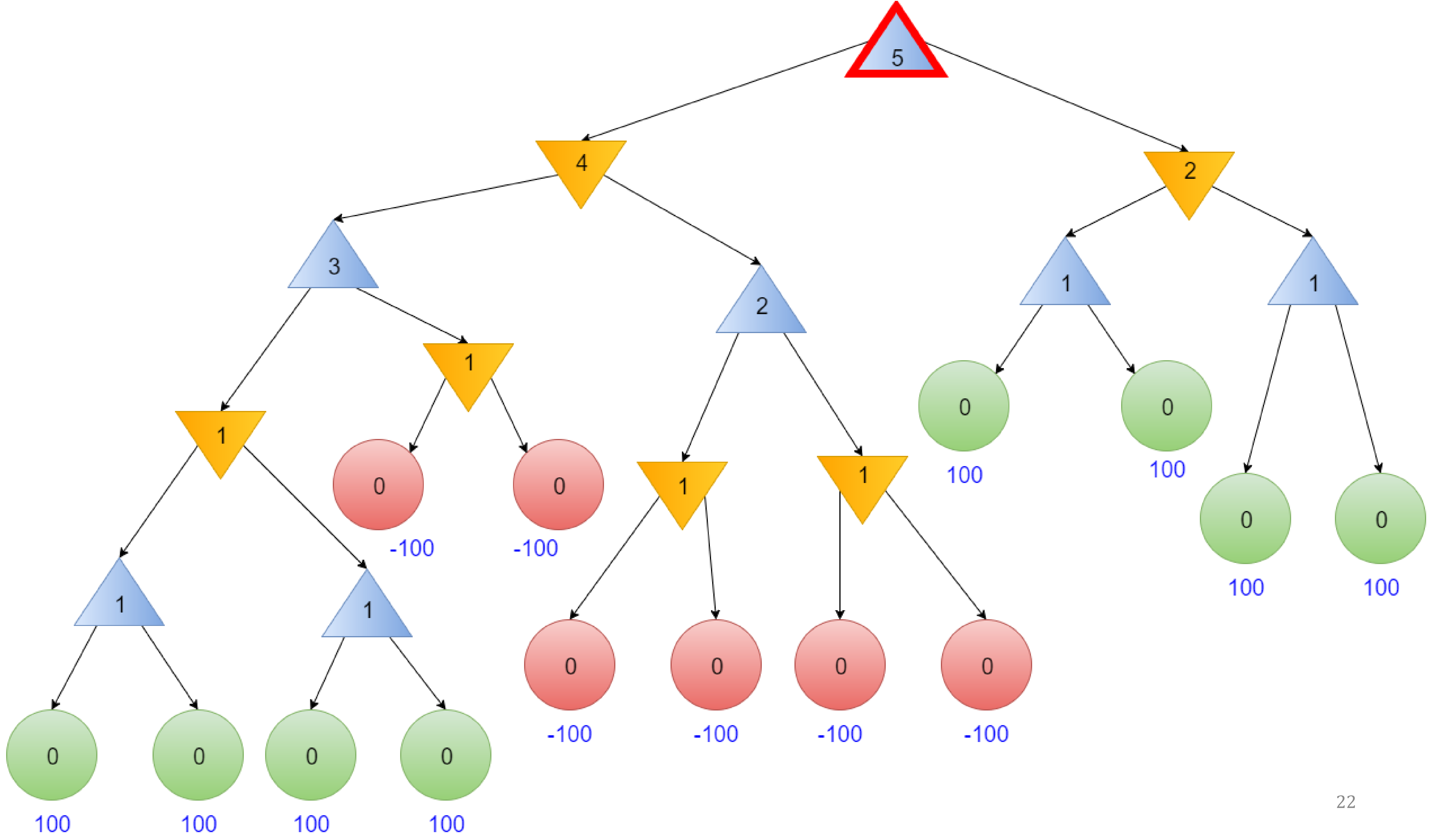
5
100
4
0
2
100
3
100
1
100
2
-100
1
100
1
100
1
100
1
-100
1
-100
1
-100
0
100
0
100
0
-100
0
-100
1
100
1
100
0
100
0
100
0
-100
0
-100
0
-100
0
-100
0
100
0
100
0
100
0
100

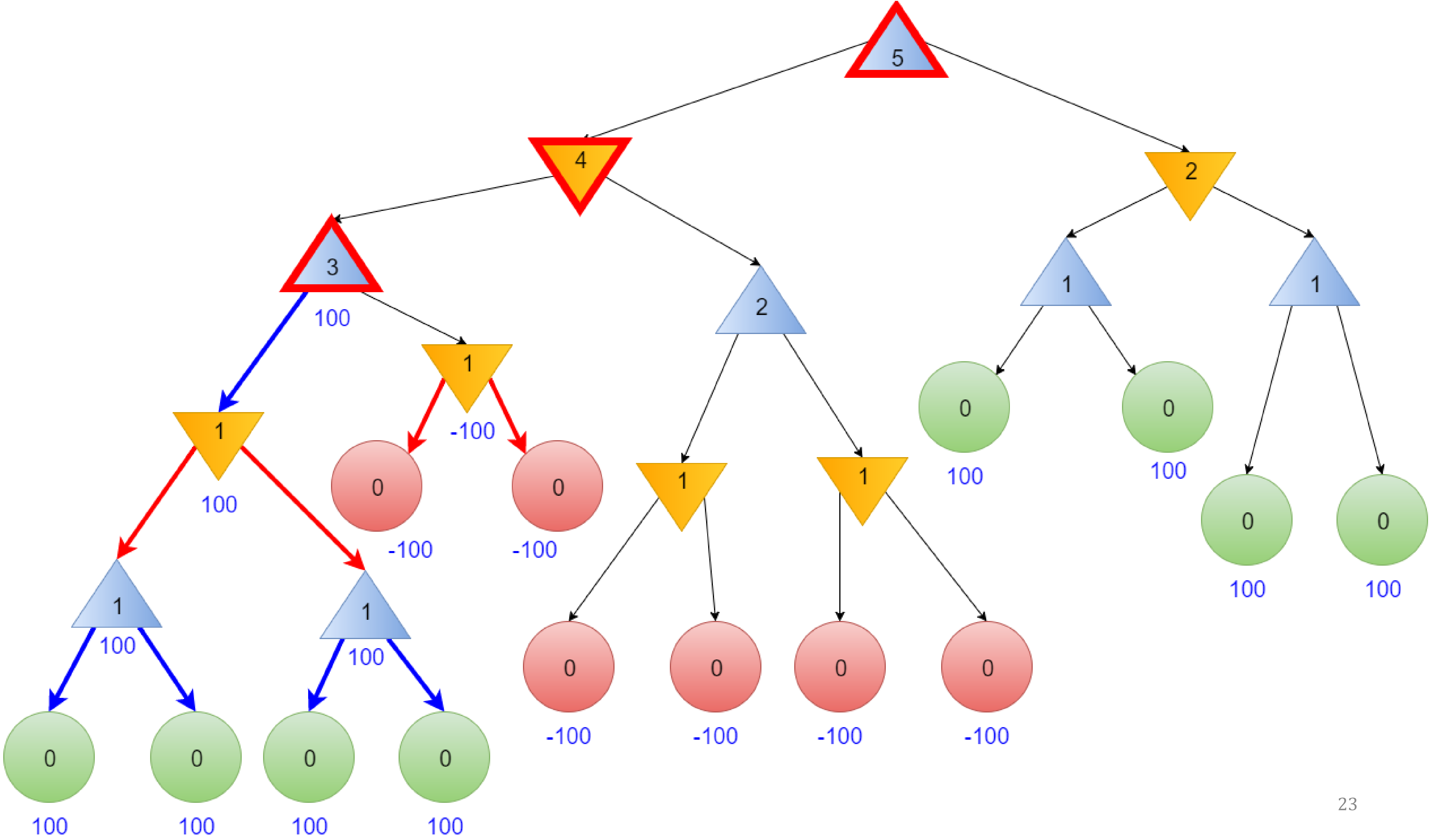
- Game Trees
- Expectimax
- Minimax
- TD Learning

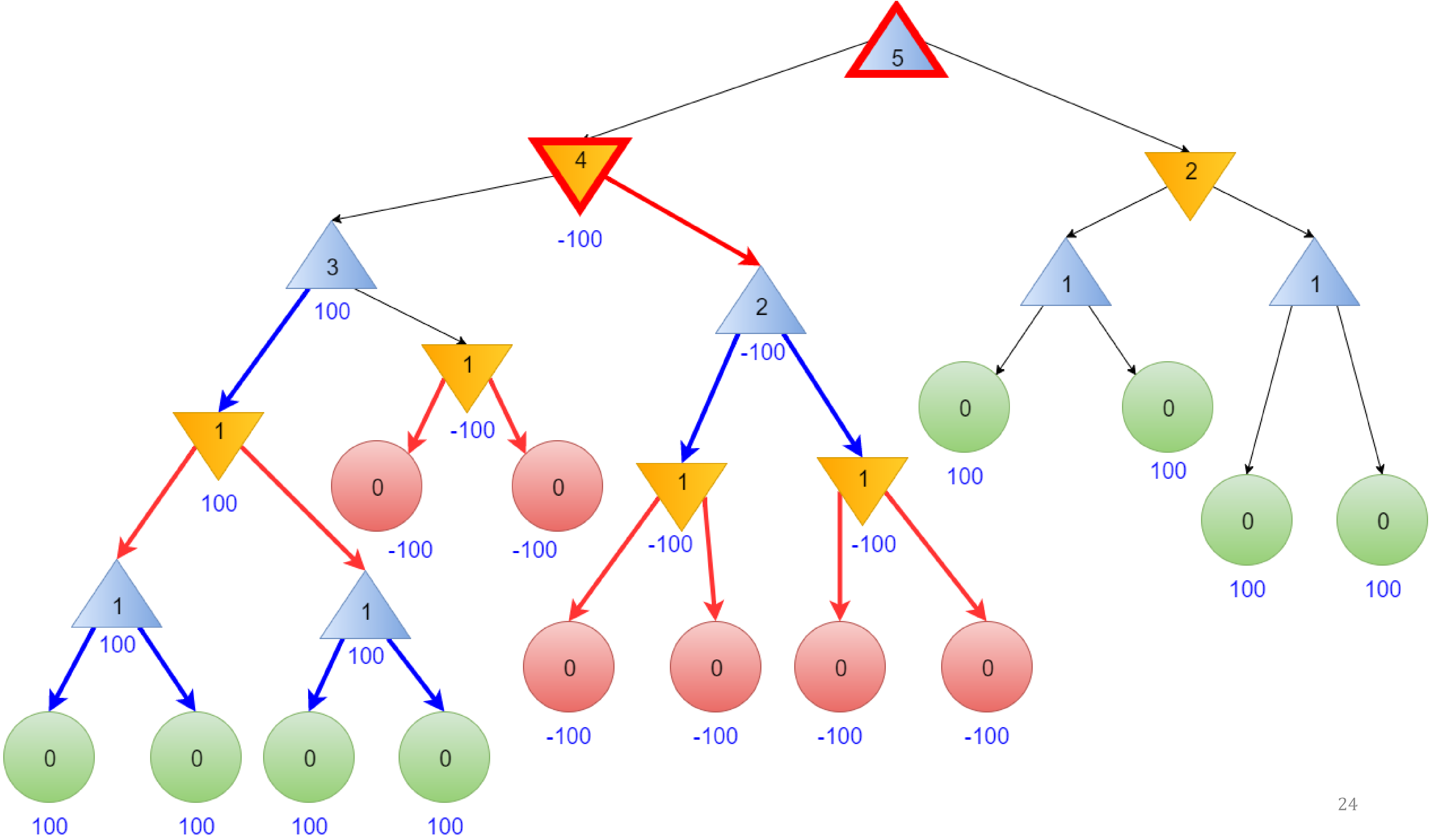
Minimax Recurrence

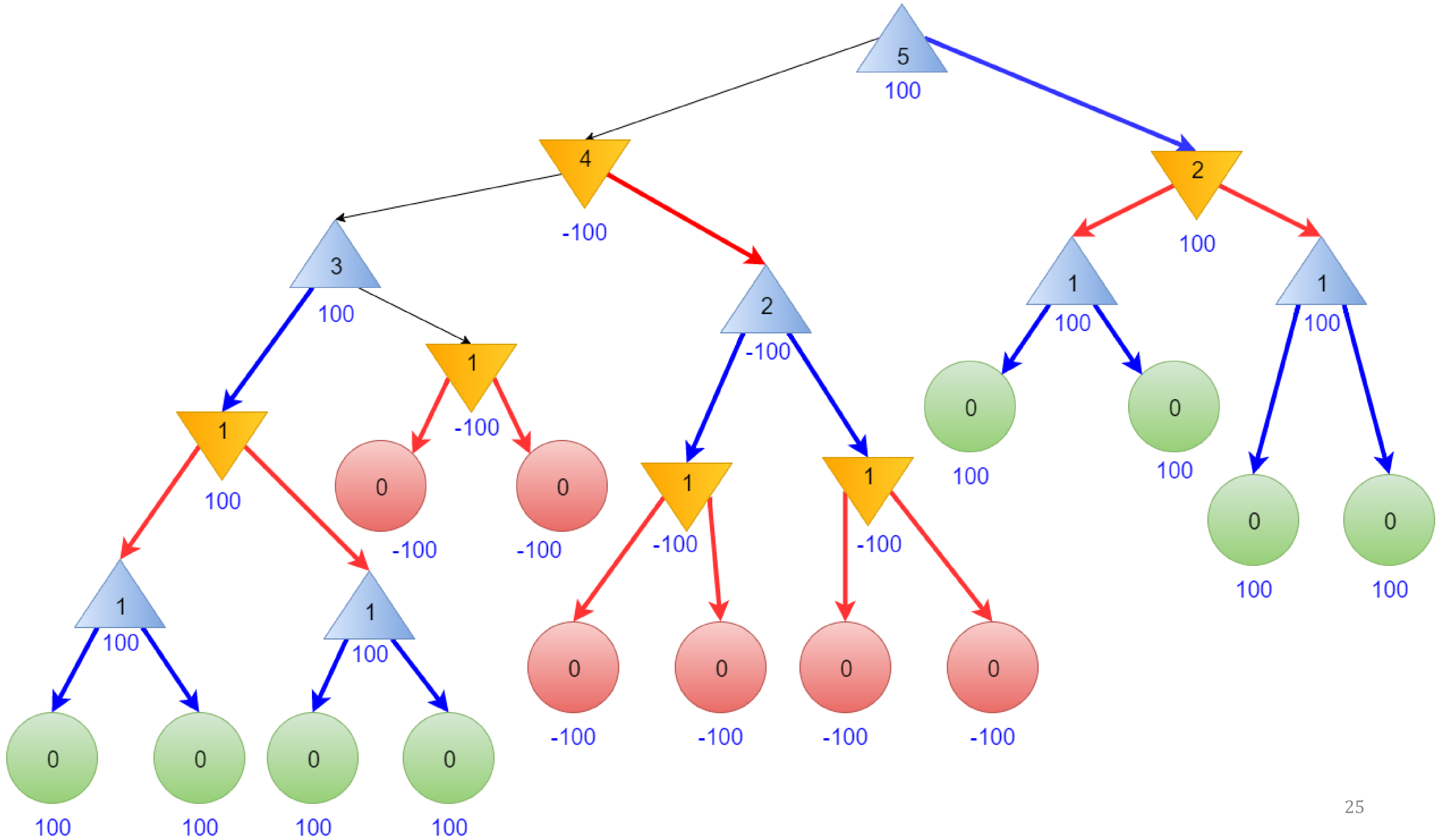


$$V_{opt}(s) = \begin{cases} \text{Utility}(s) & \text{IsEnd}(s) \\ \max\limits_{a \in \text{Actions}(s)} V_{opt}\big(Succ(s,a)\big) & \text{Player}(s) = \text{agent} \\ \min\limits_{a \in \text{Actions}(s)} V_{opt}\big(Succ(s,a)\big) & \text{Player}(s) = \text{opp} \end{cases}$$


5
4
2
3
2
1
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
-100
-100
-100
-100
-100
-100
100
100
100
100
100
100
100
100


5
4
2
3
2
1
1
100
1
1
1
1
0
0
100
100
-100
1
0
0
0
0
100
100
-100
-100
1
1
0
0
100
100
0
0
0
0
-100
-100
-100
-100
0
0
0
0
100
100
100
100


5
4
-100
2
3
100
2
-100
1
1
1
1
1
100
-100
-100
-100
0
0
0
0
100
100
100
100
1
1
100
100
0
0
-100
-100
0
0
0
0
-100
-100
-100
-100
0
0
0
0
100
100
100
100


5
100
4
-100
2
100
3
100
2
-100
1
100
1
100
1
-100
1
100
0
0
1
-100
1
-100
0
100
0
100
0
0
1
100
1
100
0
100
0
100
-100
-100
0
0
0
0
-100
-100
-100
-100
0
0
0
0
100
100
100
100

- Game Trees
- Expectimax
- Minimax
- TD Learning

TD Learning

Key idea: temporal difference (TD) learning

Use Monte Carlo simulations to generate data.
Learn weights $\mathbf{w}$ of evaluation function from data.

TD Learning

- Unlike Q-learning, TD learning is an on policy algorithm.

- We have fixed policies $\pi_{\text{agent}}, \pi_{\text{opp}}$ which are supposed to approximate minimax policies.

$$\pi_{\text{agent}} = \underset{a \in \text{Actions}(s)}{\text{argmax}} \; V(Succ(s, a); w)$$

$$\pi_{\text{opp}} = \underset{a \in \text{Actions}(s)}{\text{argmin}} \; V(Succ(s, a); w)$$

TD Learning

Small piece of experience from Monte Carlo Simulation

$$(s, a, r, s')$$

Prediction

$$V(s; w)$$

Target

$$r + \gamma V(s'; w)$$

TD Learning example

We define features for our example,

$$\phi(s) = \begin{bmatrix} 1\{\text{Is opponent's turn?}\} \\ 1\{\text{number is odd}\} \end{bmatrix}$$

Note that for linear value functions,

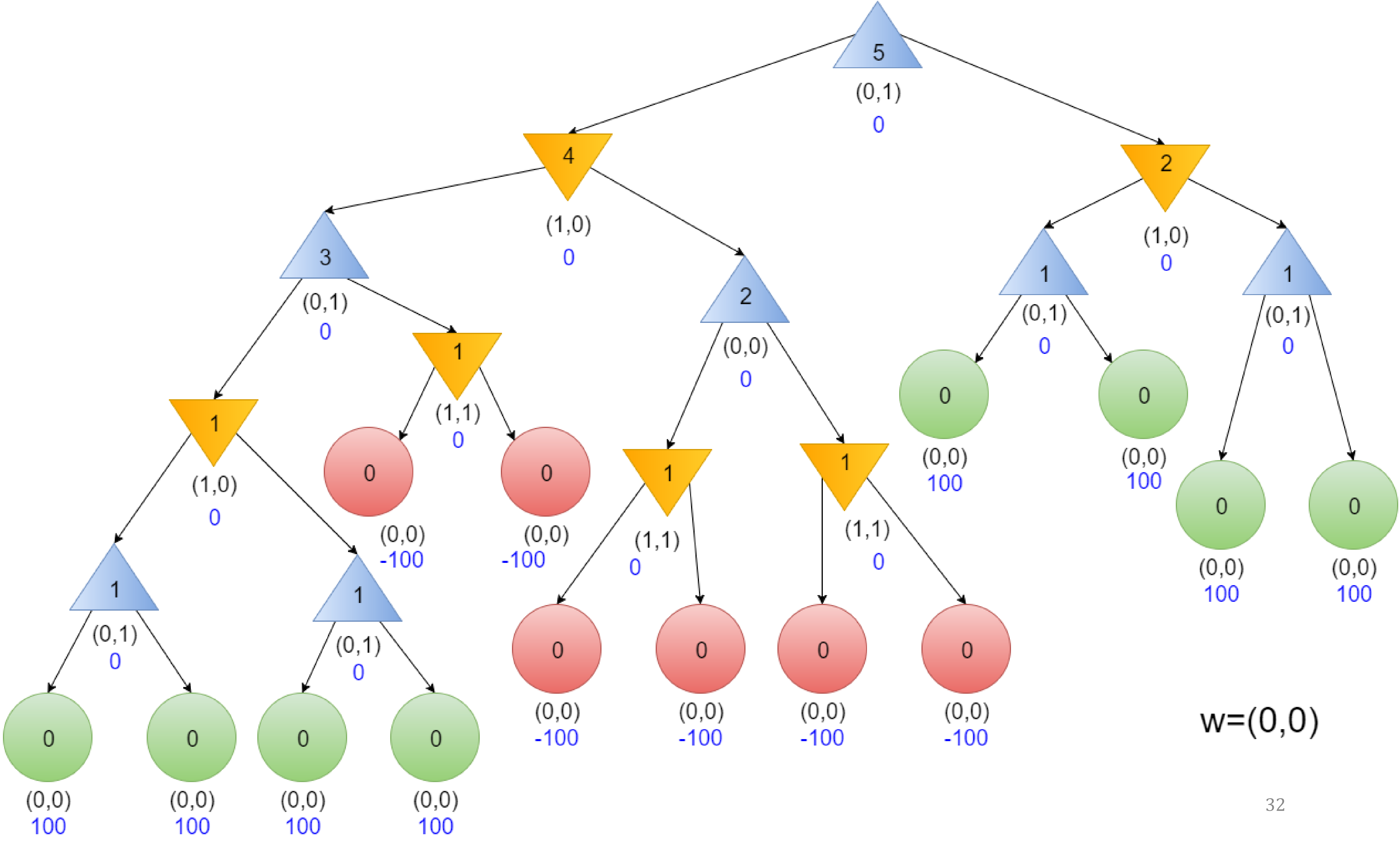
$$V(s; w) = w^T \phi(s)$$

$$\nabla_w V(s; w) = \phi(s)$$

TD Learning example

Assume $\eta = 0.5, \gamma = 1$

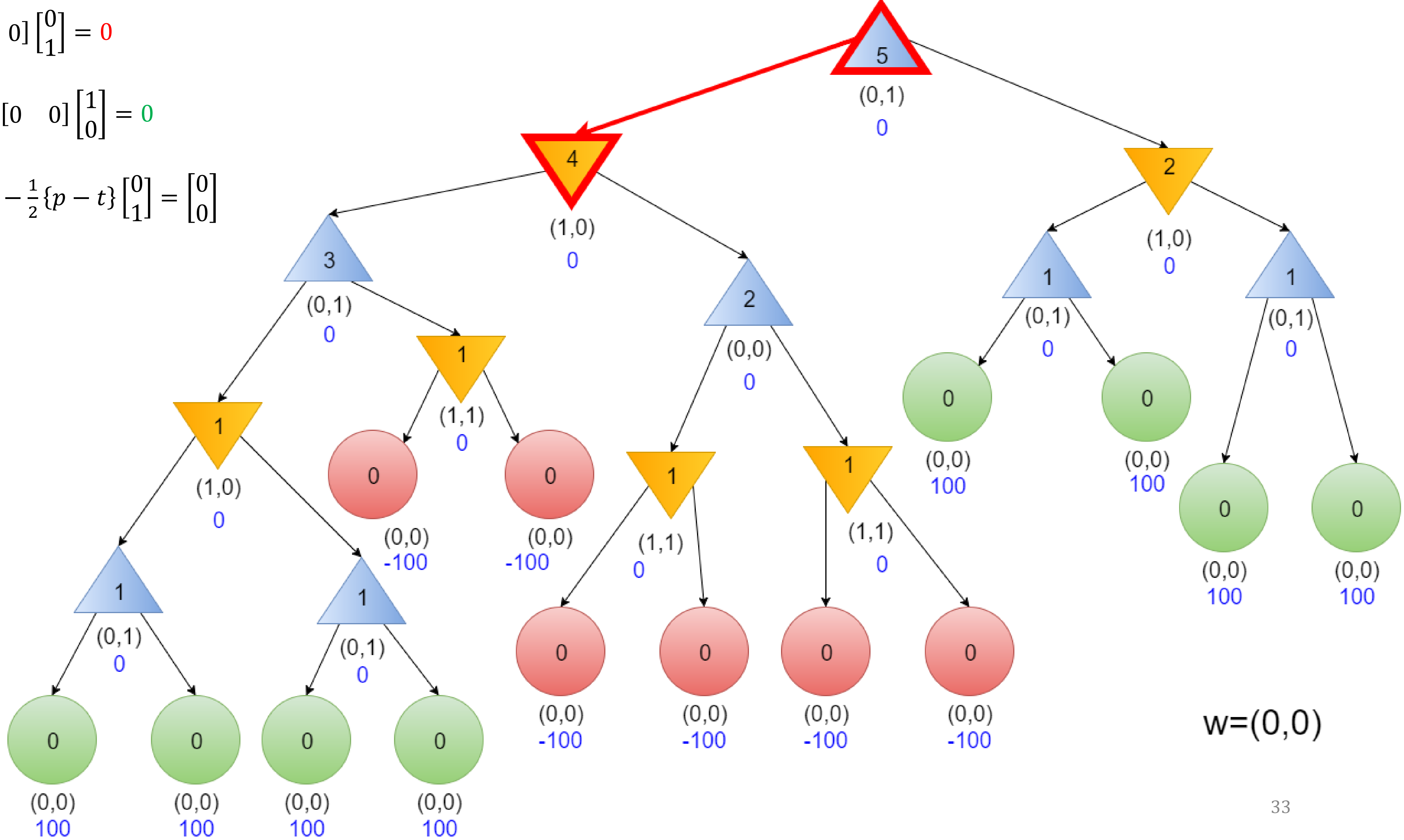
Iteration 1


5
(0,1)
0
4
(1,0)
0
2
(1,0)
0
3
(0,1)
0
2
(0,0)
0
1
(0,1)
0
1
(0,1)
0
1
(1,0)
0
1
(1,1)
0
1
(1,1)
0
1
(1,1)
0
0
(0,0)
100
0
(0,0)
-100
w=(0,0)

$$p = \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0$$

$$t = 0 + \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0$$

$$w = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \frac{1}{2}\{p - t\} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

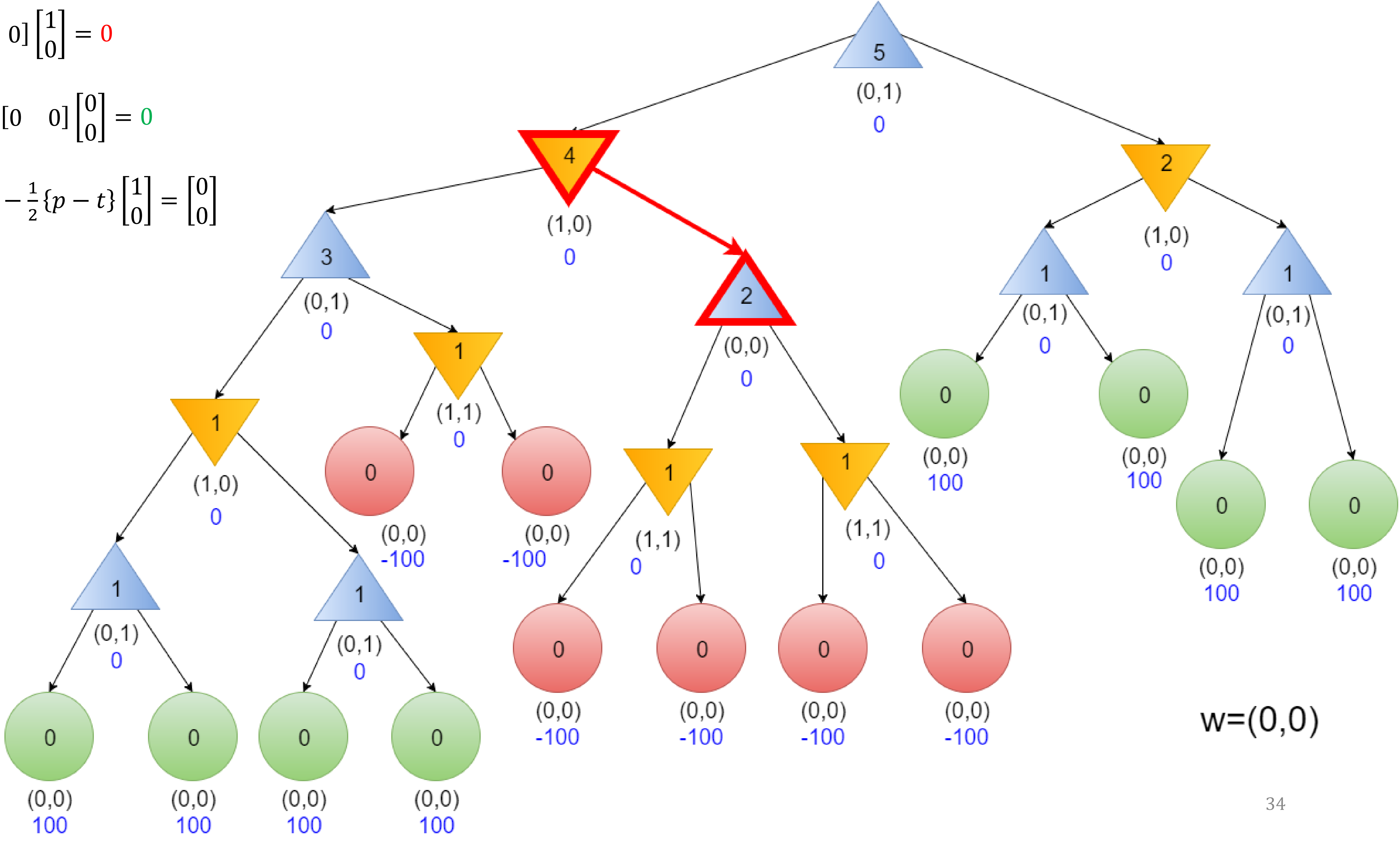


w=(0,0)

$$p = [0 \quad 0]\begin{bmatrix}1\\0\end{bmatrix} = 0$$

$$t = 0 + [0 \quad 0]\begin{bmatrix}0\\0\end{bmatrix} = 0$$

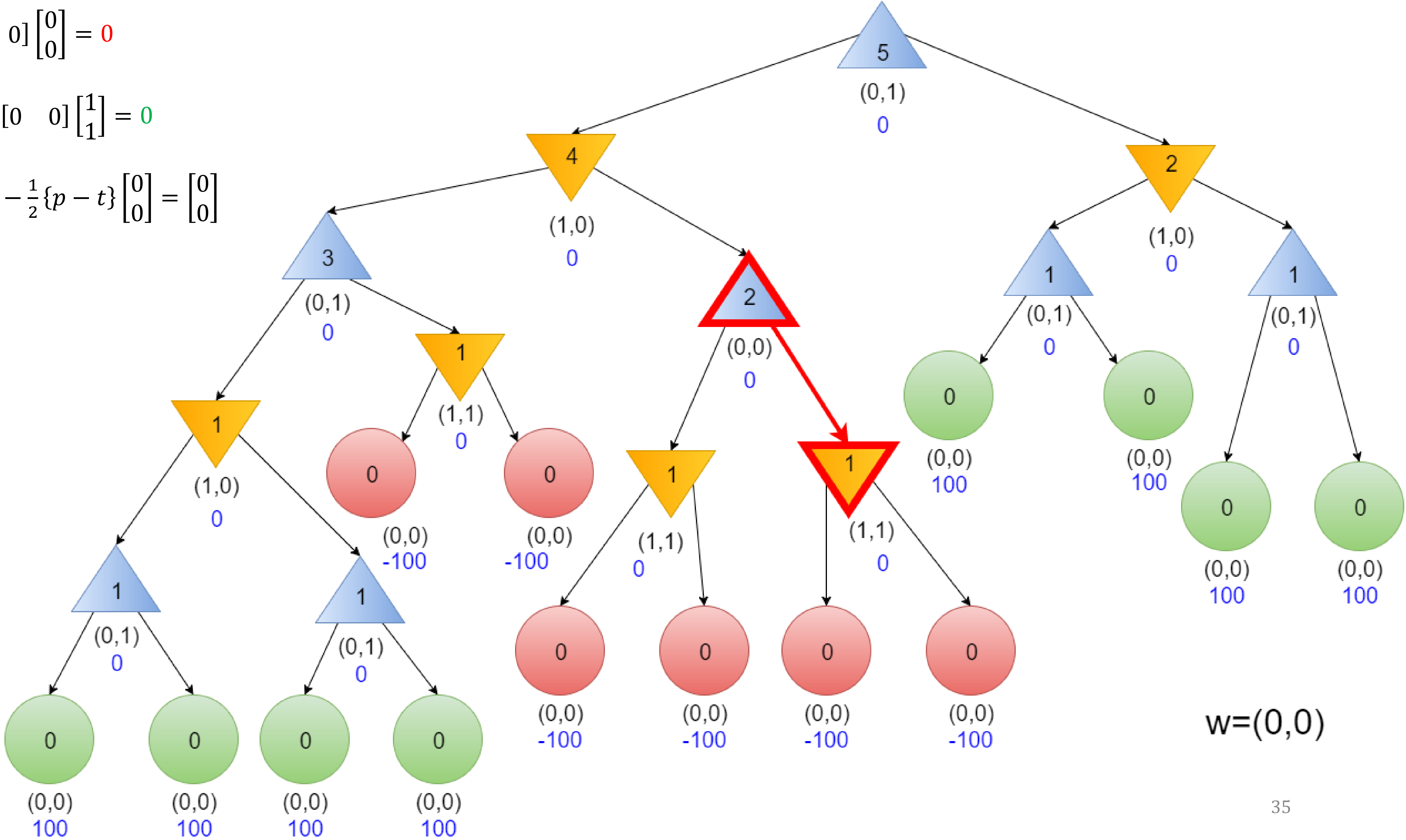
$$w = \begin{bmatrix}0\\0\end{bmatrix} - \frac{1}{2}\{p - t\}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$



$$p = \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = 0$$

$$t = 0 + \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 0$$

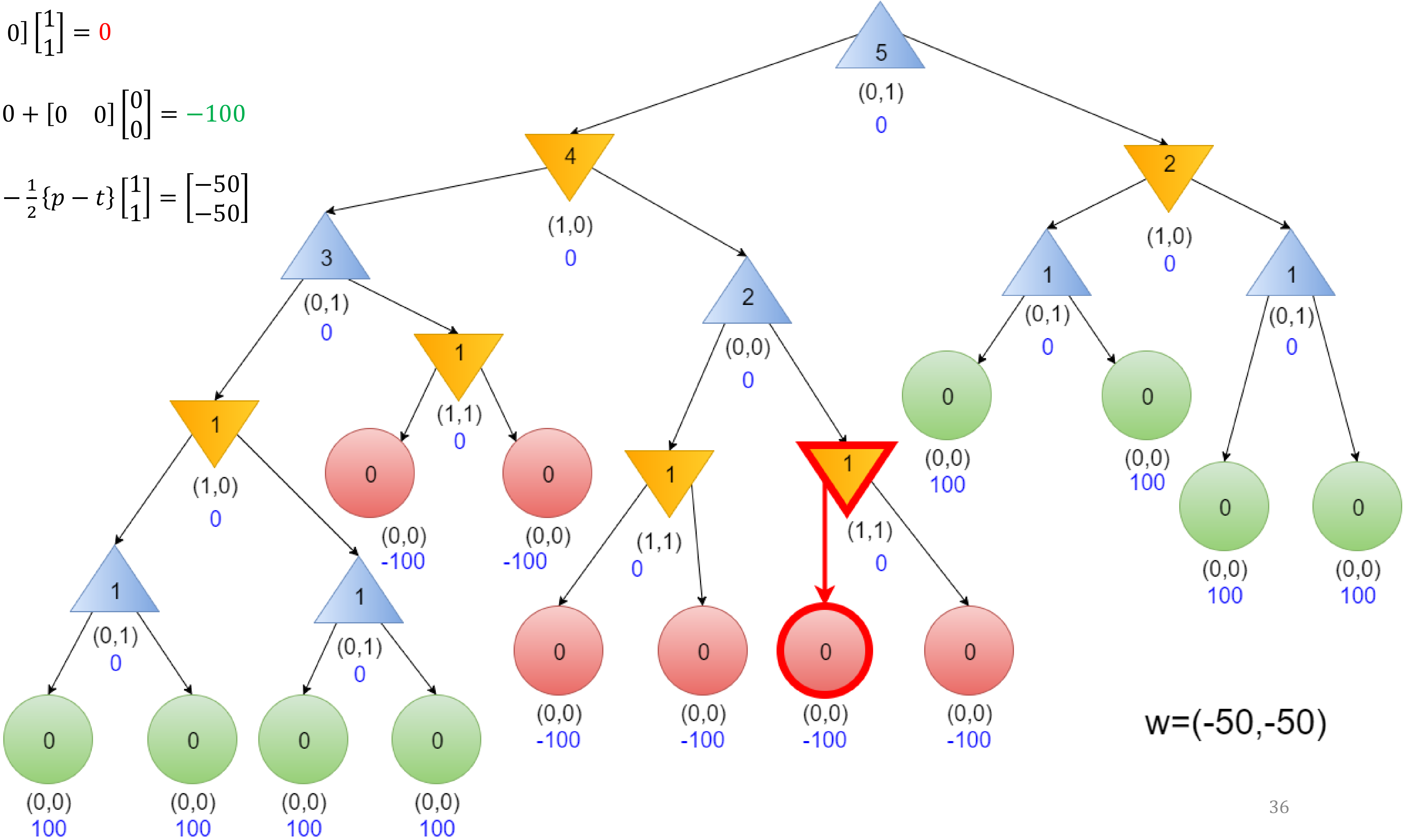
$$w = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \frac{1}{2}\{p - t\} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$



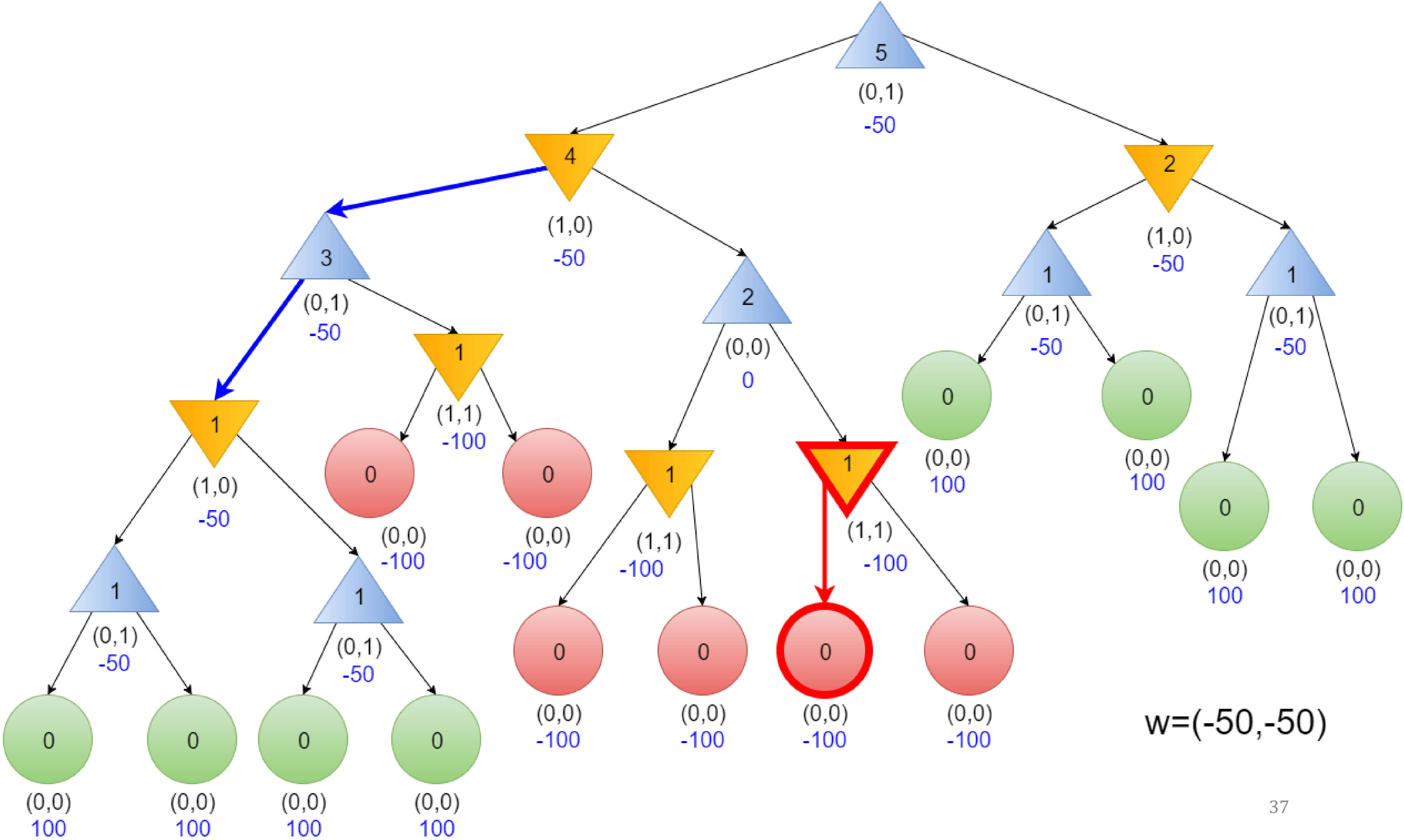
$$p = [0 \quad 0]\begin{bmatrix}1\\1\end{bmatrix} = 0$$

$$t = -100 + [0 \quad 0]\begin{bmatrix}0\\0\end{bmatrix} = -100$$

$$w = \begin{bmatrix}0\\0\end{bmatrix} - \tfrac{1}{2}\{p - t\}\begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}-50\\-50\end{bmatrix}$$



w=(-50,-50)



5
(0,1)
-50
4
(1,0)
-50
2
(1,0)
-50
3
(0,1)
-50
2
(0,0)
0
1
(0,1)
-50
1
(0,1)
-50
1
(1,0)
-50
1
(1,1)
-100
1
(1,1)
-100
1
(1,1)
-100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
-100
0
(0,0)
-100
1
(0,1)
-50
1
(0,1)
-50
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
w=(-50,-50)

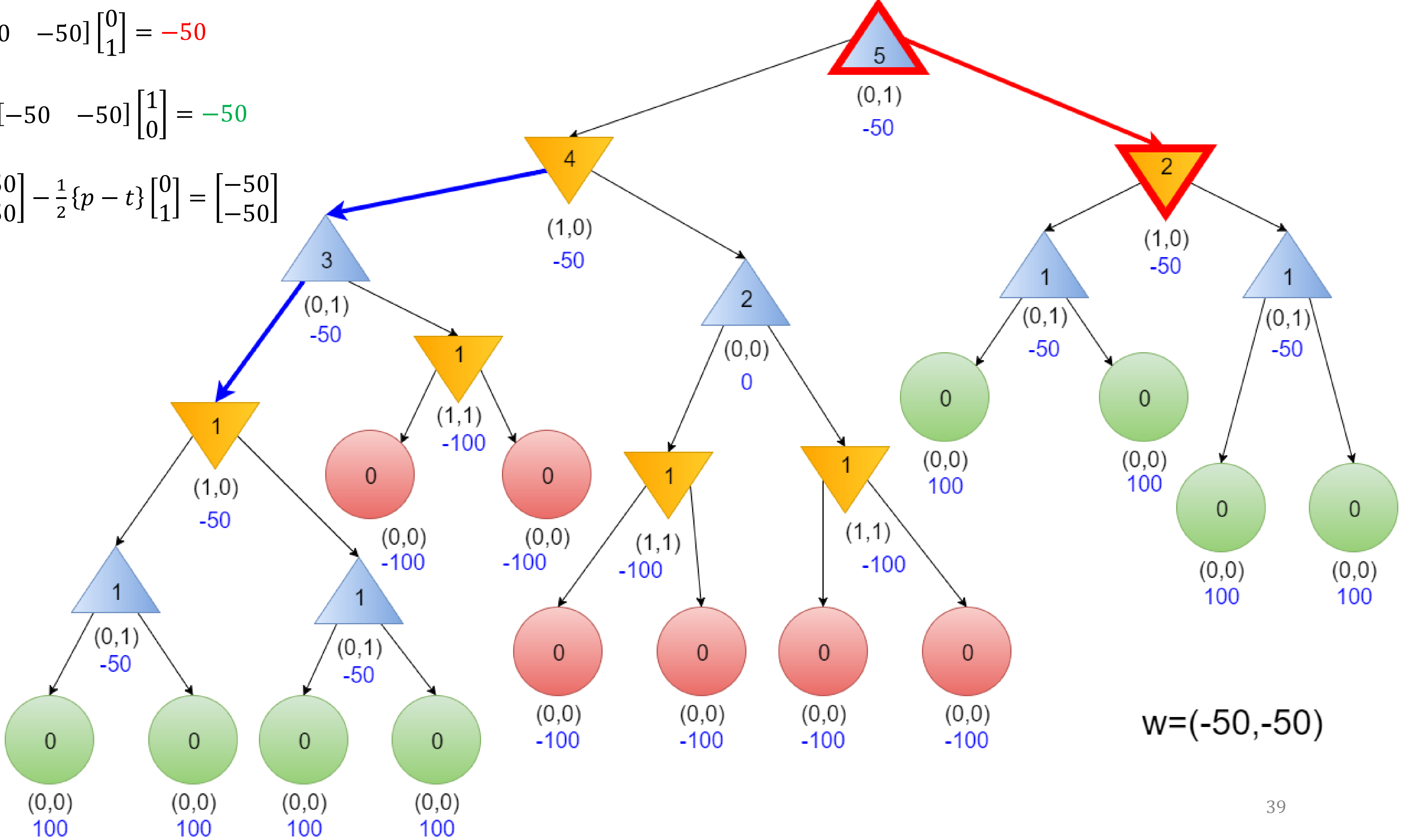
TD Learning example

Iteration 2

$$p = \begin{bmatrix}-50 & -50\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} = -50$$

$$t = 0 + \begin{bmatrix}-50 & -50\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = -50$$

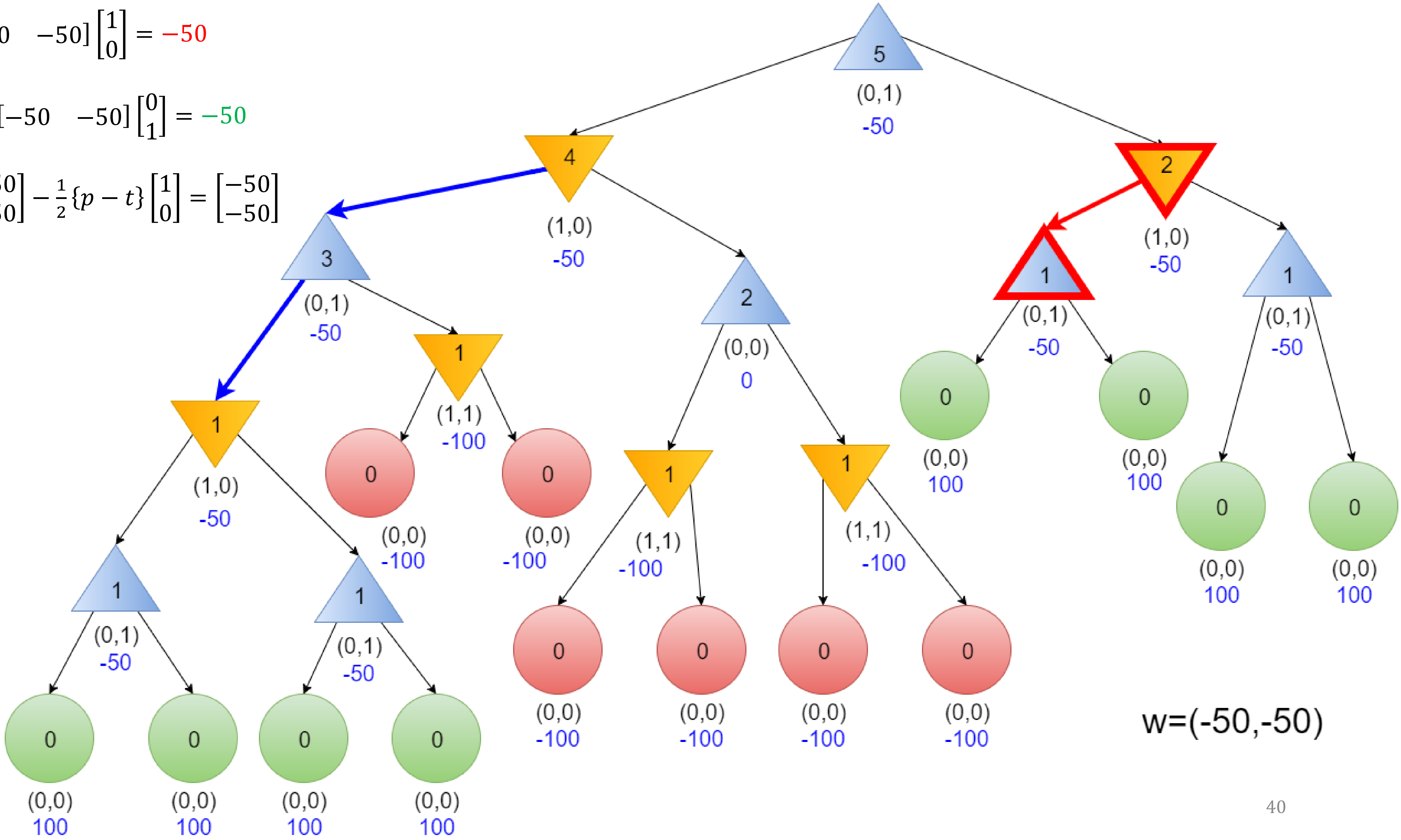
$$w = \begin{bmatrix}-50\\-50\end{bmatrix} - \frac{1}{2}\{p - t\}\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}-50\\-50\end{bmatrix}$$



$$p = [-50 \quad -50]\begin{bmatrix}1\\0\end{bmatrix} = -50$$

$$t = 0 + [-50 \quad -50]\begin{bmatrix}0\\1\end{bmatrix} = -50$$

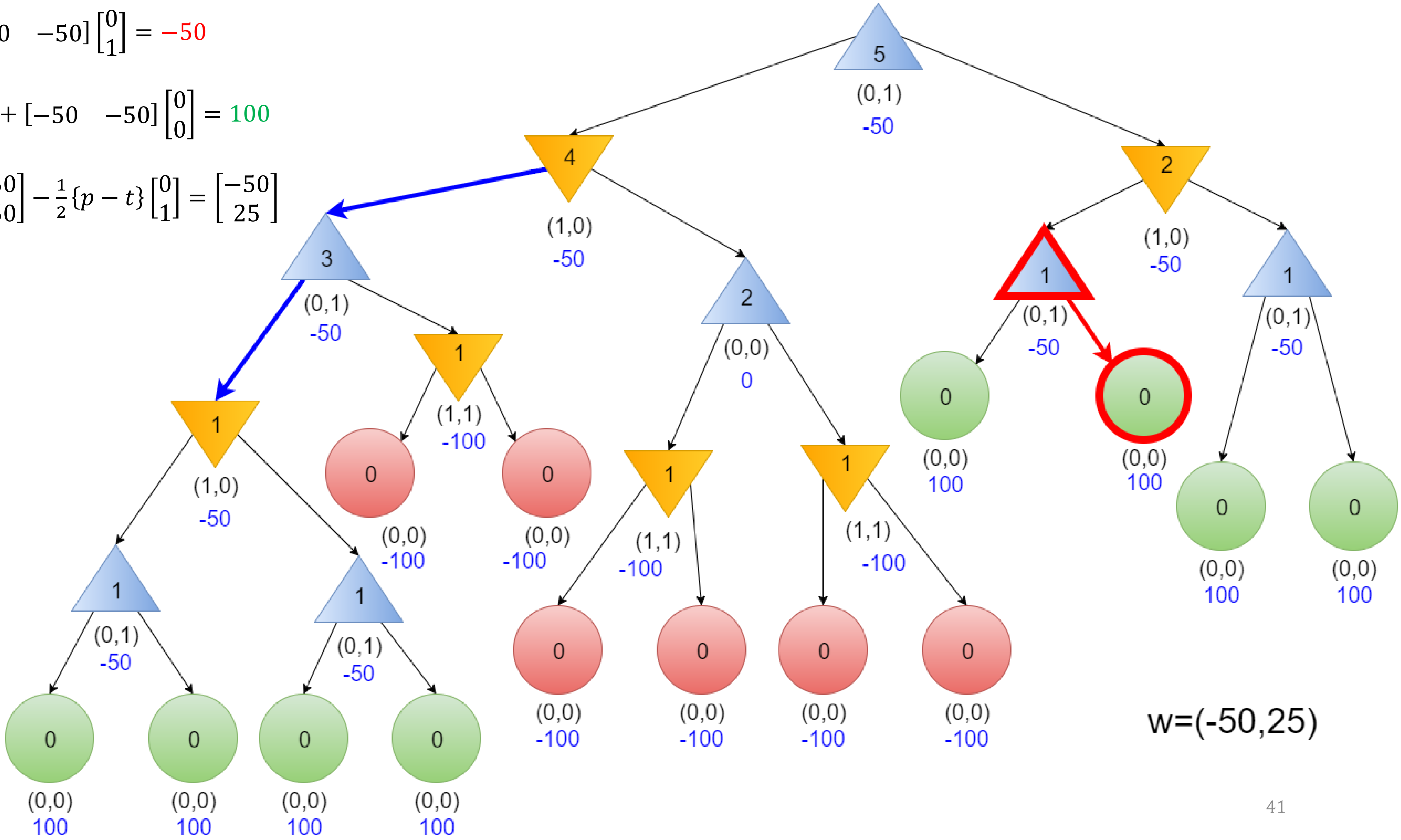
$$w = \begin{bmatrix}-50\\-50\end{bmatrix} - \frac{1}{2}\{p - t\}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}-50\\-50\end{bmatrix}$$



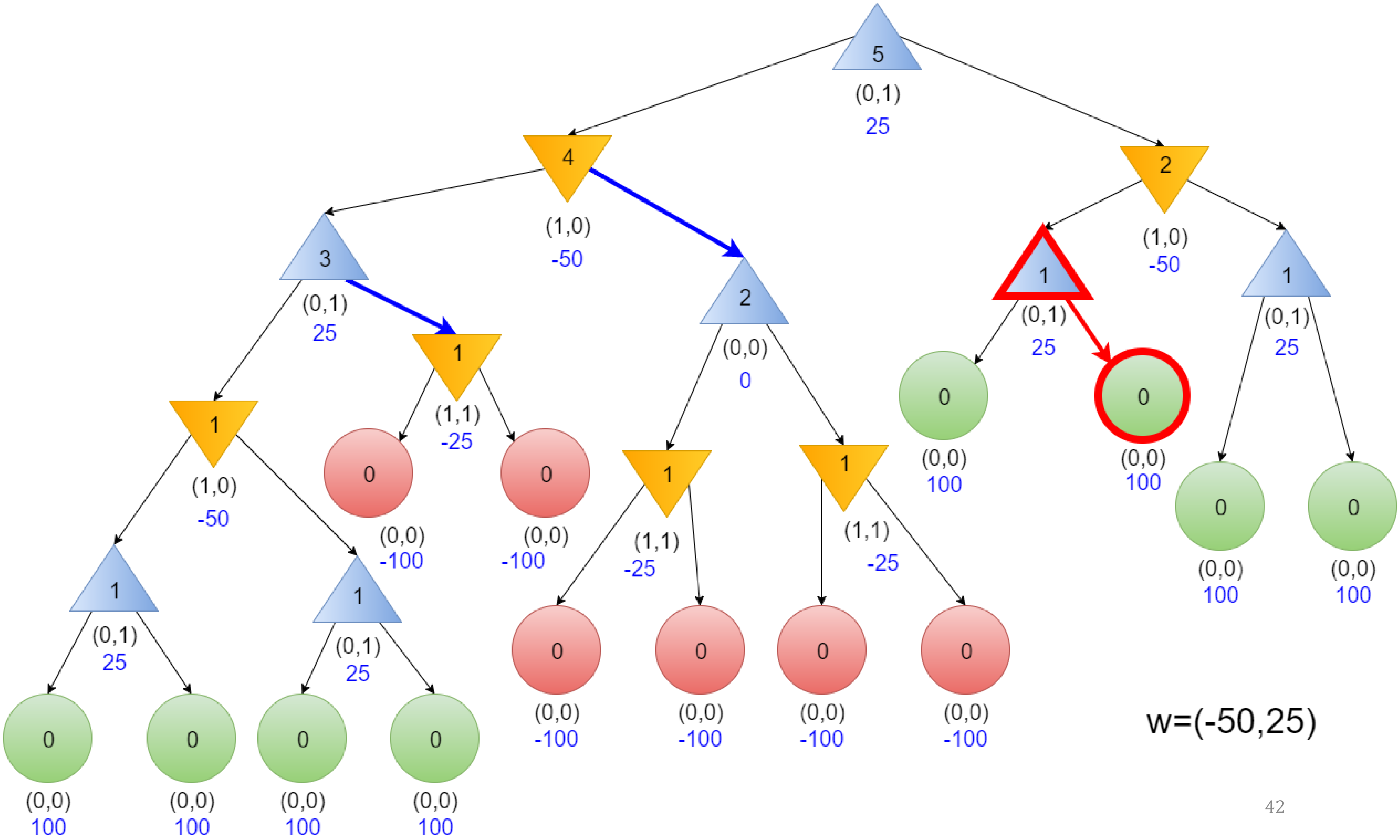
$$p = [-50 \quad -50]\begin{bmatrix}0\\1\end{bmatrix} = -50$$

$$t = 100 + [-50 \quad -50]\begin{bmatrix}0\\0\end{bmatrix} = 100$$

$$w = \begin{bmatrix}-50\\-50\end{bmatrix} - \frac{1}{2}\{p - t\}\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}-50\\25\end{bmatrix}$$



w=(-50,25)


5
(0,1)
25
4
(1,0)
-50
2
(1,0)
-50
3
(0,1)
25
2
(0,0)
0
1
(0,1)
25
1
(0,1)
25
1
(1,0)
-50
1
(1,1)
-25
1
(1,1)
-25
1
(1,1)
-25
1
(0,1)
25
1
(0,1)
25
1
(0,1)
25
1
(0,1)
25
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
-100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
0
(0,0)
100
w=(-50,25)