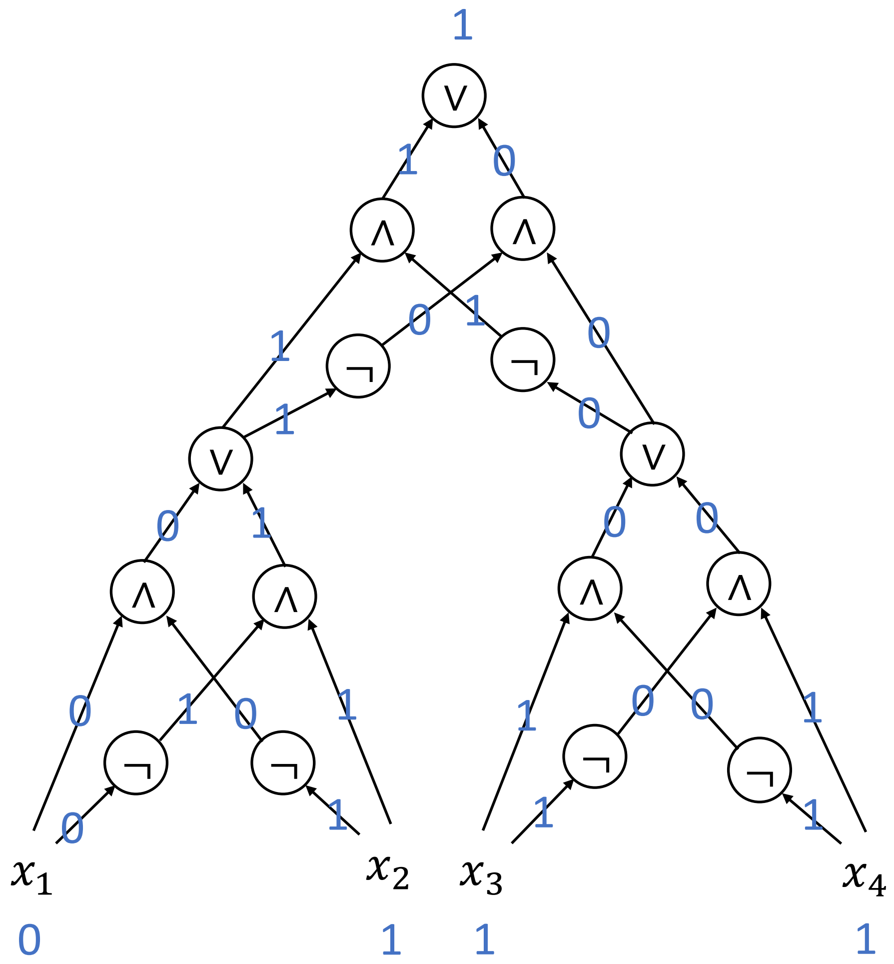
CMSC 28100

Introduction to Complexity Theory

Spring 2024
Instructor: William Hoza

Boolean circuits

- For us, a "circuit" is a network of logic gates applied to Boolean variables



- ∨ means OR
- ∧ means AND
- ¬ means NOT
- Each x_i can be either 0 or 1

  (FALSE or TRUE)

Circuit complexity of a binary language

- Let $L \subseteq \{0,1\}^*$ be a language
- For each $n \in \mathbb{N}$, we define $L_n : \{0,1\}^n \to \{0,1\}$ by the rule

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

- We define the circuit complexity of L to be the function $S: \mathbb{N} \to \mathbb{N}$ defined by $S(n) =$ the size of the smallest circuit that computes L_n
- Note: Each circuit only handles a single input length! Different from TMs

The complexity class PSIZE

- Let $S: \mathbb{N} \to \mathbb{N}$ be a function
- **Definition:** $\mathrm{SIZE}(S)$ is the set of all languages L such that the circuit complexity of L is $O(S)$
- **Definition:** PSIZE is the set of all languages with polynomial circuit complexity:

$$\mathrm{PSIZE} = \bigcup_{k=1}^{\infty} \mathrm{SIZE}(n^k)$$

Circuit complexity vs. time complexity

- Let $T: \mathbb{N} \rightarrow \mathbb{N}$ be any function (time bound)

Theorem: $\mathrm{TIME}(T) \subseteq \mathrm{SIZE}(T^2)$. In particular, $\mathrm{P} \subseteq \mathrm{PSIZE}$.

- Polynomial Time Algorithm ⇒ Polynomial Size Circuits
- The proof is based on computation histories

Locality of computation

- Let C be a configuration of a TM M
- We can write $C = c_1 c_2 \dots c_\ell$ for some $c_1, \dots, c_\ell \in \Gamma \cup Q$
- Then $\mathrm{NEXT}(C) = c_1' c_2' \dots c_\ell'$ for some $c_1', \dots, c_\ell' \in \Gamma \cup Q$
- **Fact:**

$$c_i' = \begin{cases} \text{the third symbol of } \mathrm{NEXT}(\Diamond c_{i-1} c_i c_{i+1} c_{i+2}) & \text{if } c_{i-1} \in Q \text{ or } c_i \in Q \text{ or } c_{i+1} \in Q \\ c_i & \text{otherwise} \end{cases}$$

Encoding configurations in binary

- Let C be a configuration of a TM M, say $C = u_1 u_2 \dots u_k q v_1 v_2 \dots v_m$
- Each symbol/state $b \in \Gamma \cup Q$ can be encoded in binary as $\langle b \rangle \in \{0, 1\}^r$ for some $r = O(1)$
- We define $\langle C \rangle = \langle u_1 \rangle \langle u_2 \rangle \cdots \langle u_k \rangle \langle q \rangle \langle v_1 \rangle \cdots \langle v_m \rangle$

From one configuration to the next

- Suppose $\langle C \rangle = B_1 B_2 \cdots B_\ell$ where $B_i \in \{0,1\}^r$
- Suppose $\langle \text{NEXT}(C) \rangle = B_1' B_2' \cdots B_\ell'$ where $B_i' \in \{0,1\}^r$

If we want to know B_i', which blocks of $\langle C \rangle$ do we need to look at?

A: All of them

B: Only B_i

C: B_i and B_{i+1}

D: B_{i-1}, B_i, B_{i+1}, and B_{i+2}

Respond at PollEv.com/whoza or text "whoza" to 22333

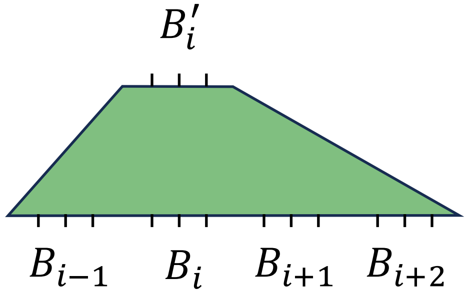
From one configuration to the next

- There is a function $\Delta_M: \{0,1\}^{4r} \to \{0,1\}^r$ such that for every configuration C, if $\langle C \rangle = B_1 B_2 \cdots B_\ell$ and $\langle \mathrm{NEXT}(C) \rangle = B_1' B_2' \cdots B_\ell'$, then for every i, we have

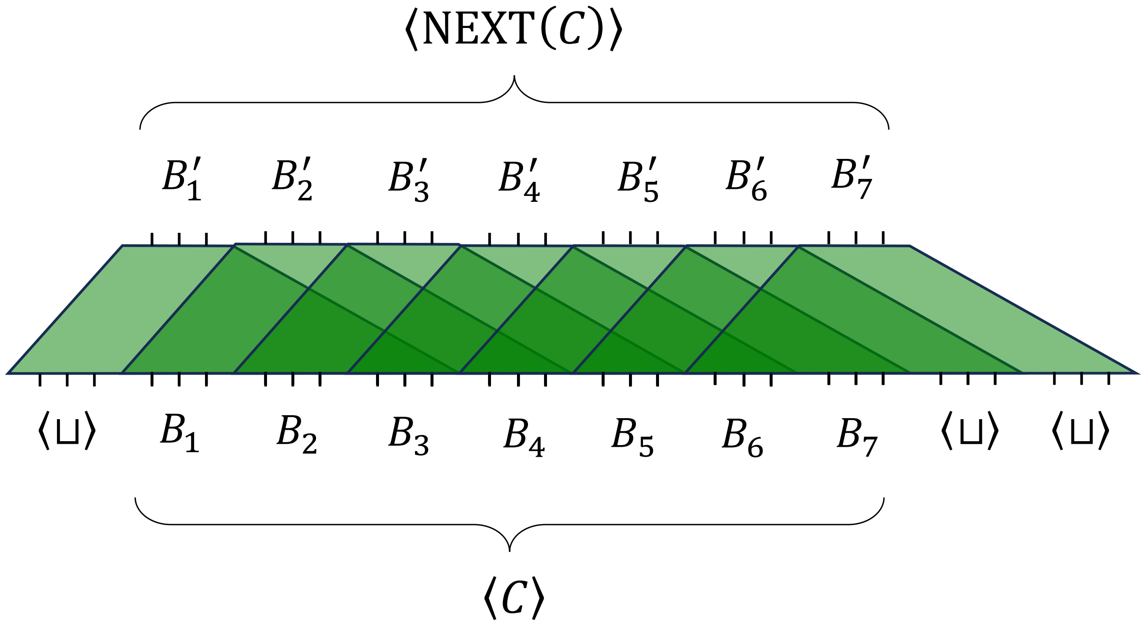
$$\Delta_M(B_{i-1} B_i B_{i+1} B_{i+2}) = B_i'$$

- This formula works for all $1 \leq i \leq \ell$, if we define $B_0 = B_{\ell+1} = B_{\ell+2} = \langle \sqcup \rangle$
- Intuitively, Δ_M is a version of the transition function of M

From one configuration to the next

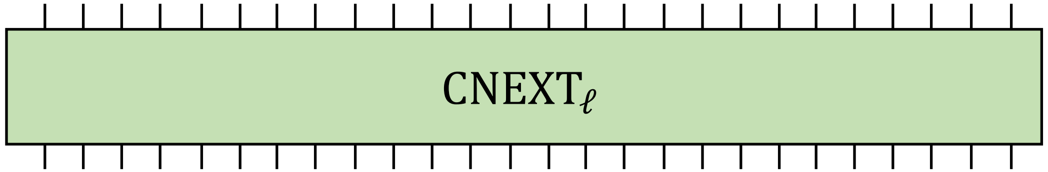


- There is a circuit of size $O(1)$ that computes Δ_M
- Now let's combine many copies of that circuit in parallel:



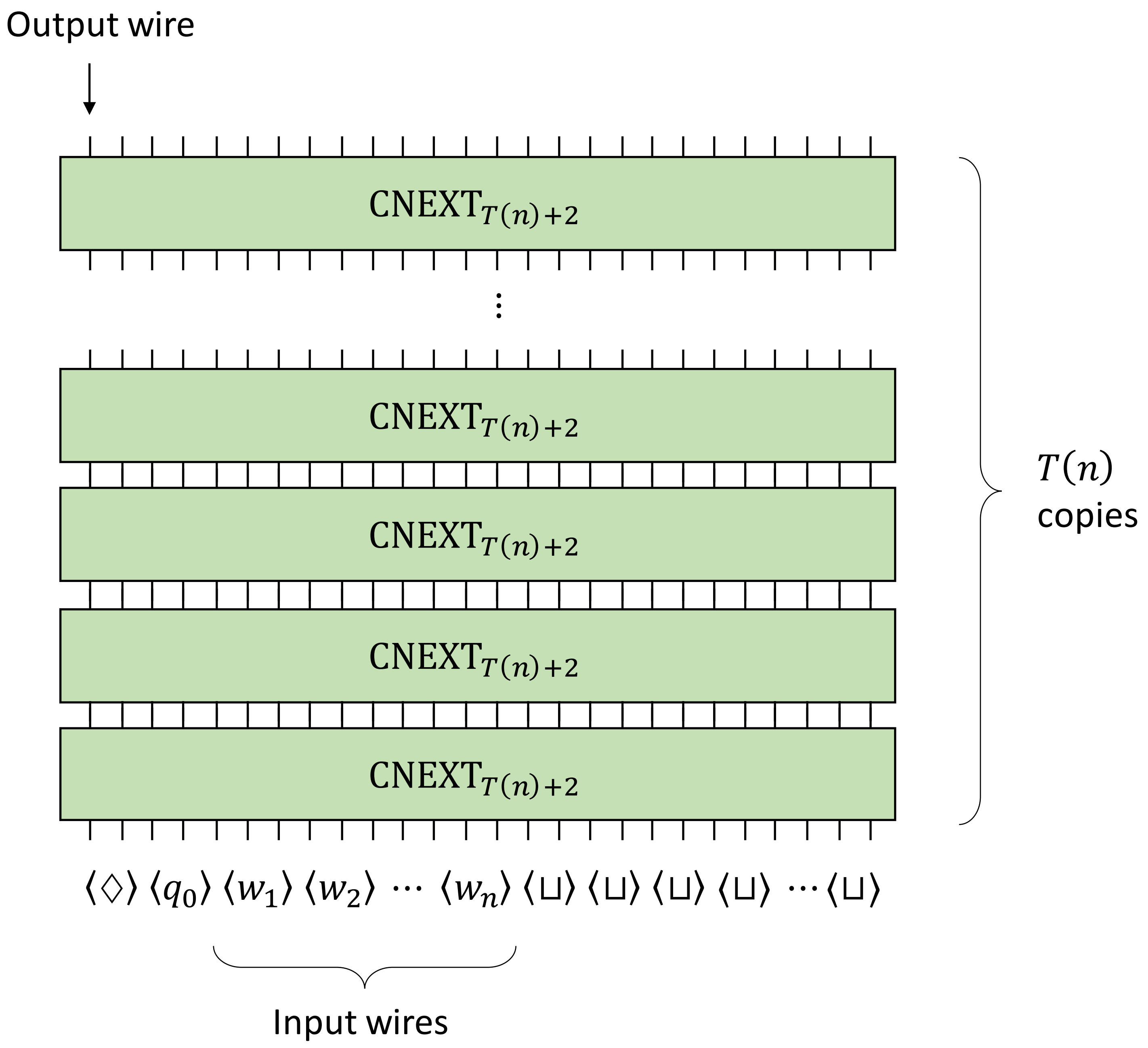
From one configuration to the next

- The construction on the previous slide shows that for every $\ell \in \mathbb{N}$, there is a circuit $\mathrm{CNEXT}_\ell : \{0,1\}^{r\ell} \to \{0,1\}^{r\ell}$ satisfying the following properties:
  - If C is a configuration such that $|C| = |\mathrm{NEXT}(C)| = \ell$, then
  $$\mathrm{CNEXT}_\ell(\langle C \rangle) = \langle \mathrm{NEXT}(C) \rangle$$
  - The size of CNEXT_ℓ is $O(\ell)$

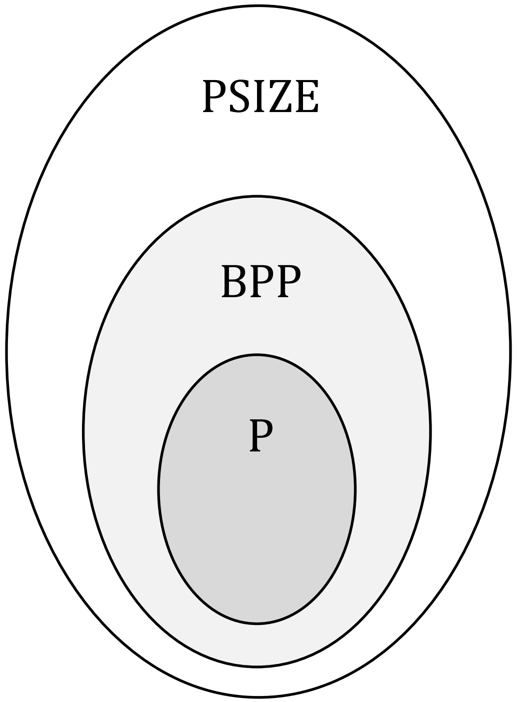


Machine ⇒ circuit

- Let M be a TM that decides L with time complexity $T(n)$
- Assume WLOG:
  - M halts in cell 1
  - $\langle q_{\text{accept}} \rangle$ begins with 1
  - $\langle q_{\text{reject}} \rangle$ begins with 0
- We get a circuit computing L_n
- Size: $O(T(n)^2)$



Adleman's theorem

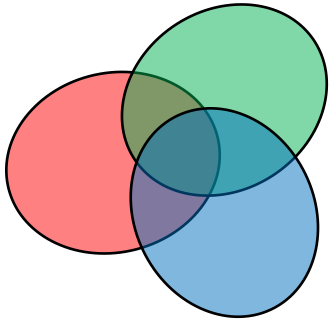


- We just showed that $\mathrm{P} \subseteq \mathrm{PSIZE}$
- Next, we will prove a stronger theorem:

Adleman's Theorem: $\mathrm{BPP} \subseteq \mathrm{PSIZE}$

- Note: The circuit model is a deterministic model of computation!
- Adleman's theorem is tantalizingly similar to the statement "$\mathrm{P} = \mathrm{BPP}$"

The union bound



- The proof of Adleman's theorem uses a key fact from probability theory:

The Union Bound: For any events $E_1, E_2, \ldots, E_k$, we have

$$\Pr[E_1 \text{ or } E_2 \text{ or} \ldots \text{or } E_k] \leq \Pr[E_1] + \Pr[E_2] + \cdots + \Pr[E_k]$$

- Example: Suppose $\Pr[\text{ice on floor}] = 0.3$ and $\Pr[\text{liquid water on floor}] = 0.4$
- Then $\Pr[\text{water on floor}] \leq 0.3 + 0.4 = 0.7$

Proof of Adleman's theorem ($\mathrm{BPP} \subseteq \mathrm{PSIZE}$)

- **Proof:** Let $L \in \mathrm{BPP}$ where $L \subseteq \Sigma^*$
- Amplification lemma $\Rightarrow$ There exists a polynomial-time randomized Turing machine M such that for every $n \in \mathbb{N}$ and every $w \in \Sigma^n$:
  - If $w \in L$, then $\Pr[M \text{ accepts } w] > 1 - 1/|\Sigma|^n$
  - If $w \notin L$, then $\Pr[M \text{ rejects } w] < 1 - 1/|\Sigma|^n$
- Let $T(n)$ be the time complexity of M

“Good” random bits

- Let $n \in \mathbb{N}$, let $w \in \Sigma^n$, and let $u \in \{0, 1\}^{T(n)}$
- We say that u is good for w if:
  - $w \in L$ and M accepts w when tape 2 is initialized with u, or
  - $w \notin L$ and M rejects w when tape 2 is initialized with u.
- Otherwise, we say that u is bad for w

Random bits: Good for all inputs simultaneously

Claim: For every n, there exists $u_* \in \{0,1\}^{T(n)}$ that is good for all $w \in \Sigma^n$

- **Proof:** By the union bound, if we pick $u \in \{0,1\}^{T(n)}$ uniformly at random,

$$\Pr\left[\begin{matrix}\text{there exists } w \in \Sigma^n \\ \text{such that } u \text{ is bad for } w\end{matrix}\right] \leq \sum_{w \in \Sigma^n} \Pr[u \text{ is bad for } w] < |\Sigma^n| \cdot \frac{1}{|\Sigma|^n} = 1$$

- There is a nonzero chance that u is good for all w, so there must exist at least one u_* that is good for all w