Abstraction-Based Decision Making

for Statistical Properties.

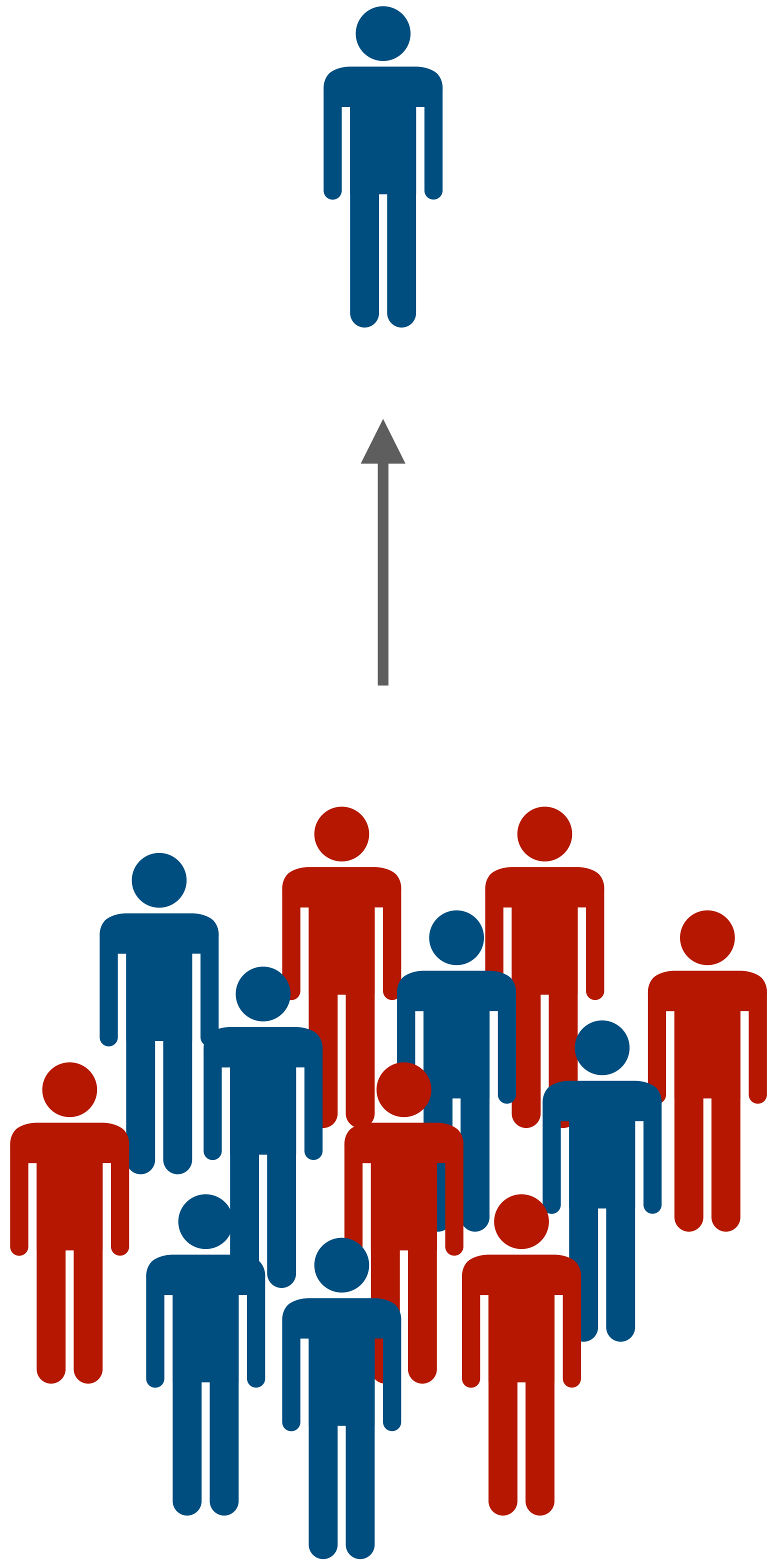
F. Cano | TA. Henzinger | B. Könighofer | K. Kueffner | K. Mallik

Institute of Science and Technology Austria

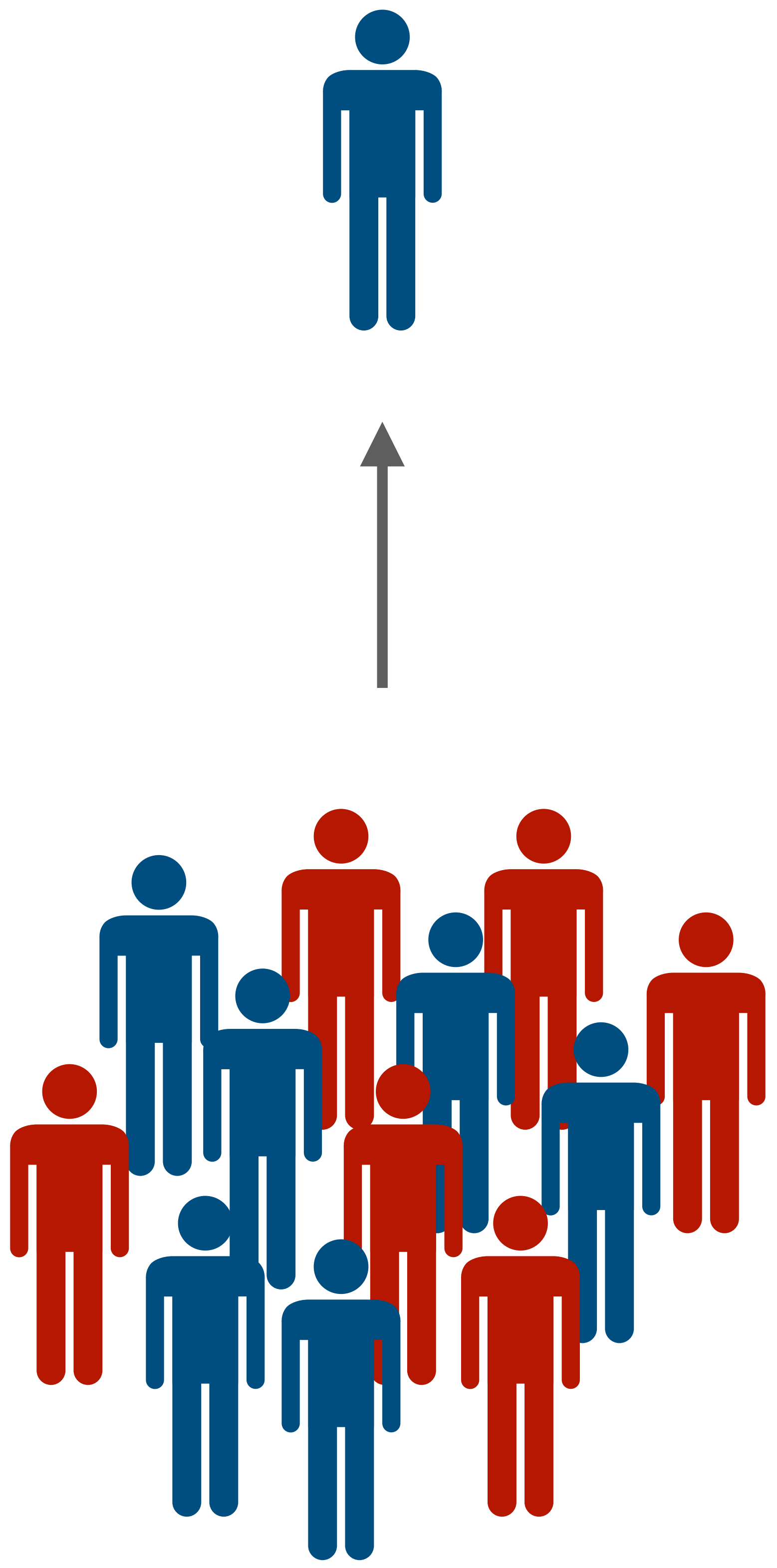
Motivation.

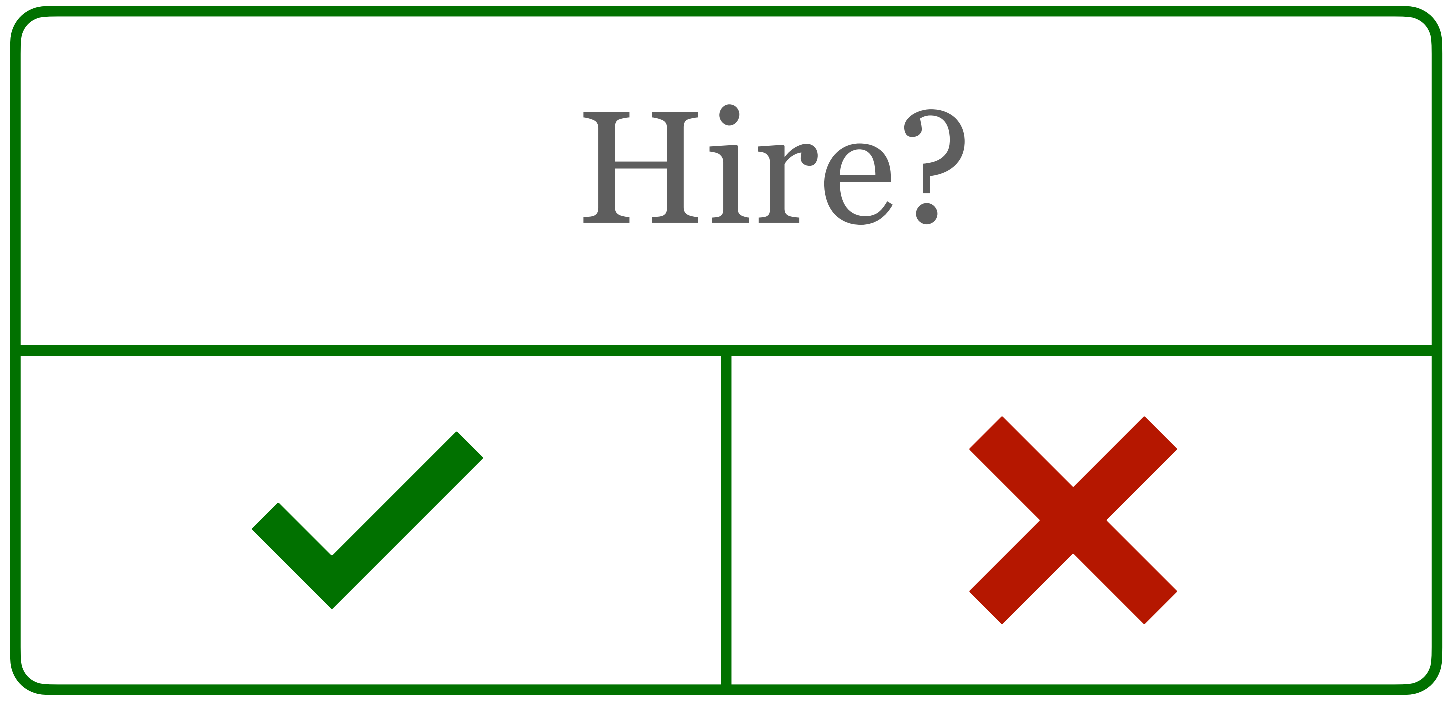
Algorithmic Fairness.

$t = 1$

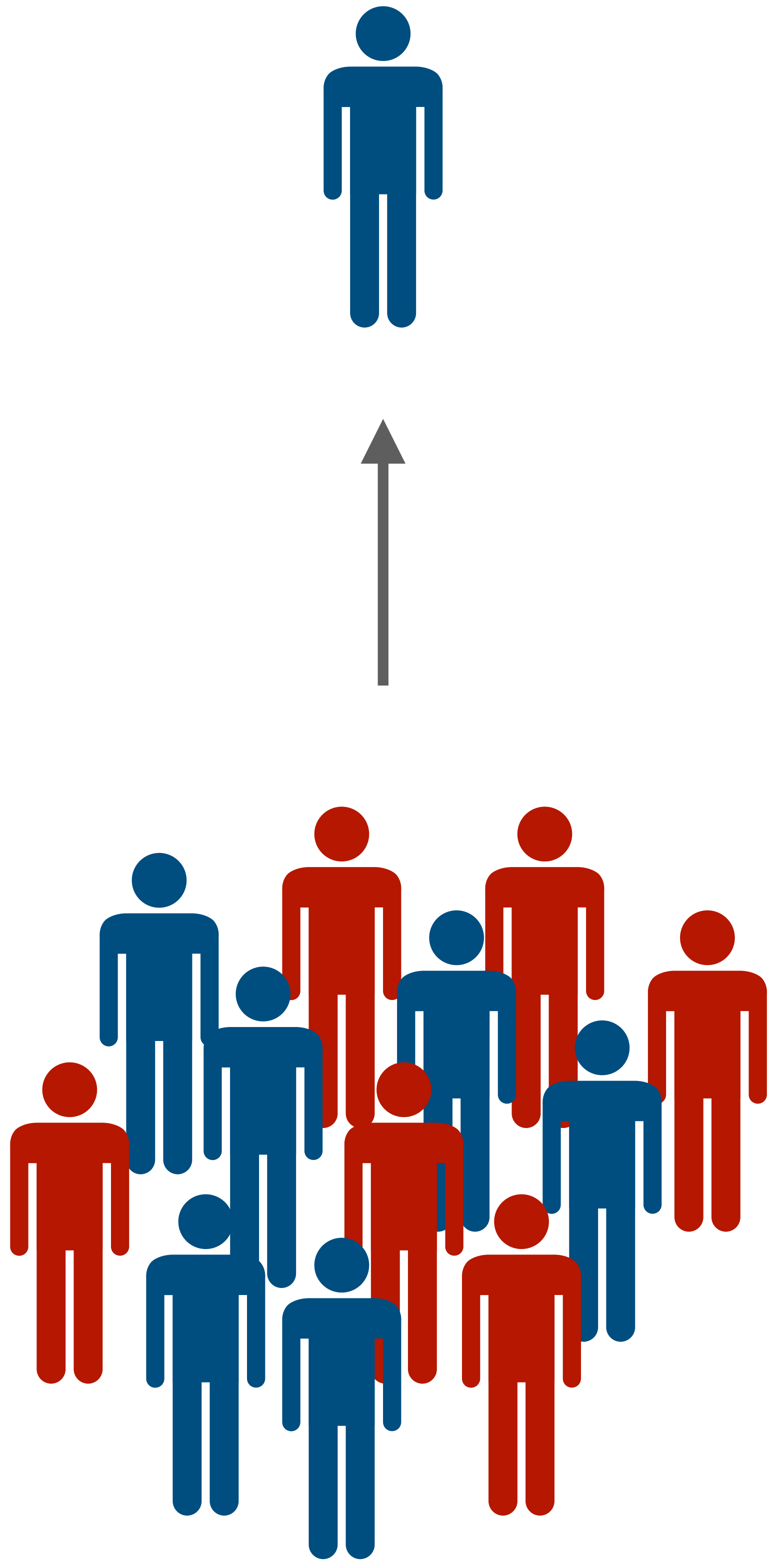


$t = 1$

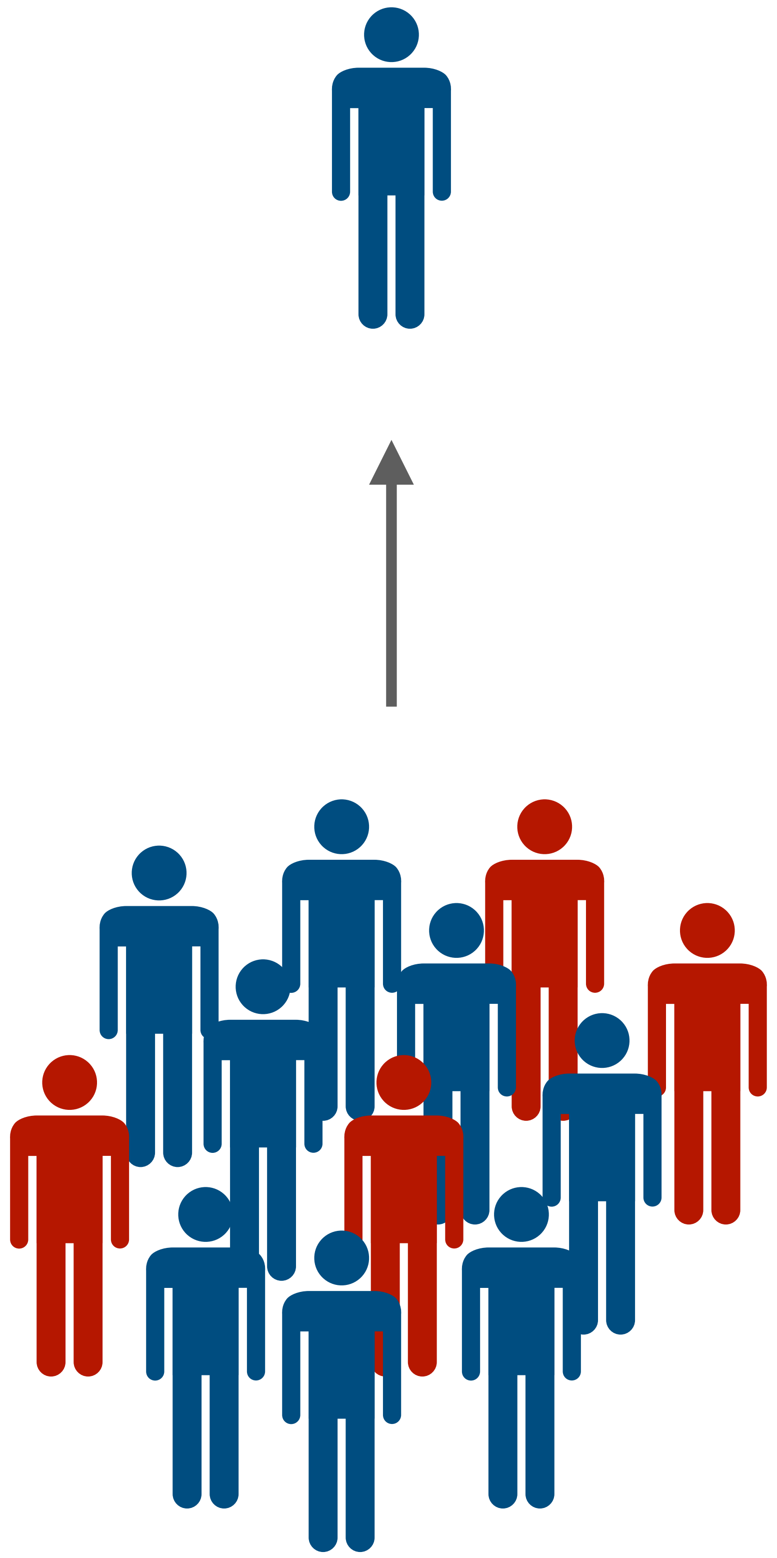


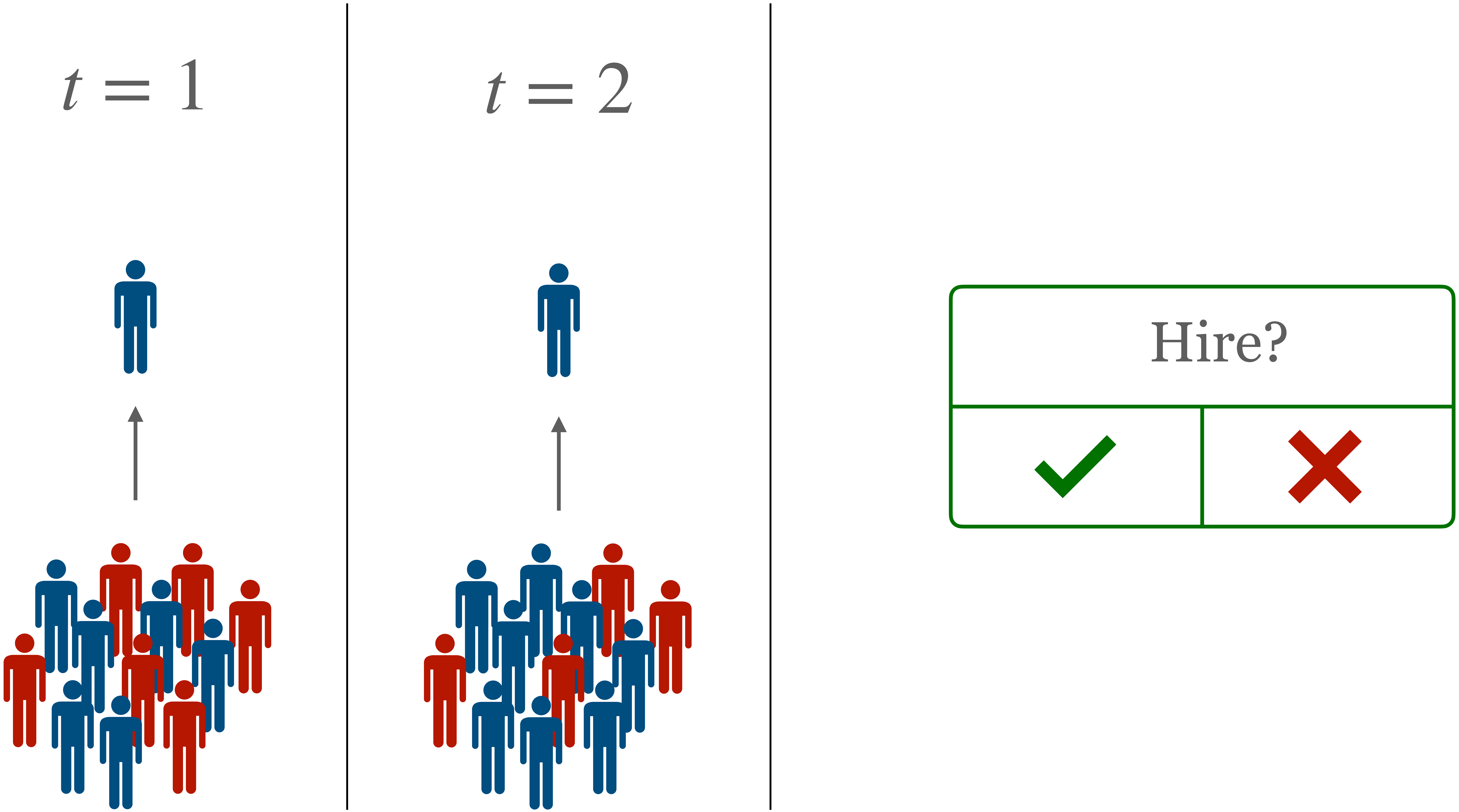


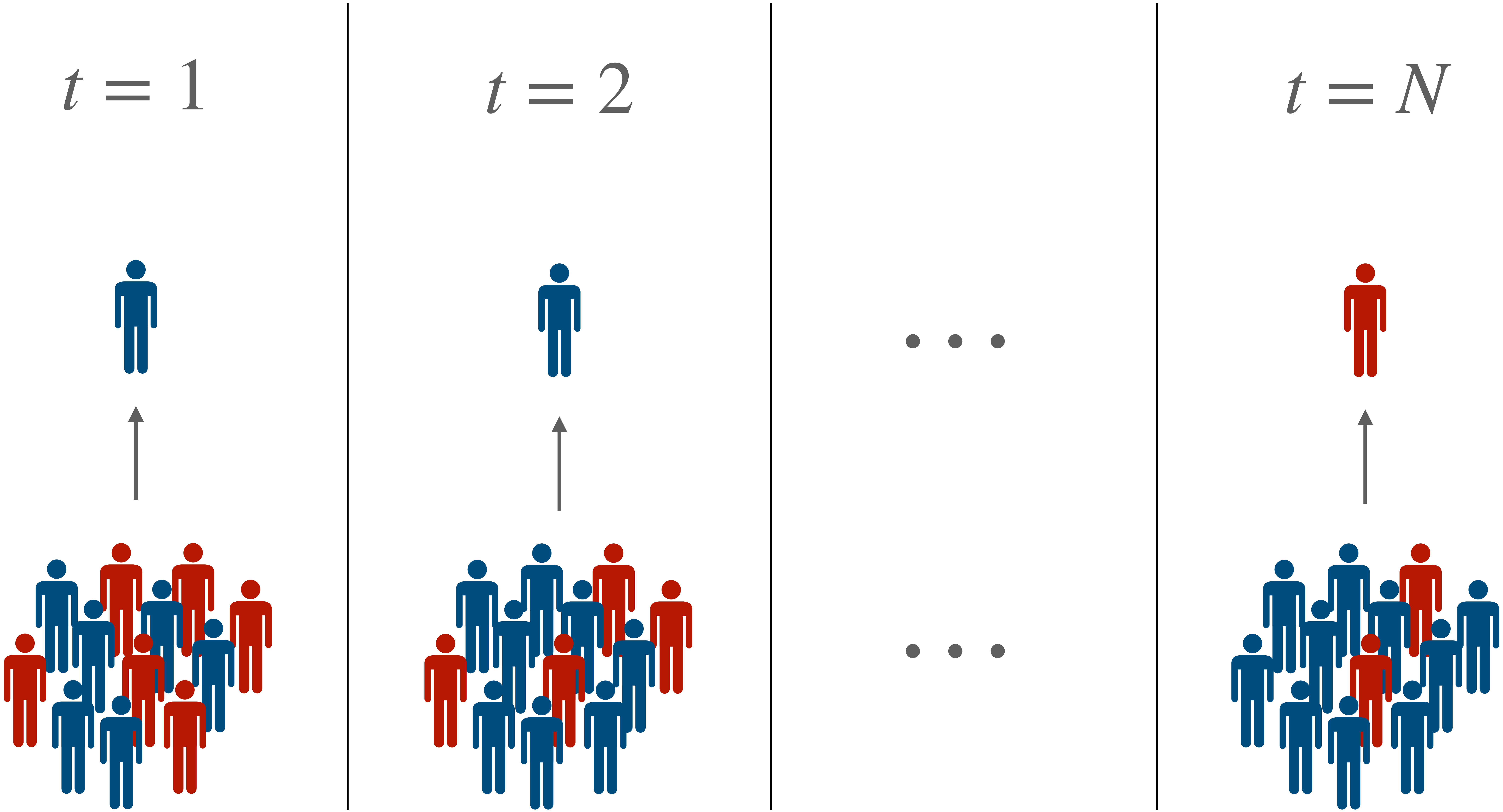
$t = 1$



$t = 2$

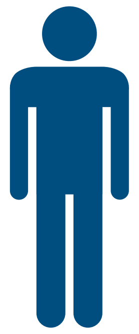
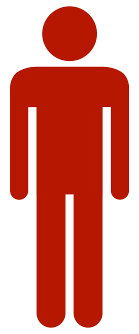



t = 1
t = 2
Hire?

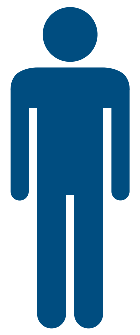
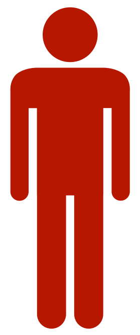

$t = 1$
$t = 2$
$t = N$

When to hire?

Decide on the spot.
To maximise reward and
ensure fairness.

$|\#($  $\wedge 1) - \#($  $\wedge 1)| \leq \varepsilon$

Balanced total acceptance

$|\#(1 \mid$  $) - \#(1 \mid$  $)| \leq \varepsilon$

Balanced acceptance rate

At that time…

…novel algorithmic fairness property/problem. (Alamdari 2024)

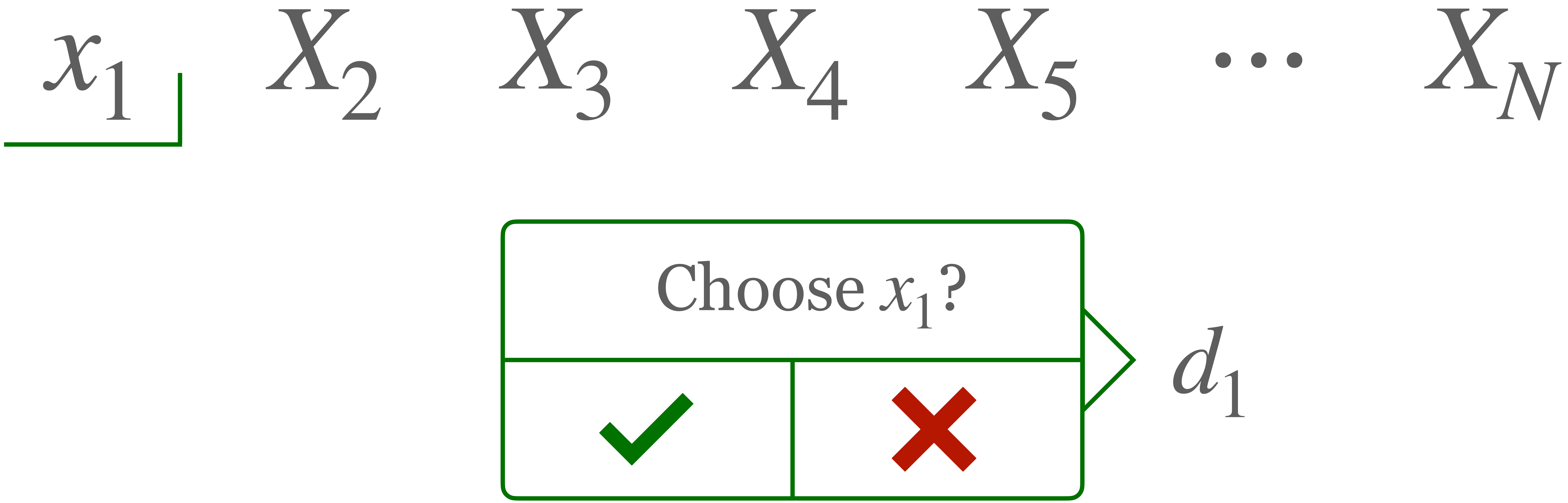
Familiar Problem?

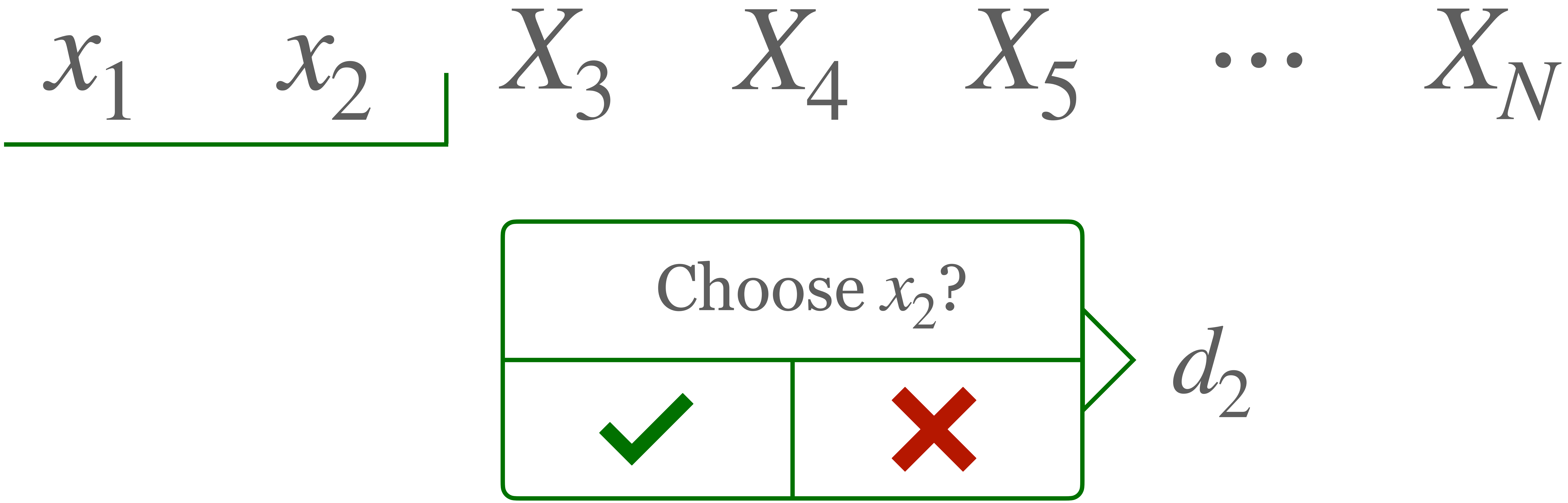
Prophet Inequality.

$$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \quad \cdots \quad X_N$$

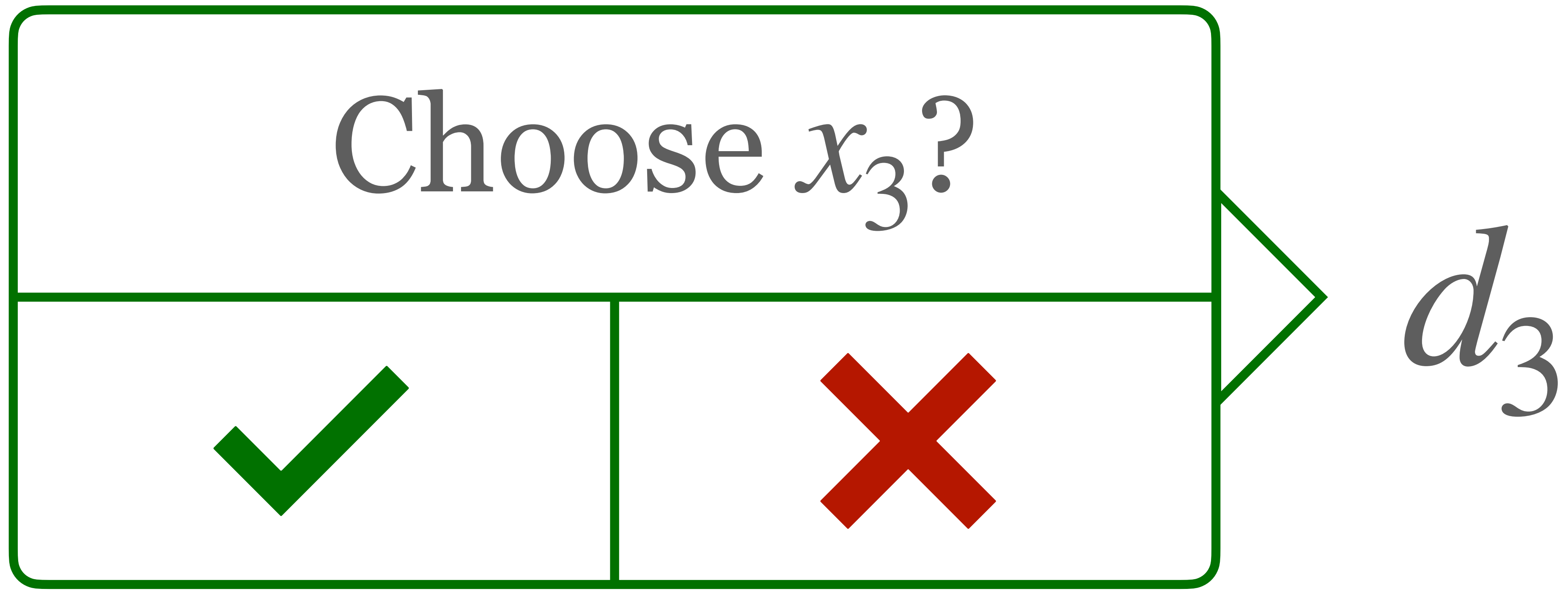
where $\mathscr{D}_1, \ldots, \mathscr{D}_N \in \Delta([K])$

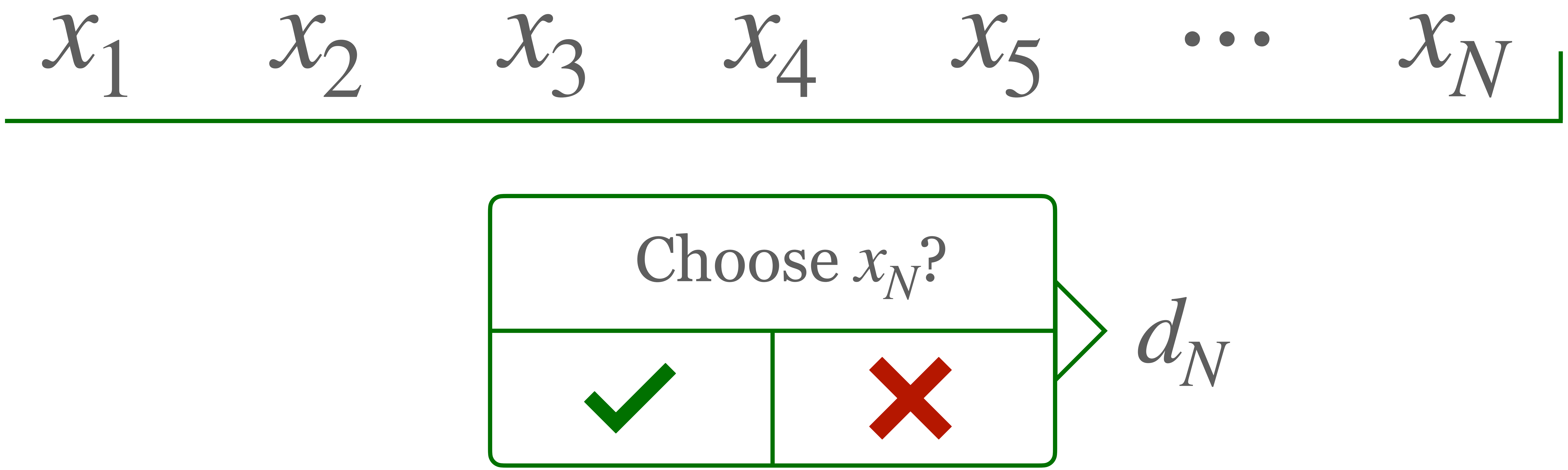
$$x_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \quad \cdots \quad X_N$$


x_1 X_2 X_3 X_4 X_5 $\cdots$ X_N
Choose x_1?
d_1


x_1 x_2 X_3 X_4 X_5 $\cdots$ X_N
Choose x_2?
d_2

$$x_1 \quad x_2 \quad x_3 \quad X_4 \quad X_5 \quad \cdots \quad X_N$$





x_1 x_2 x_3 x_4 x_5 $\cdots$ x_N
Choose x_N?
d_N

$$\max_{\vec{d}} \mathbb{E}\left(\sum_{i=1}^{N} X_i \cdot d_i\right)$$

Objective

$$\max_{\vec{d}} \mathbb{E}\left(\sum_{i=1}^{N} X_i \cdot d_i\right) \text{ s.t. } \sum_{i=1}^{N} d_i \leq 1$$

Problem

$$\max_{\vec{d}} \mathbb{E}\left(\sum_{i=1}^{N} X_i \cdot d_i\right) \text{ s.t. } \sum_{i=1}^{N} d_i \leq k$$

Problem

$$\max_{\vec{d}} \mathbb{E}\left(\sum_{i=1}^{N} X_i \cdot d_i\right) \text{ s.t. } \vec{d} \in \Gamma$$

Problem

Not Covered…

…by existing works:

dependent distributions and complex constraint.

(Kleinberg 2019)

Formal Problem.

What are we looking at?

States

$$\theta : (\mathscr{X} \times \mathscr{Y})^* \rightarrow \Delta(\mathscr{X})$$

Environment

Actions

$$\theta : (\mathscr{X} \times \mathscr{Y})^* \to \Delta(\mathscr{X})$$

Environment

$$\pi : (\mathscr{X} \times \mathscr{Y})^* \times \mathscr{X} \times \rightarrow \Delta(\mathscr{Y})$$

Policy

$$\overrightarrow{XY} := (X_t, Y_t)_{t>0}$$

Stochastic process

$$X_t \sim \theta(\overrightarrow{XY}_{t-1})$$

Stochastic process

$$Y_t \sim \pi(\overrightarrow{XY}_{t-1}, X_t)$$

Stochastic process

$$\texttt{rew} : (\mathscr{X} \times \mathscr{Y})^* \to \mathbb{R}$$

Reward function

$$\texttt{cost} : (\mathscr{X} \times \mathscr{Y})^* \rightarrow \{0,1\}$$

Cost function

Problem Statement:

Given the problem instance $(\mathscr{X}, \mathscr{Y}, \theta, \texttt{rew}, \texttt{cost}, N)$.

Problem Statement:

Given the problem instance $(\mathcal{X}, \mathcal{Y}, \theta, \texttt{rew}, \texttt{cost}, N)$.
Find a policy that, maximises the reward and ensure that the cost is 1 *at time* N.

$$\Gamma^N_{\texttt{cost}} := \{\pi \mid \mathbb{P}^N_{\theta,\pi}(\texttt{cost}) = 1\}$$

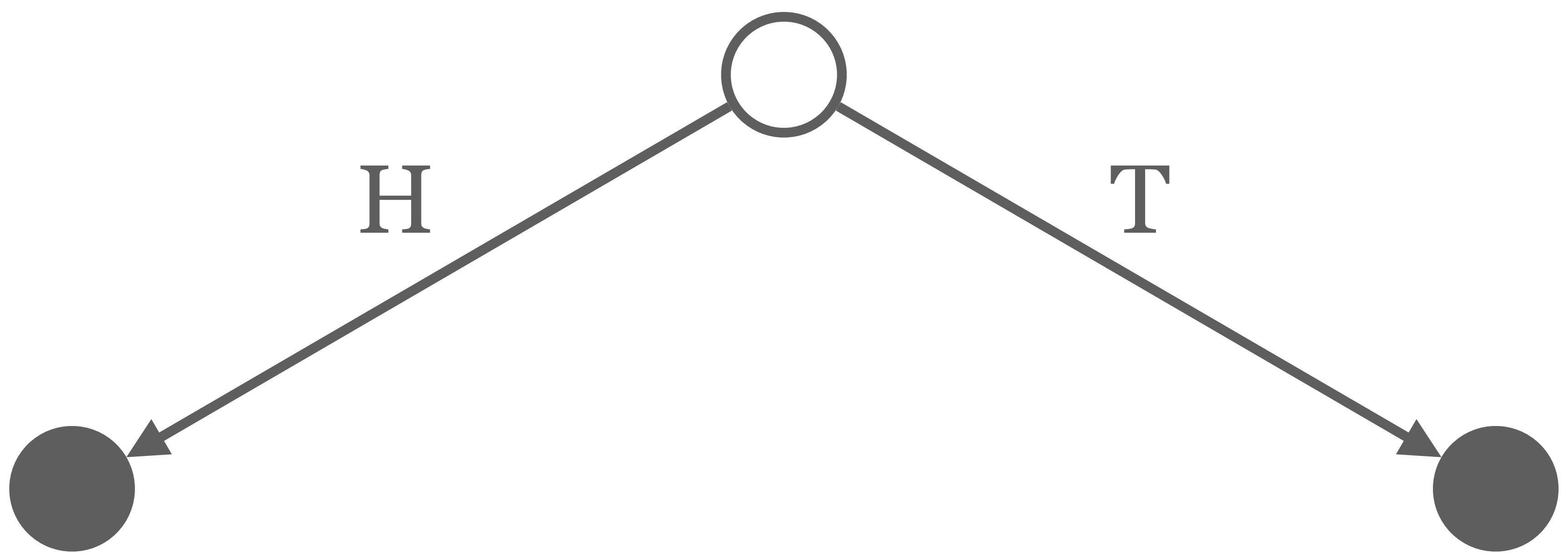
All policies that guarantee that cost is 1 at time N

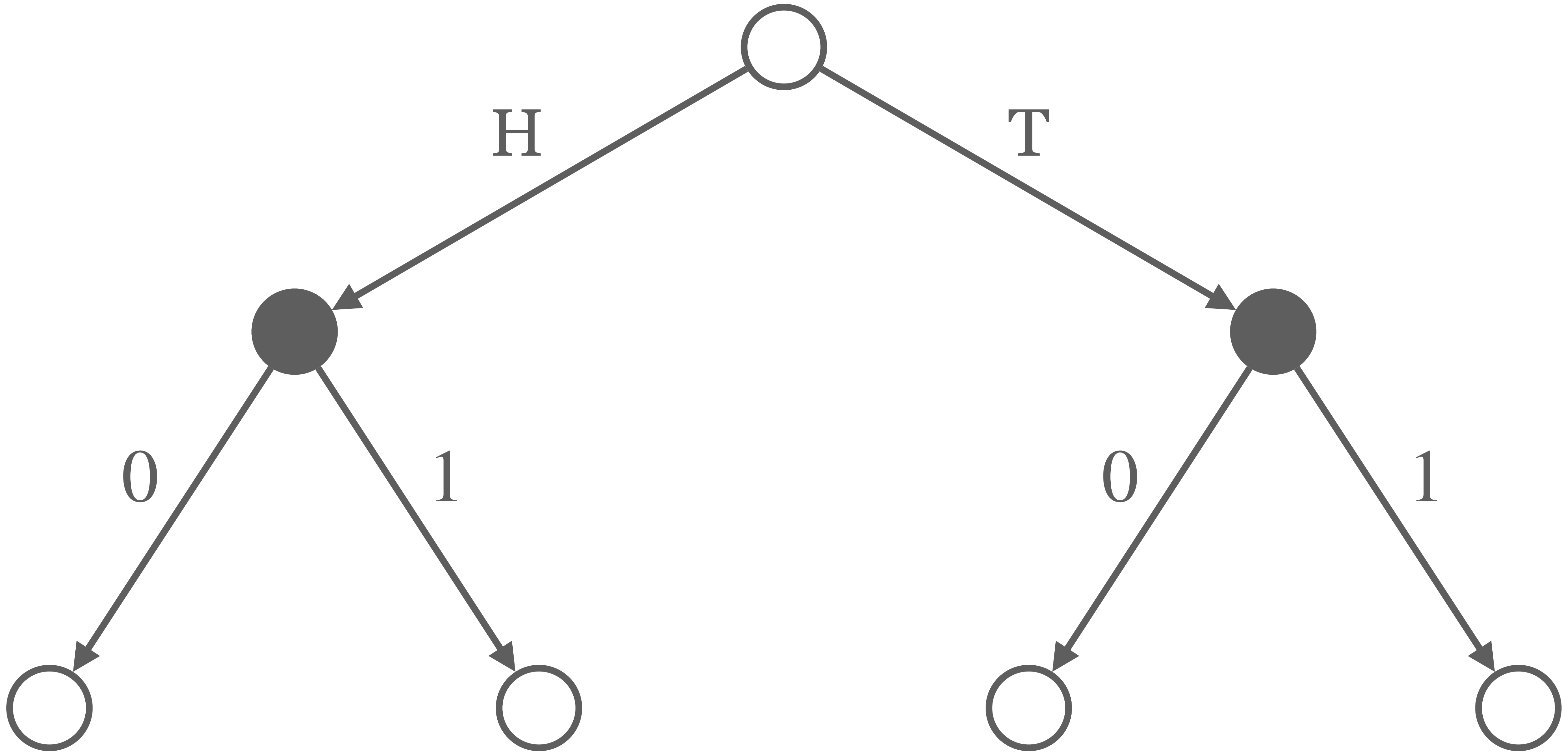
$$\arg\max_{\pi \in \Gamma^N_{\texttt{cost}}} \mathbb{E}^N_{\theta,\pi}(\texttt{rew})$$

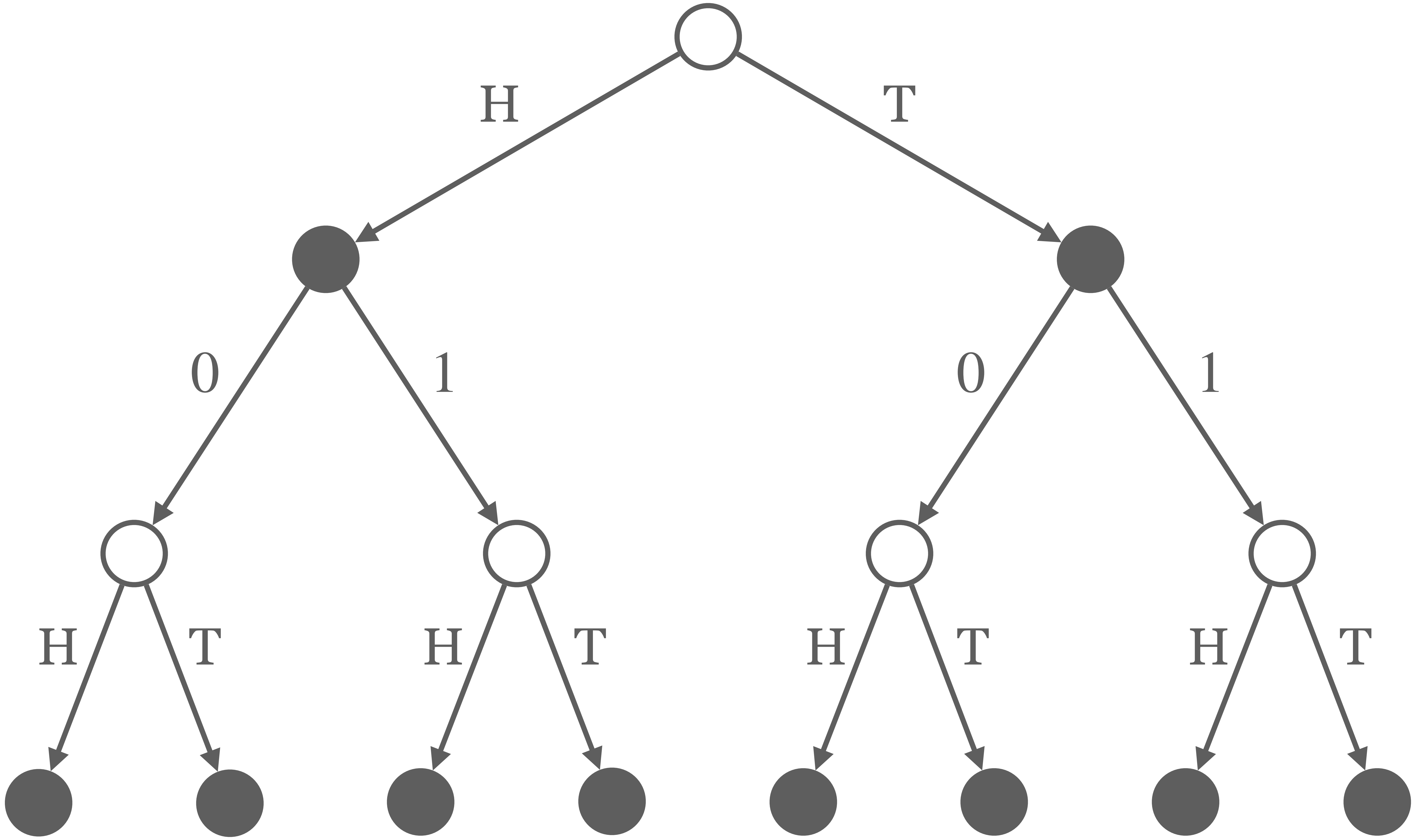
Find reward maximising feasible policy

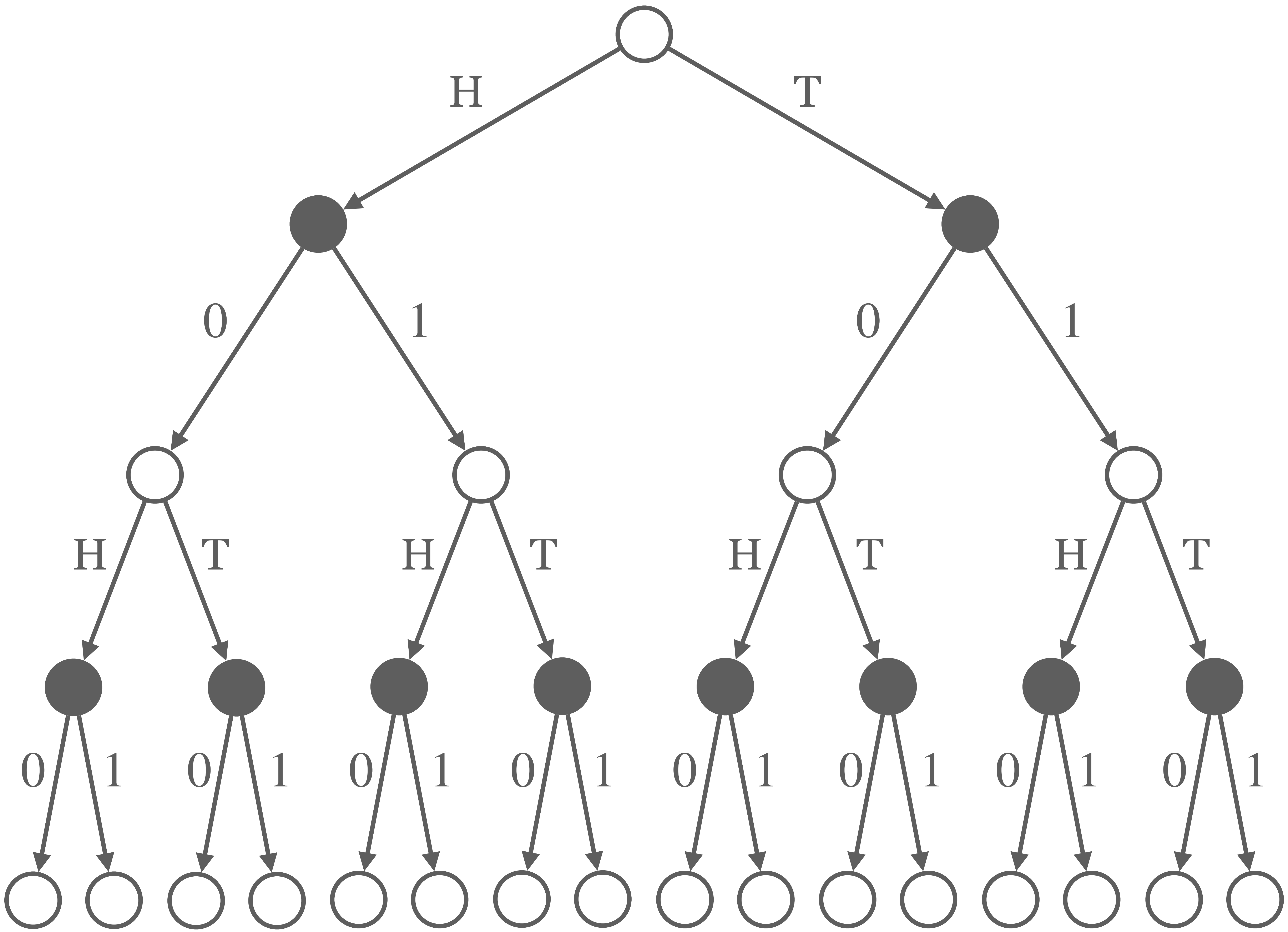
Example.

Coin toss.


H
T


H
T
0
1
0
1


H
T
0
1
0
1
H
T
H
T
H
T
H
T



H
T
0
1
0
1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1

$$\sum_{i=1}^{N} d_i$$

Reward function:
Number of accepted tosses

$$|\#(H \wedge 1) - \#(T \wedge 1)| \leq 1$$

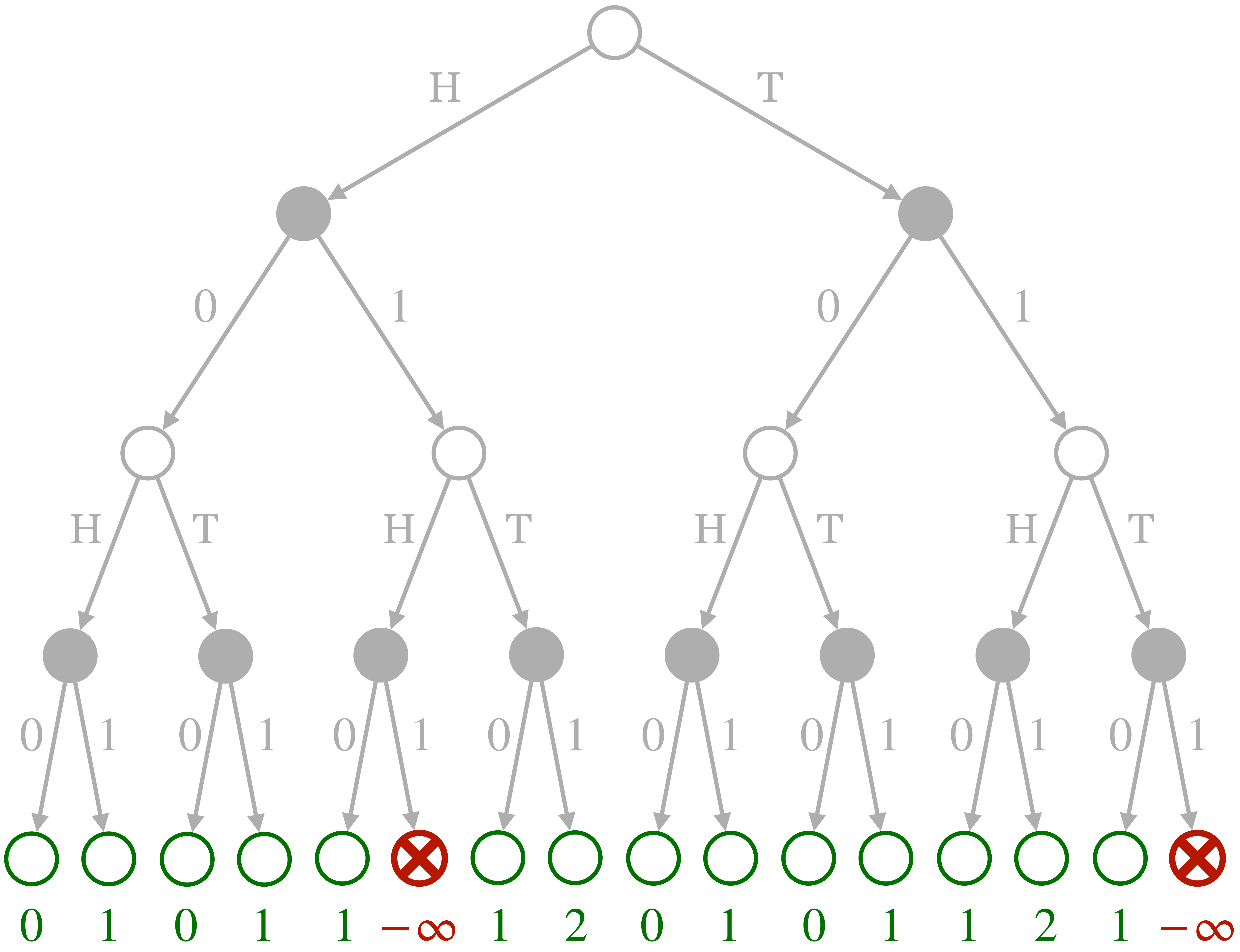
Cost function:
Balanced total acceptance

Algorithm.

Dynamic programming.

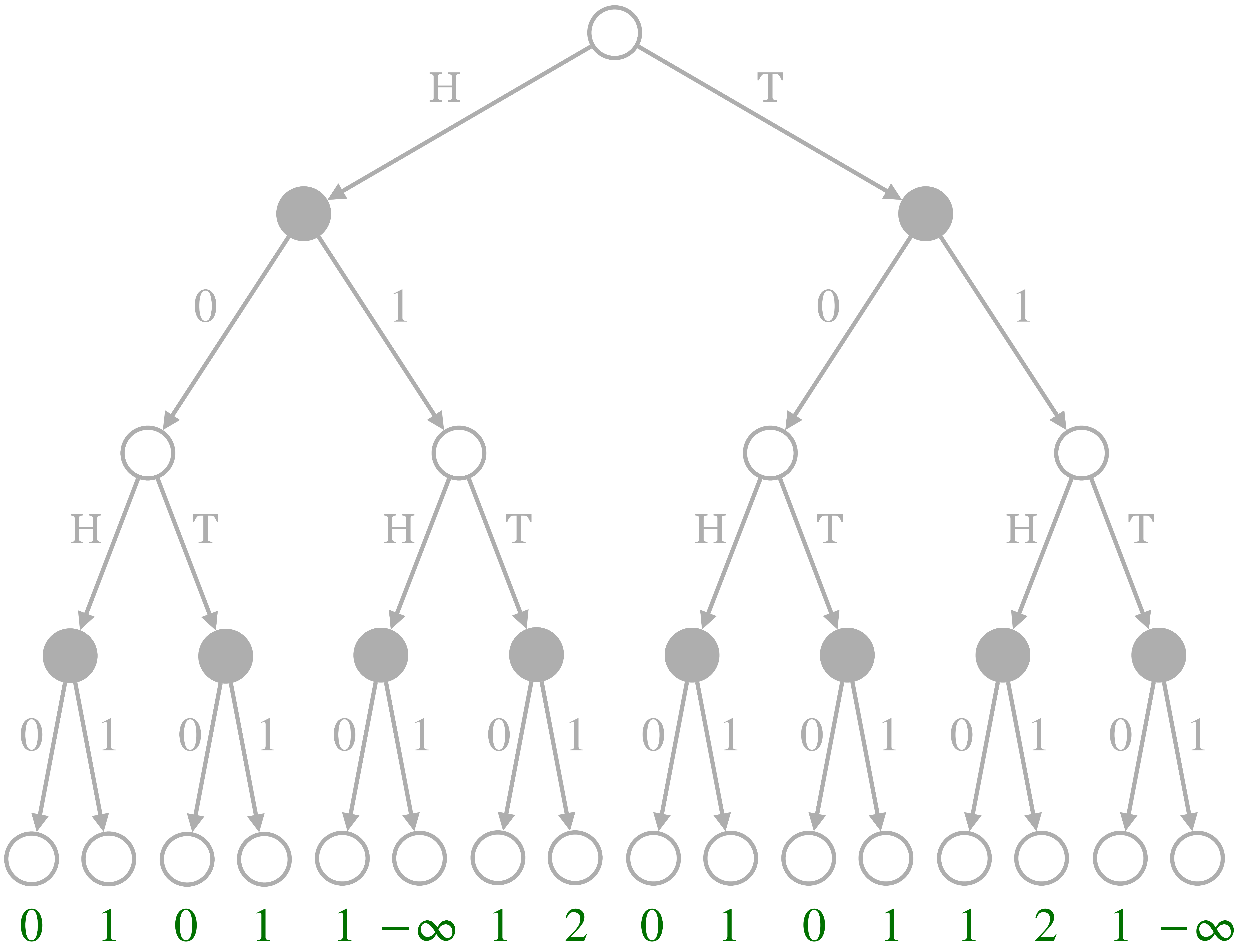
Step 1.

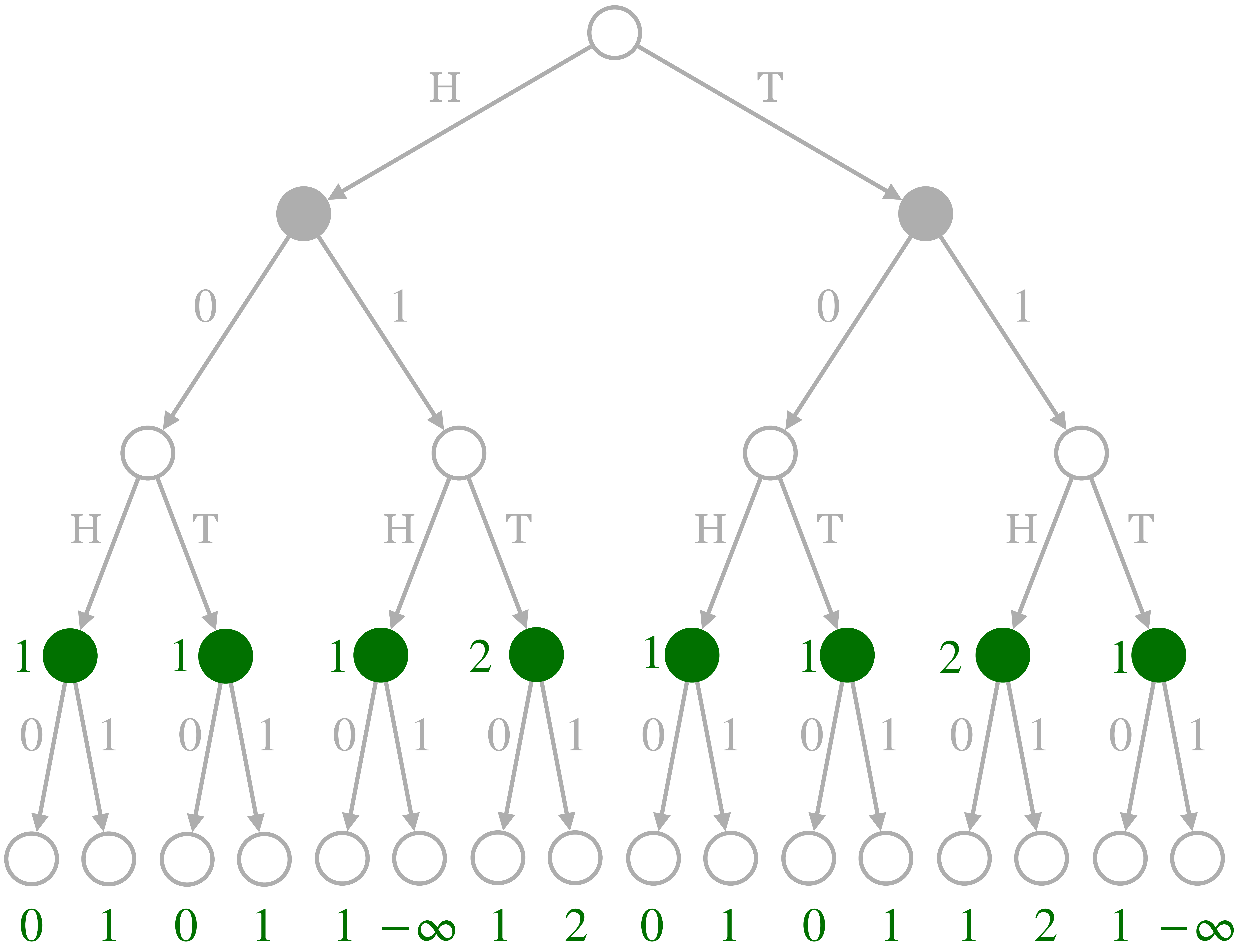
Compute cost and reward.


H
T
0
1
0
1
H
T
H
T
H
T
H
T
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 1 −∞ 1 2 0 1 0 1 1 2 1 −∞

Step 2.

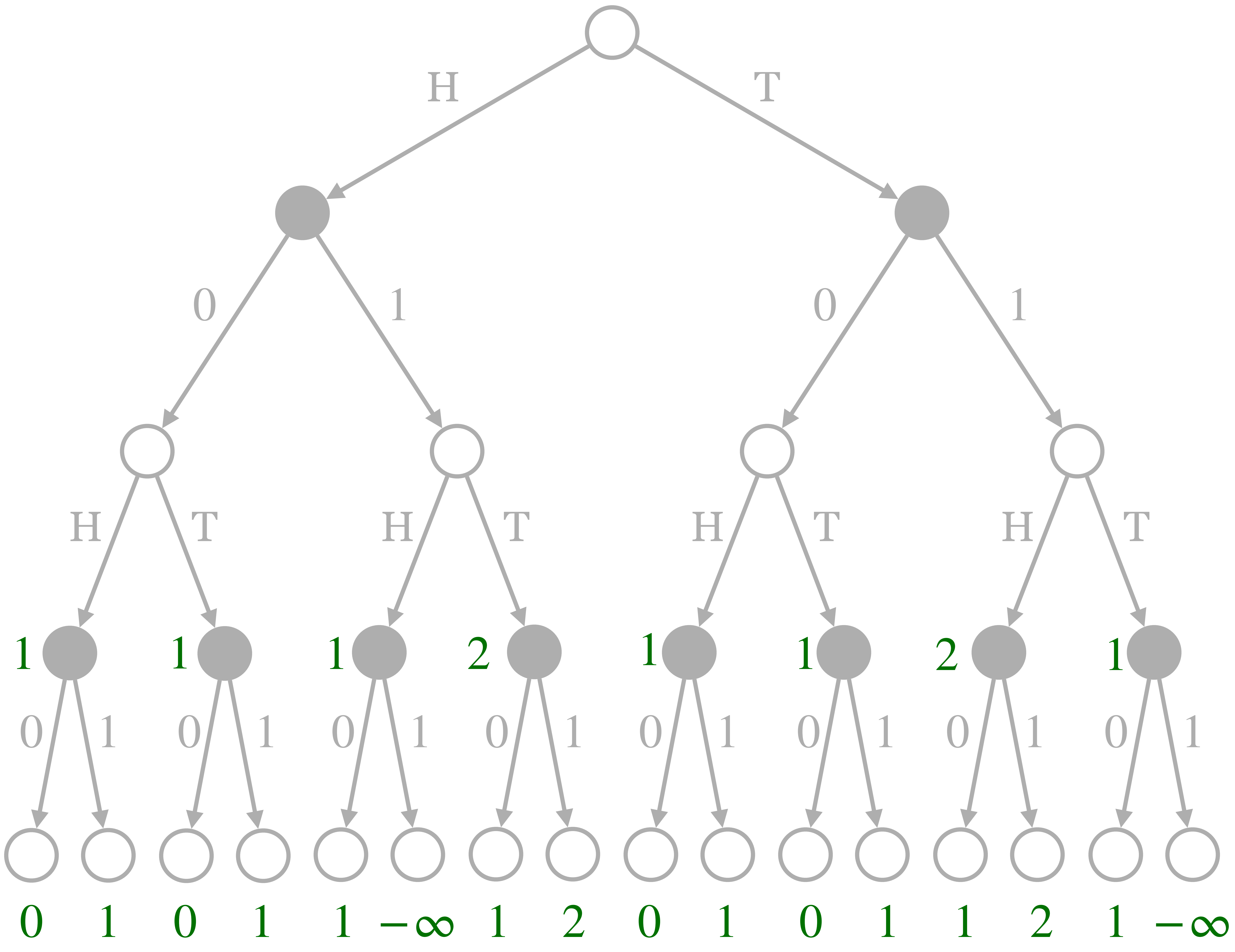
Compute max.

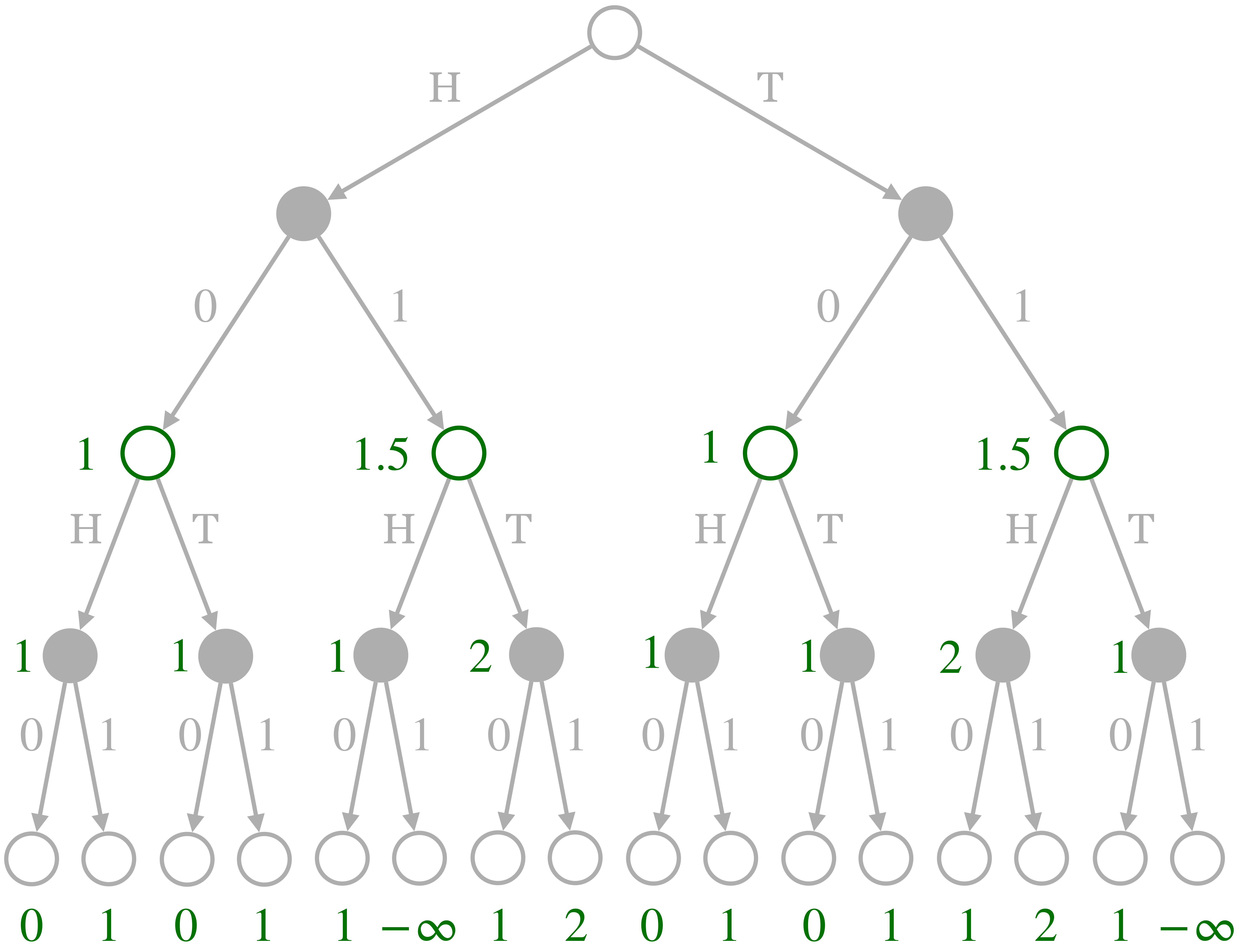

H
T
0
1
0
1
H T H T H T H T
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 1 −∞ 1 2 0 1 0 1 1 2 1 −∞


H
T
0
1
0
1
H
T
H
T
H
T
H
T
1
1
1
2
1
1
2
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
1
−∞
1
2
0
1
0
1
1
2
1
−∞

Step 3.

Compute expectation.

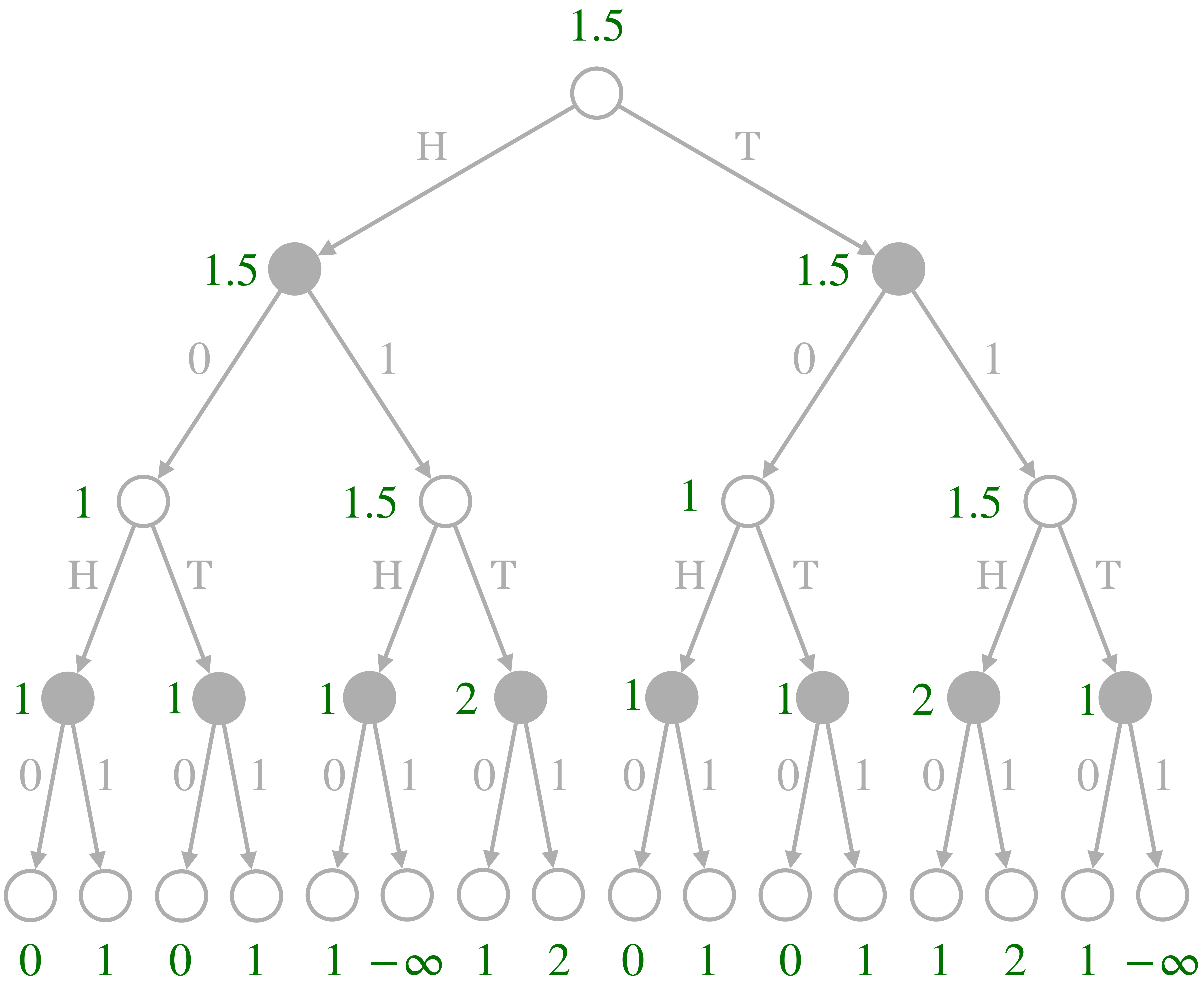

H
T
0
1
0
1
H
T
H
T
H
T
H
T
1
1
1
2
1
1
2
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0 1 0 1 1 −∞ 1 2 0 1 0 1 1 2 1 −∞



H
T
0
1
0
1
1
1.5
1
1.5
H
T
H
T
H
T
H
T
1
1
1
2
1
1
2
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
1
−∞
1
2
0
1
0
1
1
2
1
−∞

Step 4.

Repeat.


1.5
H
T
1.5
1.5
0
1
0
1
1
1.5
1
1.5
H
T
H
T
H
T
H
T
1
1
1
2
1
1
2
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
1
−∞
1
2
0
1
0
1
1
2
1
−∞

Complexity

PSPACE-hard.
(Papadimitriou 1985)

Observation.

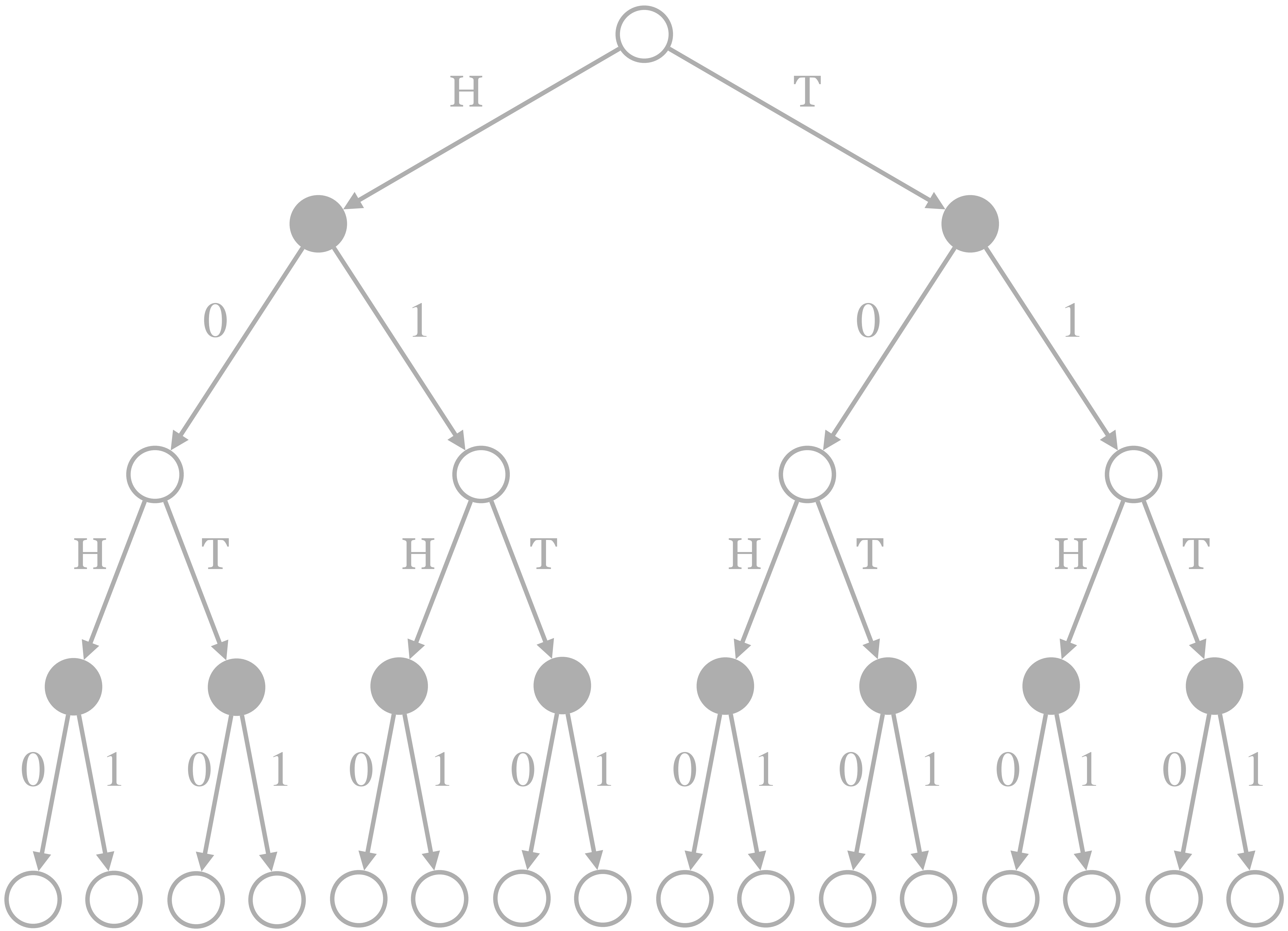
The cost function is not really a function over the entire history.

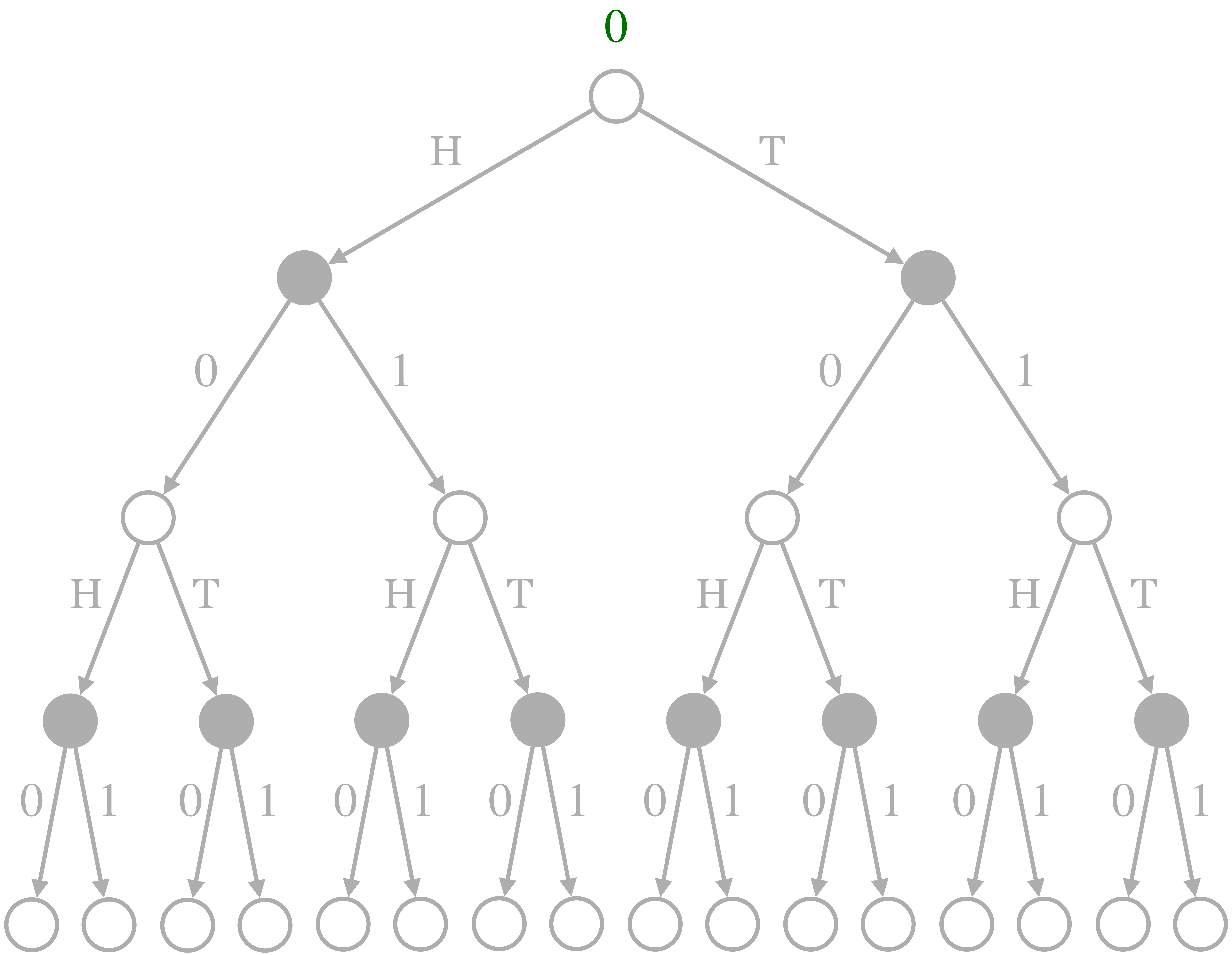
Statistic.

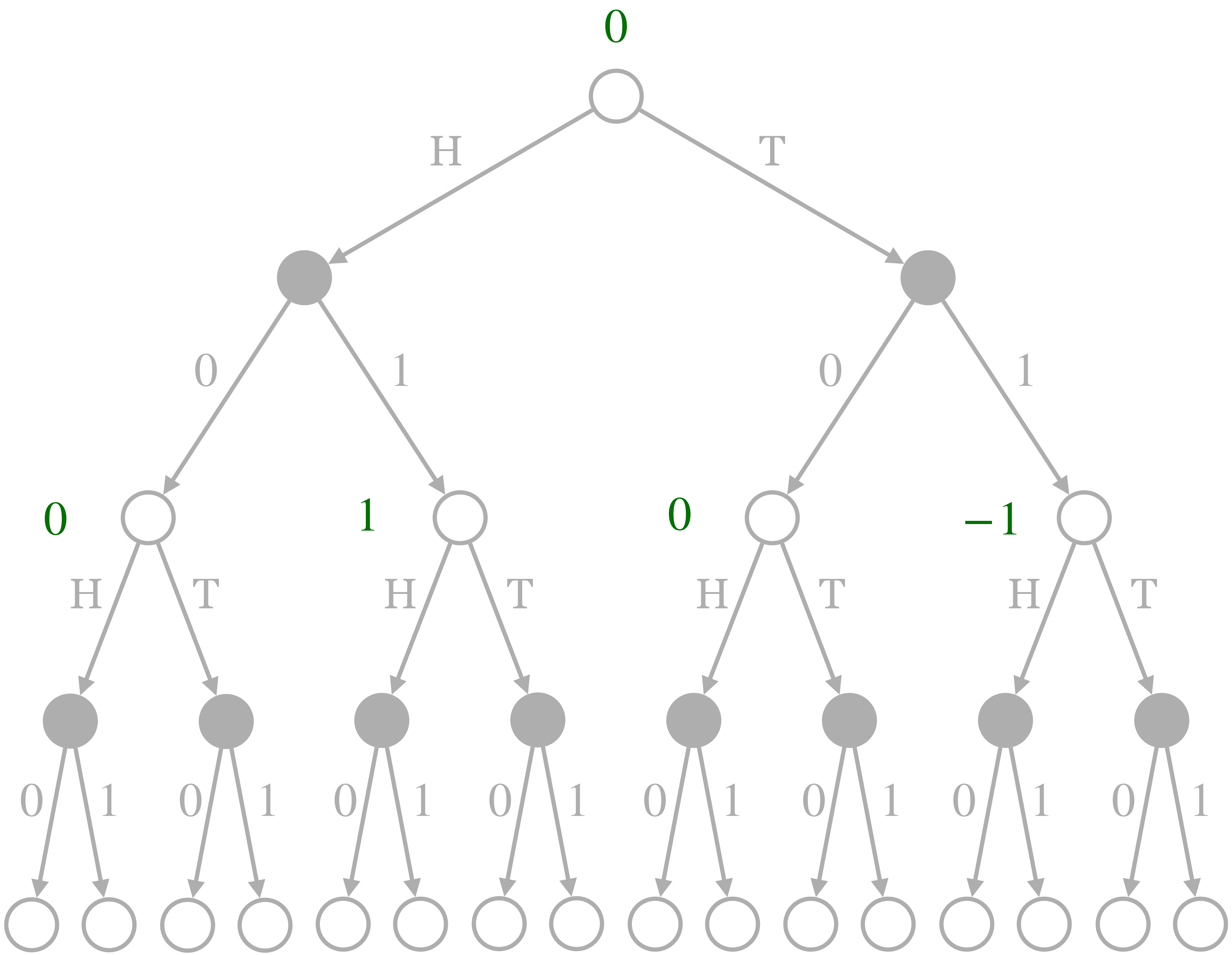
Is there a smaller representation of the history?

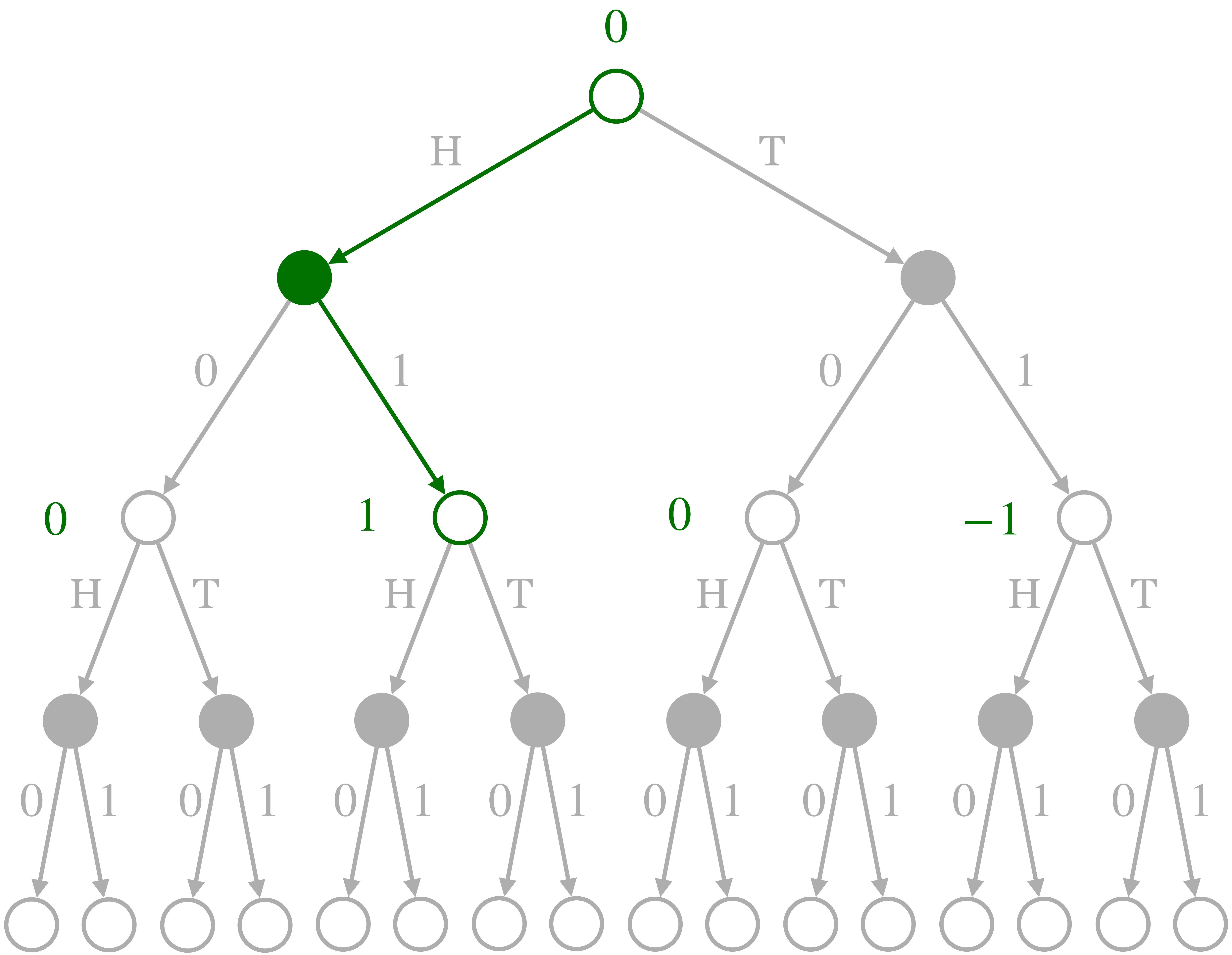
$$\#(H \wedge 1) - \#(T \wedge 1)$$

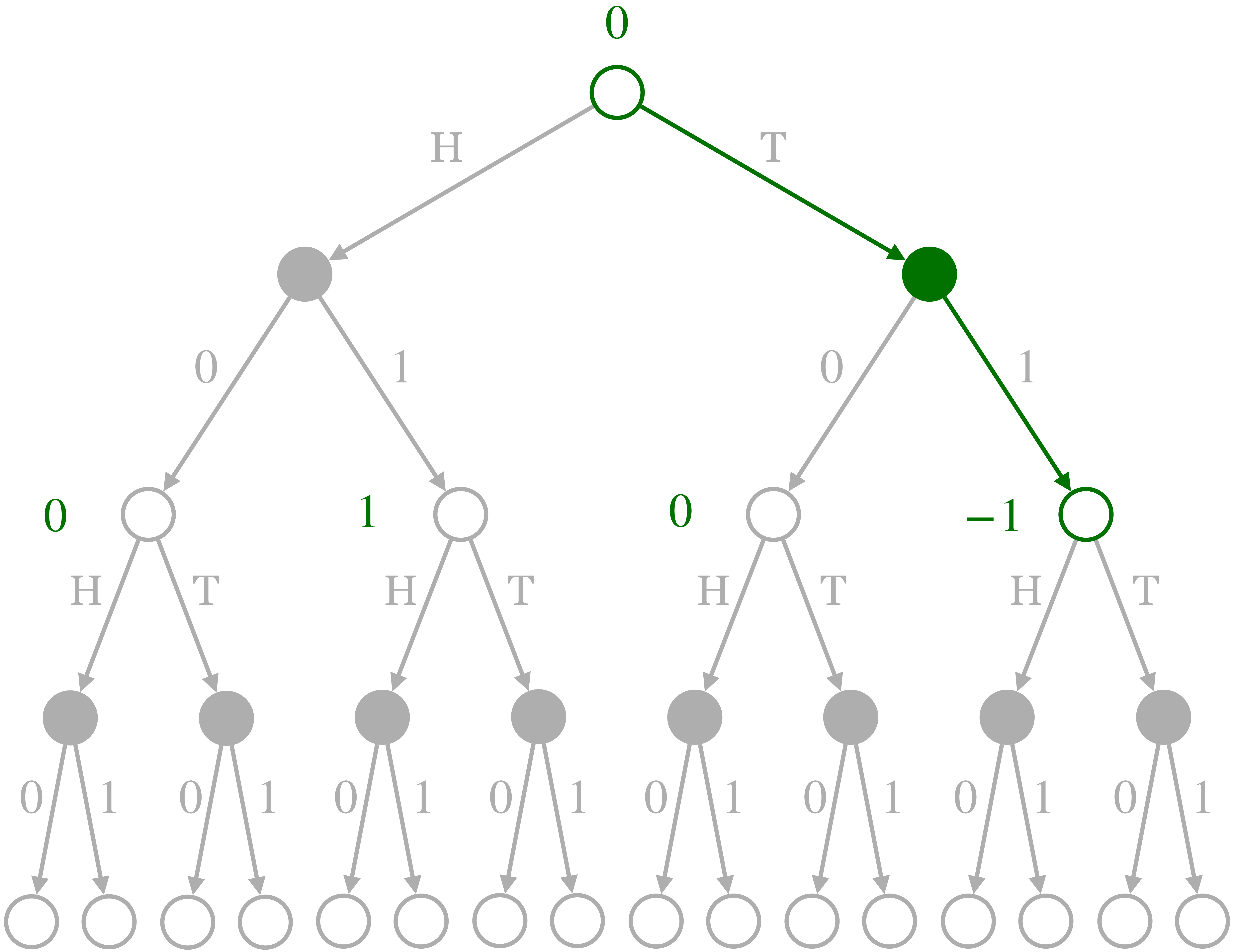
Statistic:
Difference between accepted H and T

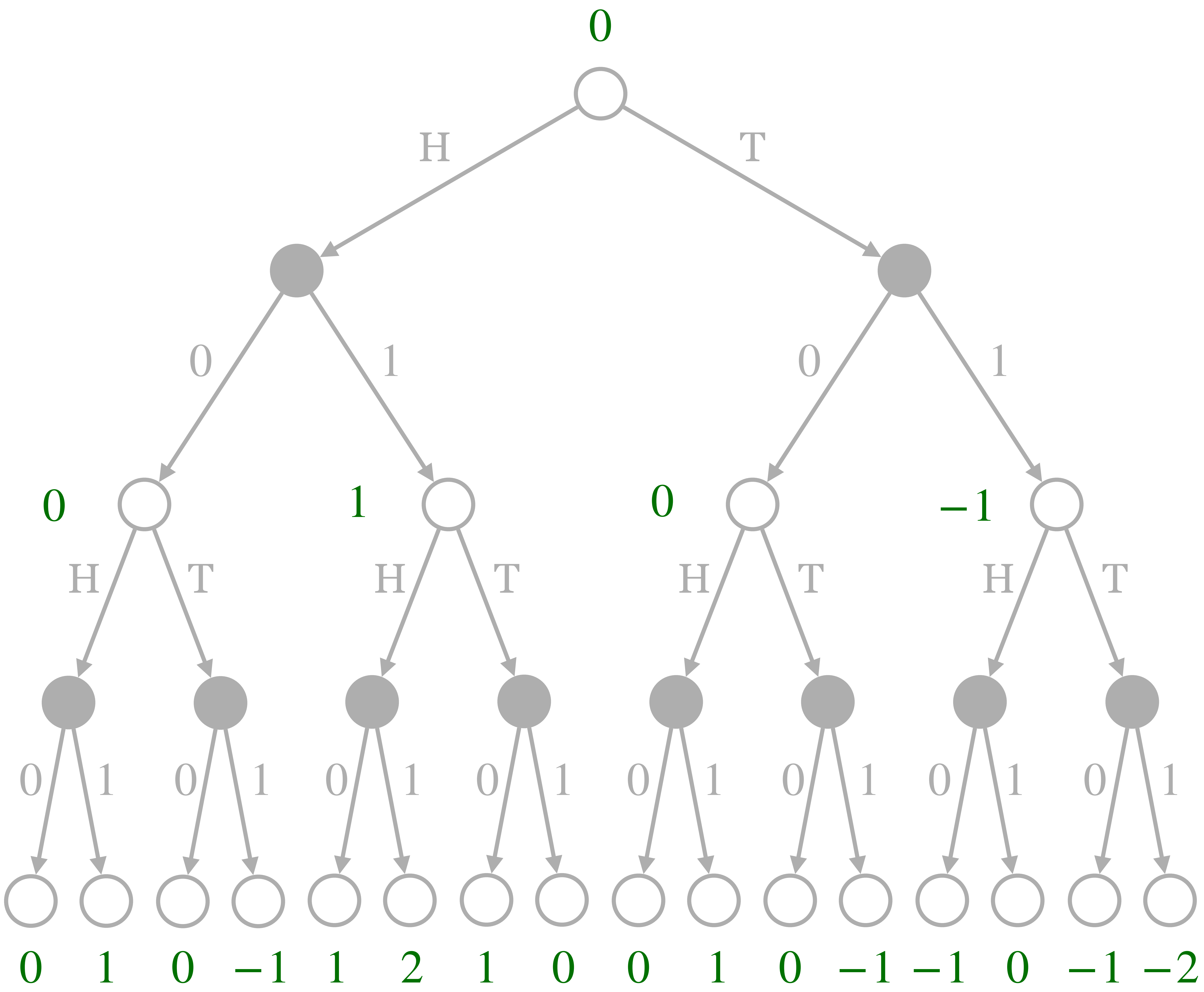

H
T
0
1
0
1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1


0
H
T
0
1
0
1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1


0
H
T
0
1
0
1
0
1
0
−1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1

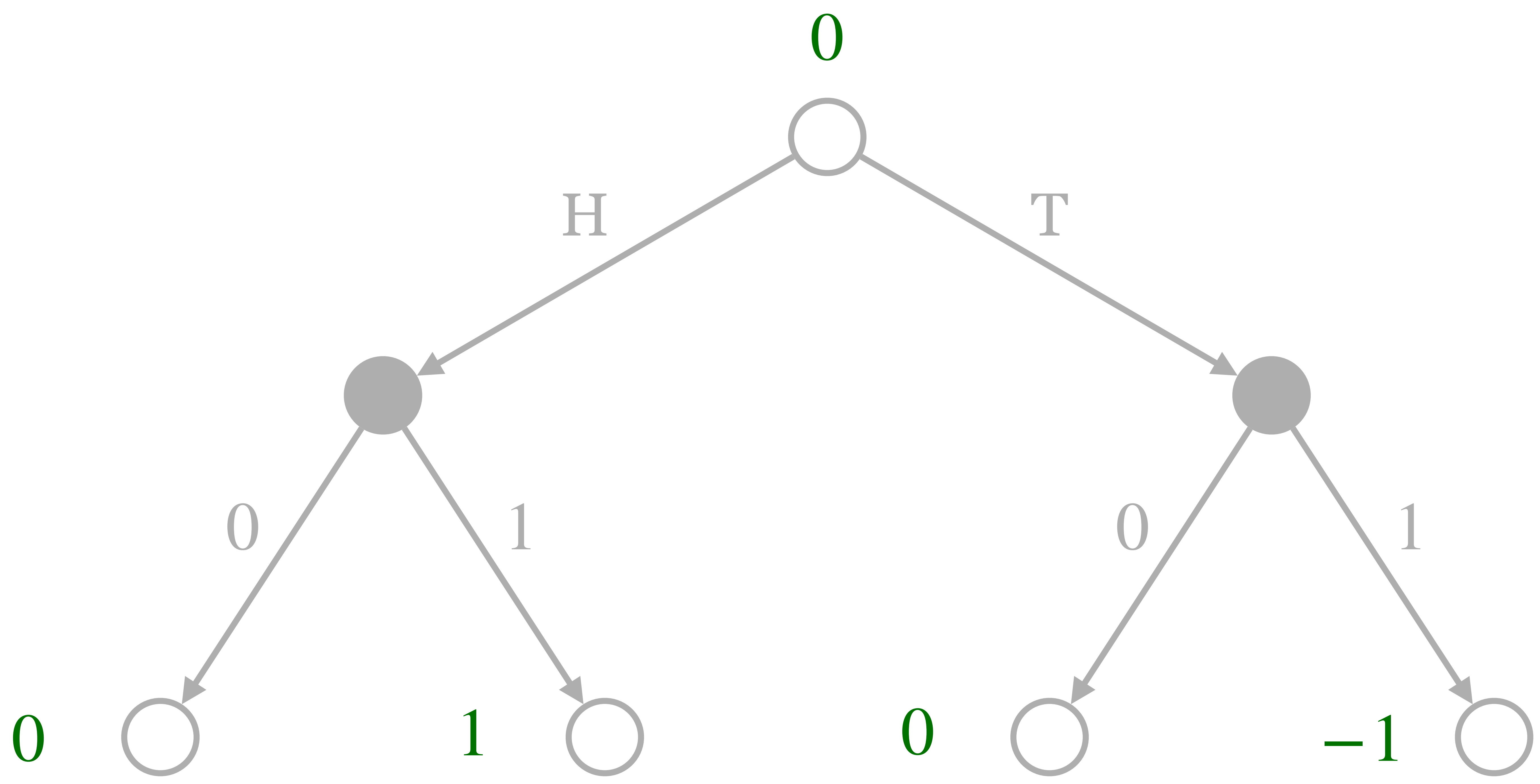

0
H
T
0
1
0
1
0
1
0
-1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1

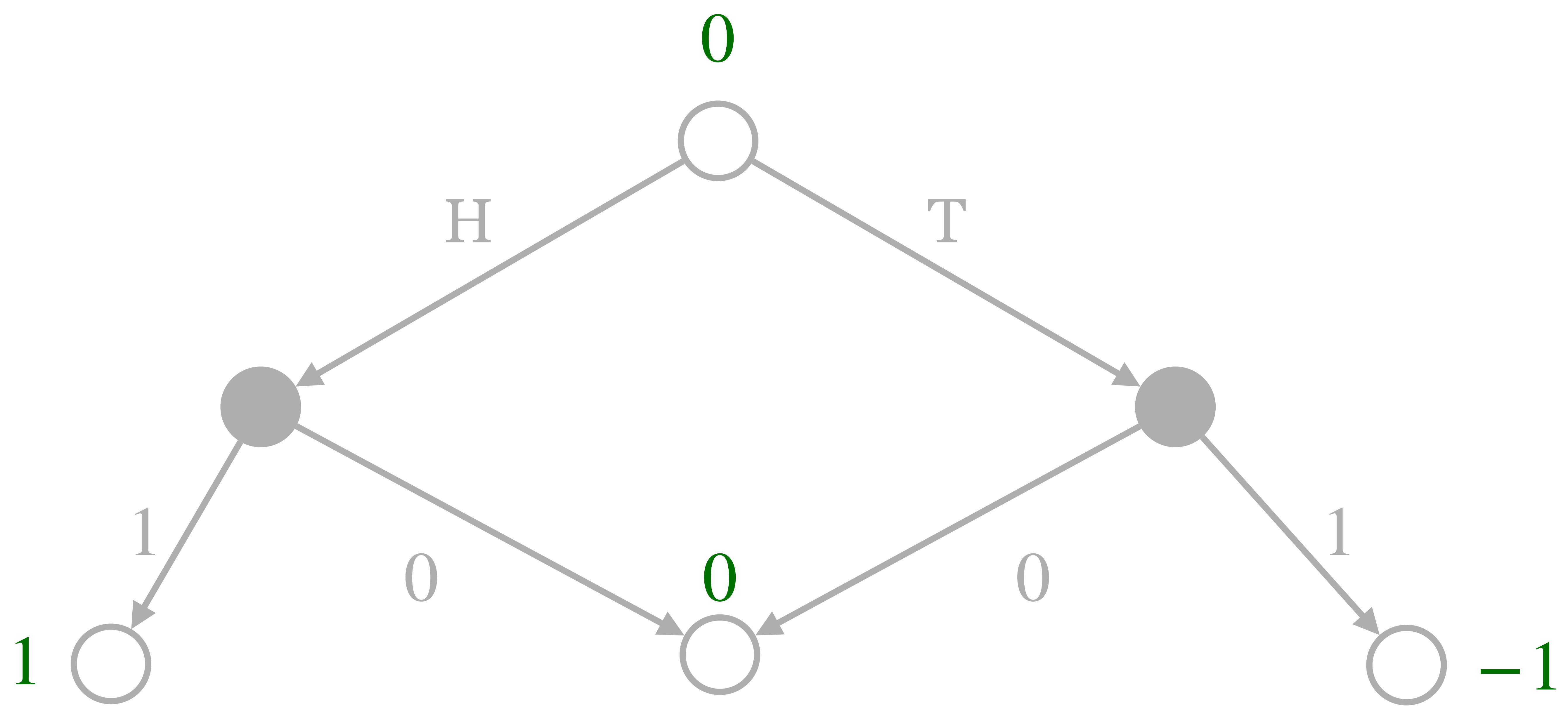

0
H
T
0
1
0
1
0
1
0
−1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1

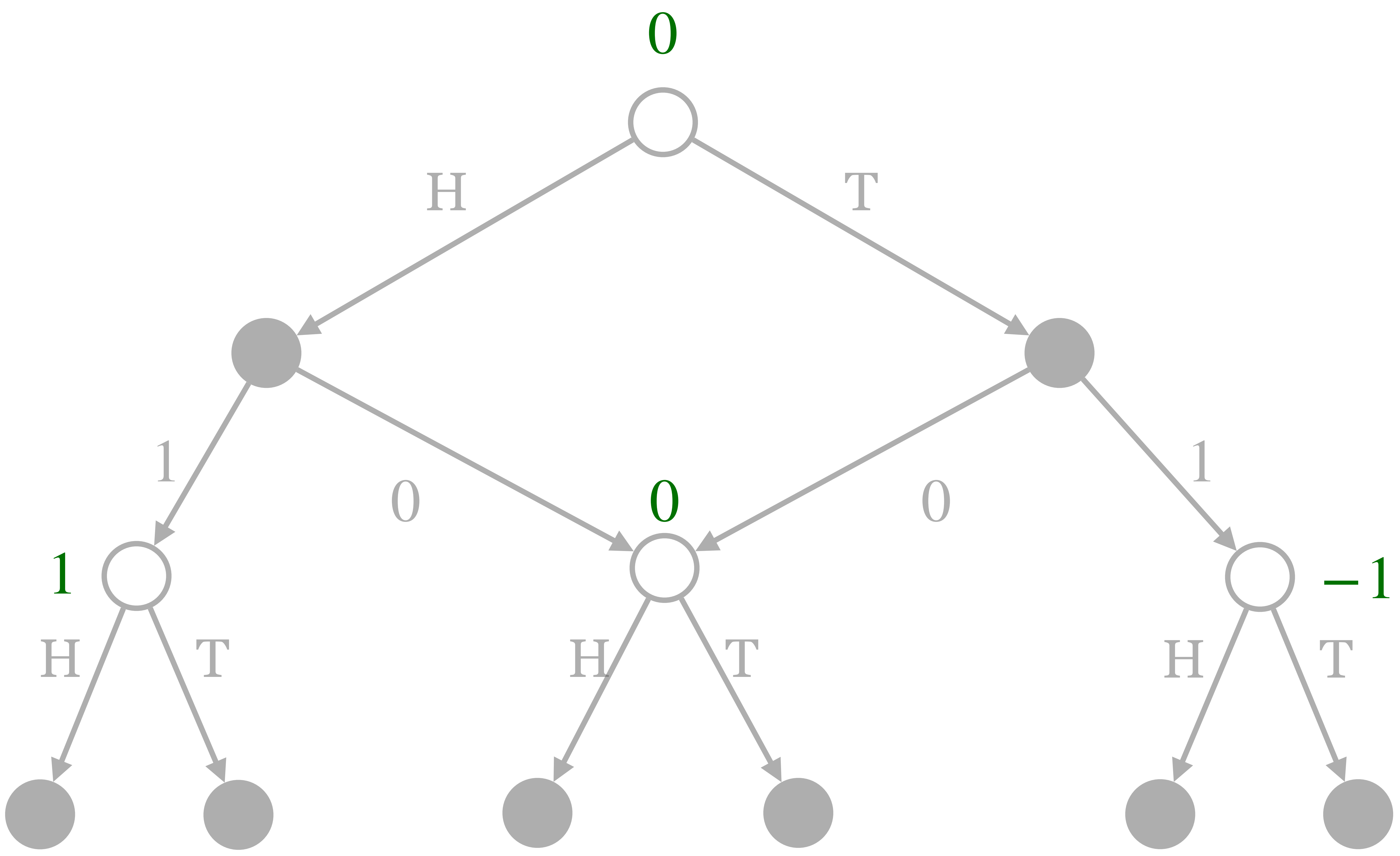

0
H
T
0
1
0
1
0
1
0
−1
H
T
H
T
H
T
H
T
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0 1 0 −1 1 2 1 0 0 1 0 −1 −1 0 −1 −2

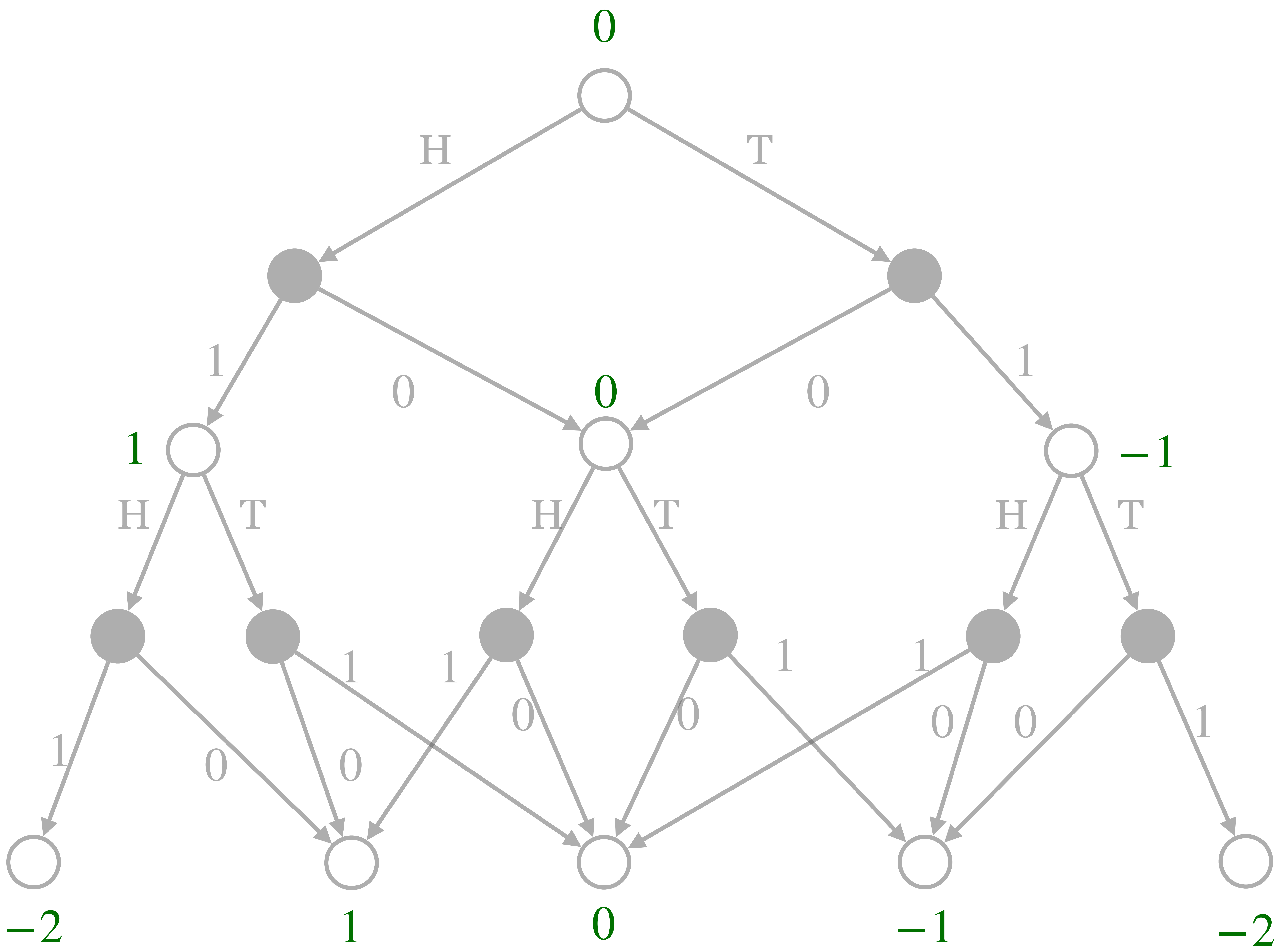
Now reduce…

…by collapsing equivalent states.


0
H
T
0
1
0
1
0
1
0
−1


0
H
T
1
0
0
0
1
1
−1


0
H
T
1
0
0
0
1
1
−1
H
T
H
T
H
T



0
H
T
1
0
0
0
1
1
−1
H
T
H
T
H
T
1
1
1
1
1
0
0
0
0
0
0
1
0
−2
1
0
−1
−2

Abstraction

From a tree of size $\mathcal{O}(2^{2N})$
to a DAG of size $\mathcal{O}\left(N^2\right)$.

Statistical Properties.

Let's be more general.

$$\mu : \mathscr{W}^* \rightarrow \mathscr{S}$$

Statistic

Well-Behaved.

Abstract the history.

$\forall \vec{u}, \vec{v}, \vec{w} \in \mathscr{W}^* :$

$$\mu(\vec{u}) = \mu(\vec{v}) \implies \mu(\vec{u}\vec{w}) = \mu(\vec{v}\vec{w})$$

Well-behaved

$$\left| \sum_{i=1}^{N} X_i - \sum_{i=1}^{N} (1 - X_i) \right|$$

Absolute Balance

$$100 \xrightarrow{\mu_{\text{bal}}} 1$$

$$110 \xrightarrow{\mu_{\text{bal}}} 1$$

$$1001 \xrightarrow{\mu_{\text{bal}}} 0$$

$$1101 \xrightarrow{\mu_{\text{bal}}} 2$$

$$10 \xrightarrow{\mu_{\mathrm{avr}}} 1/2$$

$$0101 \xrightarrow{\mu_{\mathrm{avr}}} 1/2$$

$$1011 \xrightarrow{\mu_{\text{avr}}} 3/4$$

$$010111 \xrightarrow{\mu_{\text{avr}}} 2/3$$

Statistical Abstraction.

What do we gain?

Complexity

If θ, `rew`, `cost` *are* μ*-representable*

Representability.

The function $f : \mathcal{W}^* \rightarrow \mathcal{U}$ *is* <u>*μ-representable*</u>, *if there exists* $\hat{f} : \mathcal{S} \rightarrow \mathcal{U}$ *s.t. for every* $\overrightarrow{w} \in \mathcal{W}^*, f(\overrightarrow{w}) = \hat{f}(\mu(\overrightarrow{w}))$.

Complexity

If θ, `rew`, `cost` *are* μ*-representable then*
We require $\mathcal{O}(|\mathcal{X}| \cdot |\mathcal{Y}| \cdot \Sigma_{i=1}^{N} \mathrm{size}_\mu(t))$ *time*
to solve the problem.

Size of a Statistic.

For every $t > 0$ the statistic μ
induces an equivalence relation over $\mathscr{W}^t$.
The size of the statistic $\text{size}_\mu(t)$ is
the number of equivalence classes at time t.

Example.

Acceptance rate H vs. T.

$$\left| \frac{\Sigma_i^N \mathbf{1}[x_i = H] \cdot d_i}{\Sigma_i^N \mathbf{1}[x_i = H]} - \frac{\Sigma_i^N \mathbf{1}[x_i = T] \cdot d_i}{\Sigma_i^N \mathbf{1}[x_i = T]} \right| \leq \varepsilon$$

Balanced acceptance rate

$$\frac{\begin{pmatrix} \Sigma_i^N \mathbf{1}[x_i = H] \cdot d_i \\ \Sigma_i^N \mathbf{1}[x_i = T] \cdot d_i \\ \Sigma_i^N \mathbf{1}[x_i = H] \end{pmatrix}}{\text{Statistic}}$$

$$\mathcal{O}(N^4)$$

Complexity

Specification?

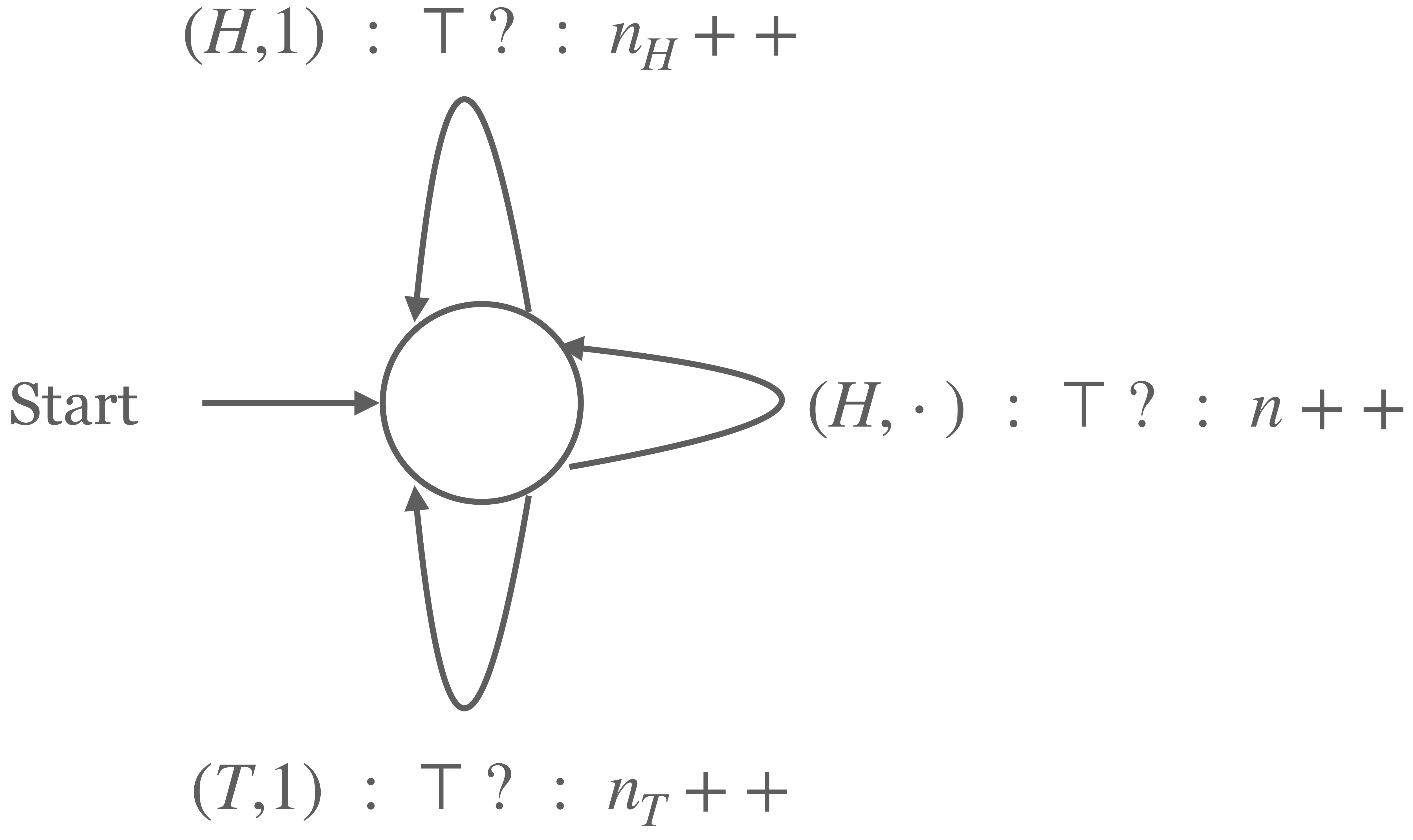
How can we obtain a statistic for a function?
Easy…if its specified by a
counter automaton.

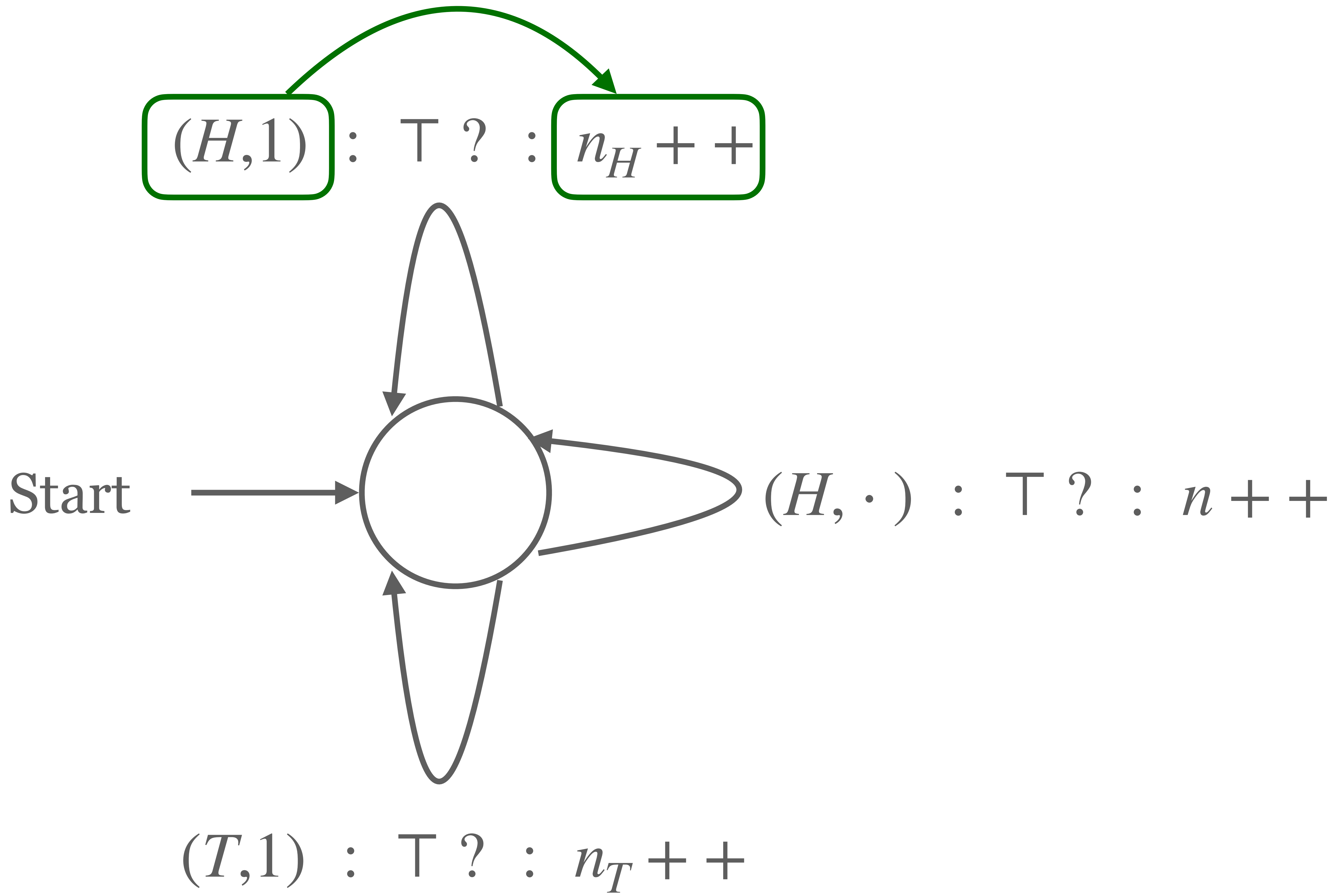
Counter Automata.

DFA + counters + output function.

Example.

Cont.


$(H,1)\ :\ \top\ ?\ :\ n_H{+}{+}$
Start
$(H,\cdot)\ :\ \top\ ?\ :\ n{+}{+}$
$(T,1)\ :\ \top\ ?\ :\ n_T{+}{+}$


(H,1) : ⊤ ? : n_H ++
Start
(H, ·) : ⊤ ? : n ++
(T,1) : ⊤ ? : n_T ++

$$\left| \frac{n_H}{n} - \frac{n_T}{N - n} \right| \leq \varepsilon$$

Output Function

CA to Statistic

$\mu(\overrightarrow{w}) = (\text{state}, \text{counters})$ *of the CA on* $\overrightarrow{w}$, *thus*

$$\text{size}_\mu(t) \leq \#\text{states} \cdot \#\text{counters} \cdot t.$$

Algorithm.

If problem specified by a CA then
we have a poly-time algorithm
in the size of the horizon and the CA.

Ready to solve…

… the fairness problem.