CMSC 28100

Introduction to Complexity Theory

Autumn 2025

Instructor: William Hoza

Circuit complexity of a binary language

- Let $Y \subseteq \{0,1\}^*$
- For each $n \in \mathbb{N}$, we define $Y_n \colon \{0,1\}^n \to \{0,1\}$ by the rule

$$Y_n(w) = \begin{cases} 1 & \text{if } w \in Y \\ 0 & \text{if } w \notin Y \end{cases}$$

- **Definition:** The circuit complexity of Y is the function $S \colon \mathbb{N} \to \mathbb{N}$ defined by

  $S(n) =$ the size of the smallest circuit that computes Y_n

- Note: Each circuit only handles a single input length! Different from TMs

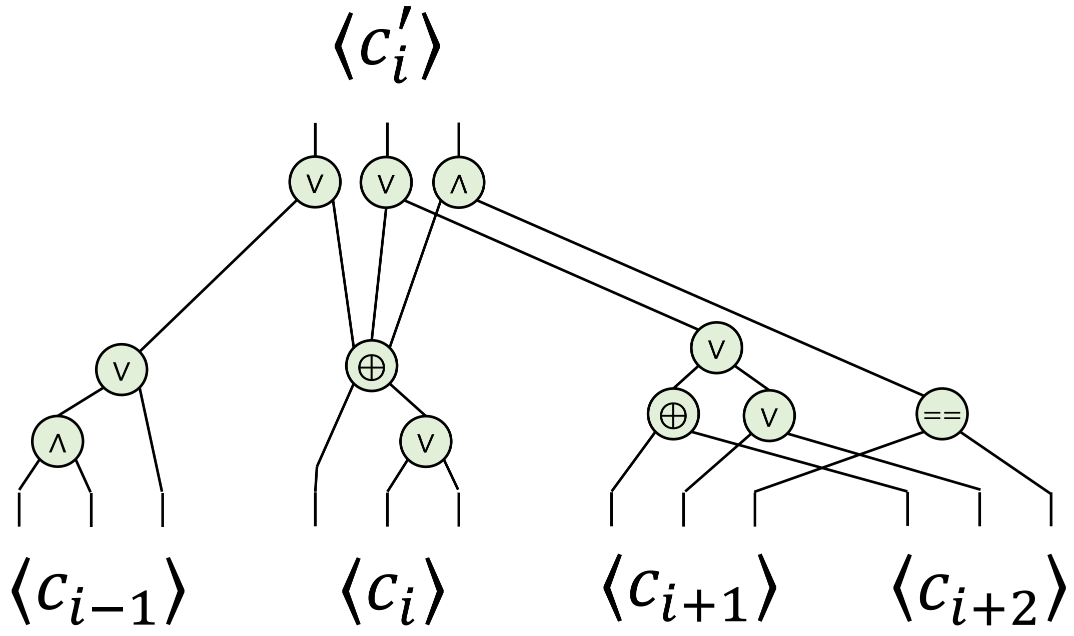
Turing machines vs. circuits

- Let M be a Turing machine that decides a language Y
- Let $T(n)$ be M's time complexity; let $S(n)$ be M's space complexity

Theorem: The circuit complexity of Y is $O\big(T(n) \cdot S(n)\big)$.

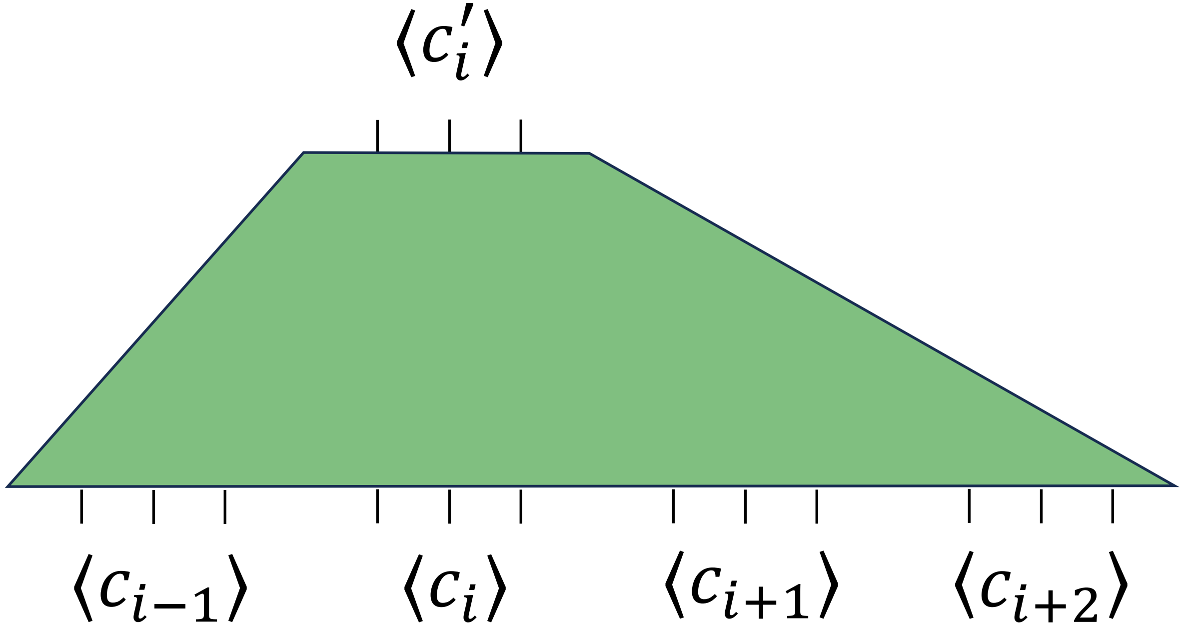
TM ⇒ Circuit

- Let $\mathrm{NEXT}(c_1 c_2 \ldots c_\ell) = c_1' c_2' \cdots c_\ell'$
- There is a circuit C_M that computes $\langle c_i' \rangle$ given $\langle c_{i-1} \rangle$, $\langle c_i \rangle$, $\langle c_{i+1} \rangle$, $\langle c_{i+2} \rangle$



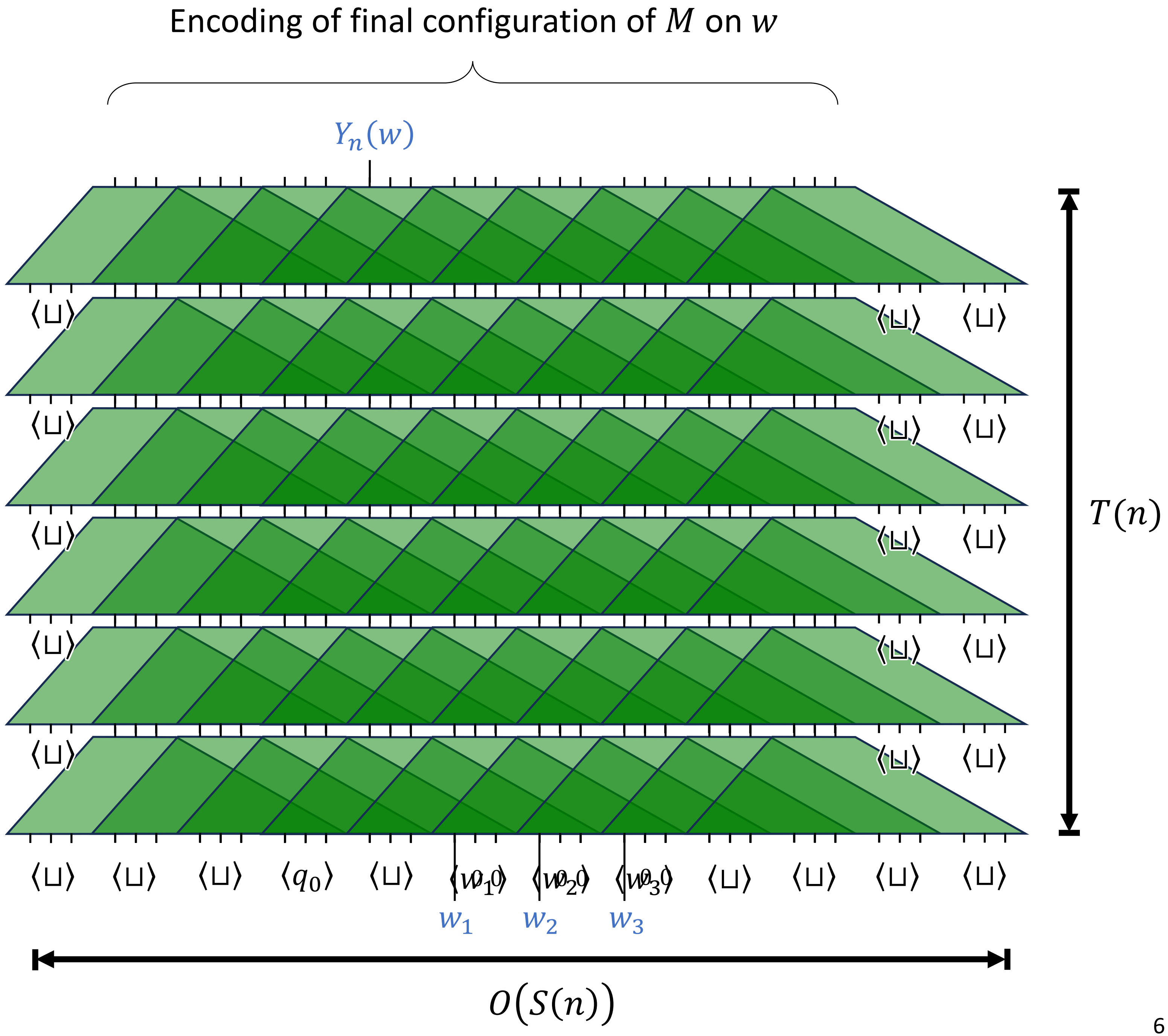
TM ⇒ Circuit

- Let $\mathrm{NEXT}(c_1 c_2 \ldots c_\ell) = c_1' c_2' \cdots c_\ell'$
- There is a circuit C_M that computes $\langle c_i' \rangle$ given $\langle c_{i-1} \rangle$, $\langle c_i \rangle$, $\langle c_{i+1} \rangle$, $\langle c_{i+2} \rangle$



TM $\Rightarrow$ Circuit

- Size: $O\big(S(n) \cdot T(n)\big)$
- Assume WLOG:
  - $\langle 0 \rangle = 0^r$ and $\langle 1 \rangle = 10^{r-1}$
  - M halts in starting cell
  - $\text{NEXT}(C) = C$ if C is a halting configuration
  - $\langle q_{\text{accept}} \rangle = 1^r$
  - $\langle q_{\text{reject}} \rangle = 01^{r-1}$



Turing machines vs. circuits

- Let $Y \subseteq \{0, 1\}^*$
- We just proved: If $Y \in \mathrm{P}$, then Y has polynomial circuit complexity
- ~~Converse?~~

Theorem: There exists an undecidable language $Y \subseteq \{0, 1\}^*$ with circuit complexity $O(n)$

An undecidable language with small circuits

- **Definition:** A unary language is a subset $Y \subseteq \{1\}^*$
- **Exercise 13:** There exists an undecidable unary language
- **Claim:** Every unary language Y has circuit complexity $O(n)$
  - **Proof:** If $1^n \in Y$, then $Y_n(w) \equiv w_1 \wedge w_2 \wedge \cdots \wedge w_n$
  - If $1^n \notin Y$, then $Y_n(w) \equiv 0$
  - Either way, # gates is $\leq n$

The complexity class $\mathbf{PSIZE}$

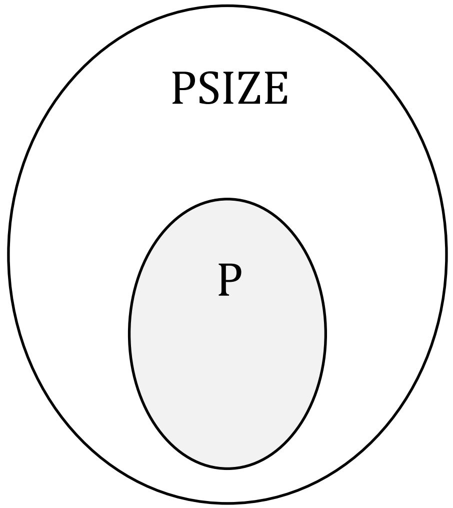
- Let $S: \mathbb{N} \to \mathbb{N}$ be a function
- **Definition:**

$$\mathrm{SIZE}(S) = \{Y \subseteq \{0,1\}^* : \text{the circuit complexity of } Y \text{ is } O(S)\}$$

- **Definition:**

$$\mathrm{PSIZE} = \{Y \subseteq \{0,1\}^* : \text{the circuit complexity of } Y \text{ is poly}(n)\} = \bigcup_{k=1}^{\infty} \mathrm{SIZE}(n^k)$$

How to interpret PSIZE



- We proved $\mathrm{P} \subseteq \mathrm{PSIZE}$ and $\mathrm{P} \neq \mathrm{PSIZE}$
- Circuits are more powerful than Turing machines
- Does this mean something is wrong with the Turing machine model?
- No! Something is “wrong” with the circuit model!
- PSIZE is not a good model of tractable languages!

Nonuniformity

- Let $Y \subseteq \{0, 1\}^*$
- "$Y \in \mathrm{PSIZE}$" means that there is a family of polynomial-size circuits that decide Y (one circuit for each input length)
- Each circuit performs only a polynomial amount of "work…"
- But what about the work required to construct these circuits?

Nonuniformity

- PSIZE allows us to use different “algorithms” for different input lengths
- Computing in this “nonuniform” manner is cheating / unrealistic
- However, it is a valuable conceptual tool!
- Alternative perspective: “Advice”

Computing with advice

- **Informal definition:** A language is in $\mathrm{P/poly}$ if it can be computed in polynomial time with the help of an "advisor"
- Advisor can instantly solve any computational problem
- Advisor is benevolent/trustworthy and will give you advice…
- …but the advice depends only on the length of your input!

Computing with advice

- Let $Y \subseteq \{0,1\}^*$
- **Definition:** $Y \in \mathrm{P/poly}$ if there exist "advice strings" $a_0, a_1, a_2, \ldots \in \{0,1\}^*$ and a polynomial-time Turing machine M such that:
  - $|a_n| \leq \mathrm{poly}(n)$
  - For every $w \in Y$, the machine M accepts $\langle w, a_{|w|} \rangle$
  - For every $w \notin Y$, the machine M rejects $\langle w, a_{|w|} \rangle$

Example: Unary languages

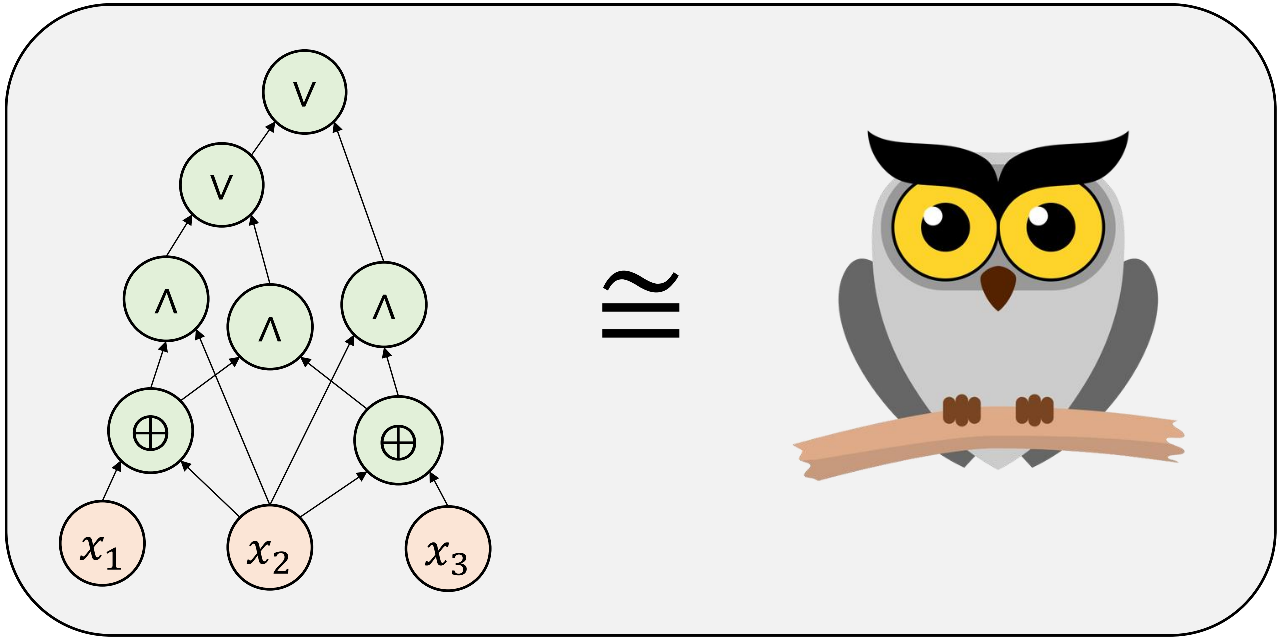
- **Claim:** If $Y \subseteq \{1\}^*$, then $Y \in \mathrm{P/poly}$

- **Proof:** Advice: $a_n = \begin{cases} 1 & \text{if } 1^n \in Y \\ 0 & \text{otherwise} \end{cases}$

- Given $\langle w, a \rangle$, the machine M operates as follows:
  - If $a = 1$ and w is all ones, accept
  - Otherwise, reject

Circuits vs. advice

Theorem: $\mathrm{PSIZE} = \mathrm{P/poly}$



- Step 1: Prove $\mathrm{P/poly} \subseteq \mathrm{PSIZE}$

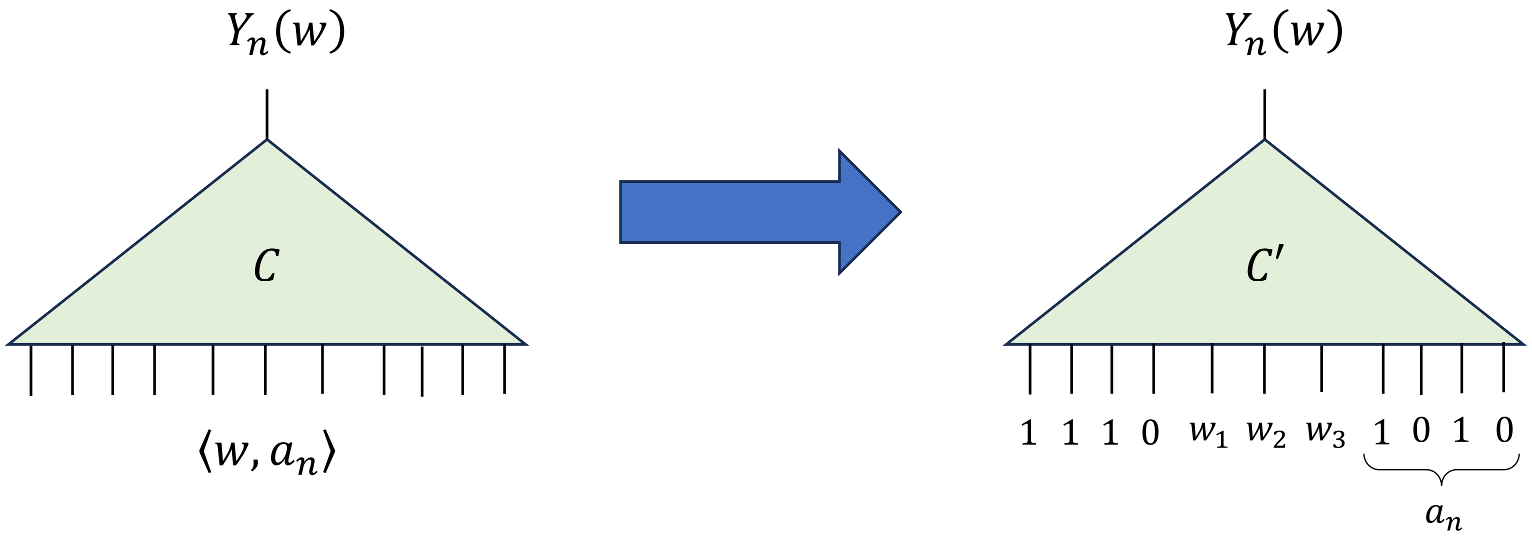
Proof that $\mathrm{P/poly} \subseteq \mathrm{PSIZE}$

- Let $Y \in \mathrm{P/poly}$ and let $n \in \mathbb{N}$
- Goal: Design a circuit of size $\mathrm{poly}(n)$ that decides Y_n
- There is a poly-time Turing machine M that decides Y using advice

$$\langle w, a_n \rangle \longrightarrow \boxed{M} \longrightarrow \begin{cases} \text{Accept} & \text{if } w \in Y \\ \text{Reject} & \text{if } w \notin Y \end{cases}$$

Proof that $\mathrm{P/poly} \subseteq \mathrm{PSIZE}$

- Polynomial-Time Turing Machine ⇒ Polynomial-Size Circuits



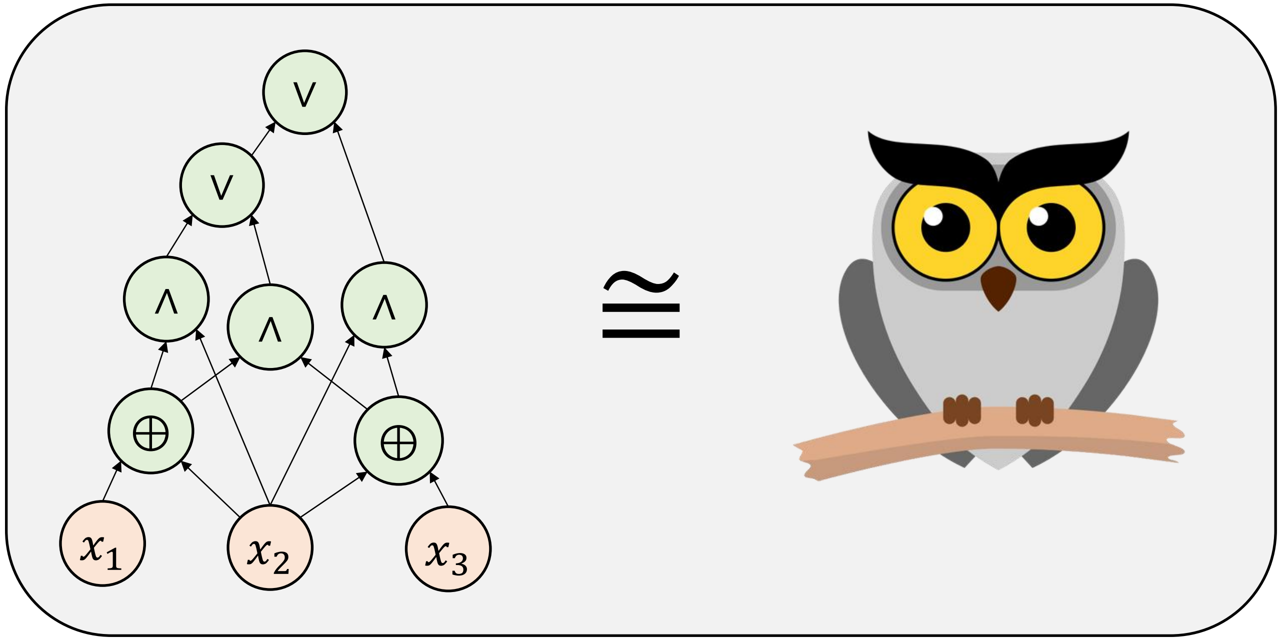
- Size of C' is $\mathrm{poly}(n')$ where $n' = |\langle w, a_n \rangle| = \mathrm{poly}(n)$ ✔

- Final step: "hard-code" the advice a_n

- Technicality: Use the encoding $\langle w, a \rangle = 1^{|w|}0wa$

Circuits vs. advice

Theorem: $\mathrm{PSIZE} = \mathrm{P/poly}$



- Step 1: Prove $\mathrm{P/poly} \subseteq \mathrm{PSIZE}$ ✔
- Step 2: Prove $\mathrm{PSIZE} \subseteq \mathrm{P/poly}$

Code as data III

- Recall principle: A Turing machine M can be encoded as a string $\langle M \rangle$
  - M is an algorithm, but at the same time, $\langle M \rangle$ can be an input to another algorithm!
- Similar idea: A circuit C can be encoded as a string $\langle C \rangle$
  - C is an "algorithm," but at the same time, $\langle C \rangle$ can be an input to another algorithm!
  - You'll explore encoding details in a homework exercise

Circuit value problem

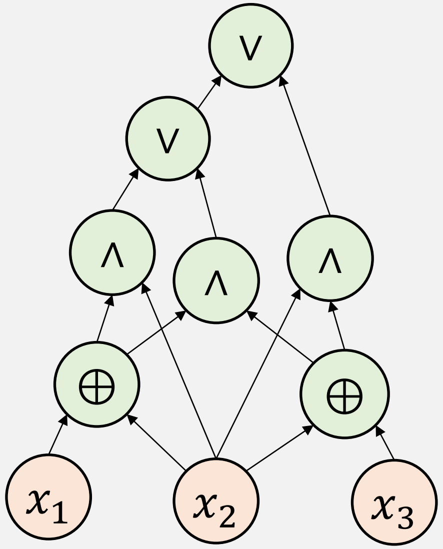
- Let $\mathrm{CIRCUIT\text{-}VALUE} = \{\langle x, C \rangle : C \text{ is a circuit and } C(x) = 1\}$
- **Lemma:** $\mathrm{CIRCUIT\text{-}VALUE} \in \mathrm{P}$
- **Proof sketch:** Suppose C has m nodes. To compute $C(x)$:
  1) Mark all the input nodes with their values
  2) While there is an unmarked node:
     a) For every gate g, find all the nodes that feed into g. If they are all marked with their values, then mark g with its value

Proof that $\mathrm{PSIZE} \subseteq \mathrm{P/poly}$

- Let $Y \in \mathrm{PSIZE}$
- Y_n can be computed by a circuit C_n of size $\mathrm{poly}(n)$
- Advice: $a_n = \langle C_n \rangle$
- Given w and $\langle C_n \rangle$, we can figure out whether $w \in Y$ by computing $C_n(w)$
  - This is exactly the circuit value problem

Circuits vs. advice

Theorem: $\mathrm{PSIZE} = \mathrm{P/poly}$



$\cong$

- Step 1: Prove $\mathrm{P/poly} \subseteq \mathrm{PSIZE}$ ✔
- Step 2: Prove $\mathrm{PSIZE} \subseteq \mathrm{P/poly}$ ✔