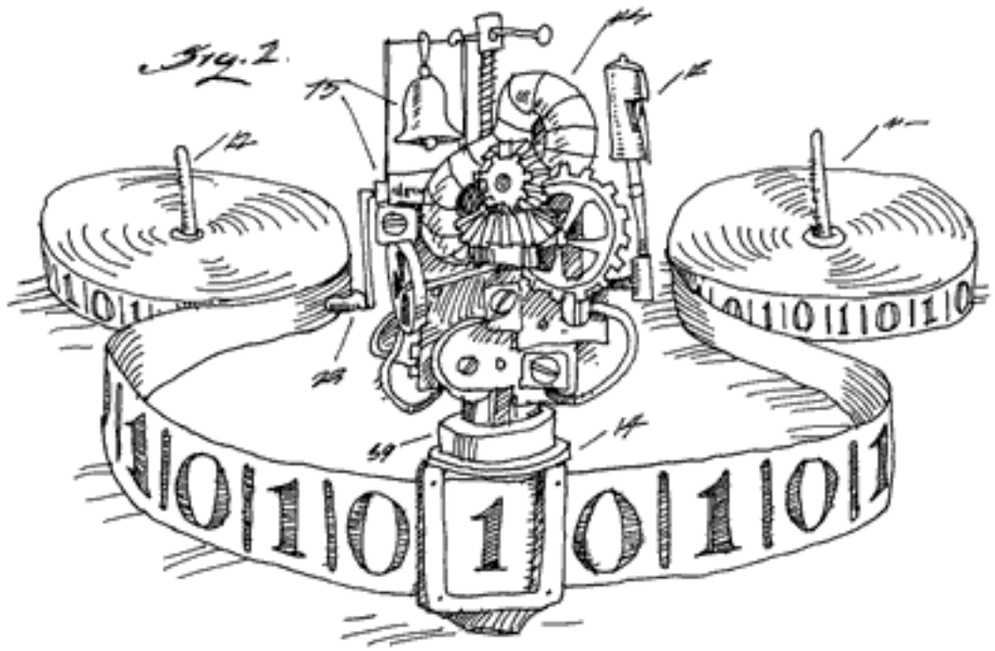
CSC4010: Introduction to Theoretical Computer Science



Lecture 17

Goal: Cook-Levin Theorem?

SAT is NP-complete

SAT = {⟨F⟩ | F is satisfiable}

Steps:

1) CIRCUIT-SAT is NP-complete.
(show it is NP-hard)

2) In order to show this, we will use the fact that a TM M can be simulated by a family of circuits.

Recap: Simulating Turing Machines as Circuits

Theorem: Let $T : \mathbb{N} \to \mathbb{N}$ be any time constructible function (time bound). Then, TIME(T) $\subseteq$ SIZE(T^2) . In particular, $P \subseteq$ PSIZE .

{L | L is decidable in time O(T)} → {L | L is decidable using circuits of size T^2}.

Key Observation:

Locality of computation: Given a Turing Machine M, let $B = b_1 b_2 \cdots b_l$ be the current configuration and let NEXT(B) $= b'_1 b'_2 \cdots b'_l$ then, b'_i depends on $b_{i-1}, b_i, b_{i+1}, b_{i+2}$.

Proof: If we have a TM that runs in time $O(T)$

↳ $b_i \in Q \cup \Sigma$.

↳ tape contents + placement of head + state in which TM is.

↳ derived from B using δ. last time: b'_i depends only on $b_{i-1}, b_i, b_{i+1}, b_{i+2}$.

Recap: Simulating Turing Machines as Circuits

Theorem: Let $T : \mathbb{N} \to \mathbb{N}$ be any time constructible function (time bound). Then, $\mathrm{TIME}(T) \subseteq \mathrm{SIZE}(T^2)$. In particular, $P \subseteq \mathrm{PSIZE}$.

Let $M = (Q, \Sigma, q_{\text{acc}}, q_{\text{rej}}, q_0, \sqcup, \delta)$. Then, for any configuration $B = b_1 b_2 \cdots b_l$, where $b_i \in Q \cup \Sigma$. Suppose $|Q \cup \Sigma| = 2^r$

b_i is a string in $Q \cup \Sigma$. Each b_i can be encoded in $\langle B \rangle \in \{0,1\}^{lr}$. binary using at most r bits.

$\langle B \rangle = \langle b_1 \rangle \langle b_2 \rangle \cdots \langle b_l \rangle$ and $|\langle b_i \rangle| = r$.

There is a function $\Delta_M : \{0,1\}^{4r} \to \{0,1\}^r$ such that for every configuration B, if $\langle B \rangle = b_1 b_2 \cdots b_l$ and $\langle \mathrm{NEXT}(B) \rangle = b'_1 b'_2 \cdots b'_l$ then for every i, we have,

$$\Delta_M(b_{i-1} b_i b_{i+1} b_{i+2}) = b'_i$$

Since the TM description is fixed, we have $r = O(1)$.

From the previous slide, b'_i is a function of $b_i, b_{i+1}, b_{i+2}, b_{i-1}$

Δ_M must take i/p $b_{i-1}, b_i, b_{i+1}, b_{i+2}$ and output b'_i.

Recap: Simulating Turing Machines as Circuits

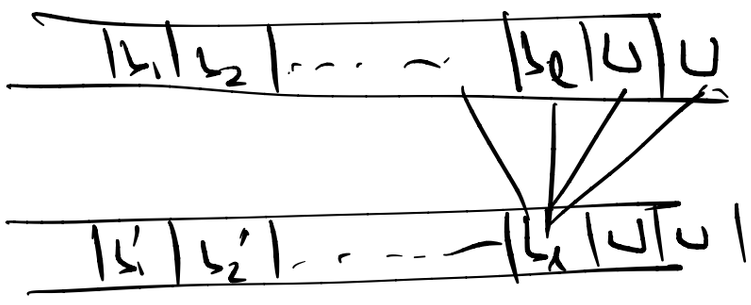
Theorem: Let $T : \mathbb{N} \to \mathbb{N}$ be any time constructible function (time bound). Then, $\mathsf{TIME}(T) \subseteq \mathsf{SIZE}(T^2)$. In particular, $P \subseteq \mathsf{PSIZE}$.

Let $M = (Q, \Sigma, q_{\text{acc}}, q_{\text{rej}}, q_0, \sqcup, \delta)$. Then, for any configuration $B = b_1 b_2 \cdots b_l$, where $b_i \in Q \cup \Sigma$. Suppose $|Q \cup \Sigma| = r$

$\langle B \rangle = \langle b_1 \rangle \langle b_2 \rangle \cdots \langle b_l \rangle$ and $|\langle b_i \rangle| = r$.

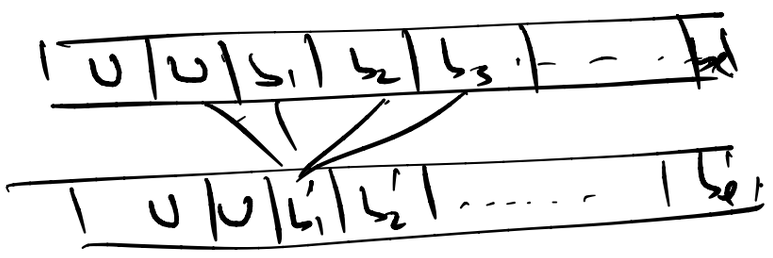
There is a function $\Delta_M : \{0,1\}^{4r} \to \{0,1\}^r$ such that for every configuration B, if $\langle B \rangle = b_1 b_2 \cdots b_l$ and $\langle \mathsf{NEXT}(B) \rangle = b'_1 b'_2 \cdots b'_l$ then for every i, we have,

$\Delta_M(b_{i-1} b_i b_{i+1} b_{i+2}) = b'_i$



$\Delta_M(\sqcup b_1 b_2 b_3) = b'_1$.

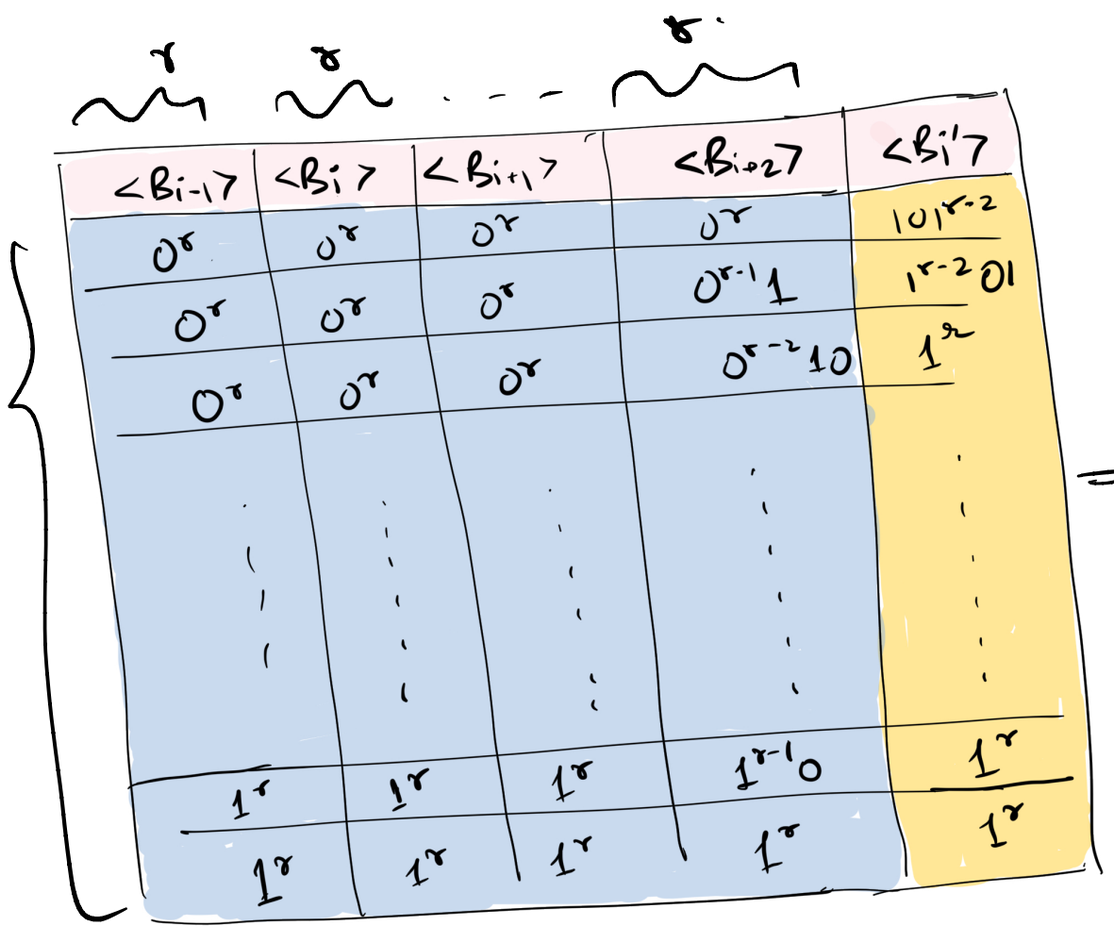
b'_1 depends on b_2, b_3, b_4.

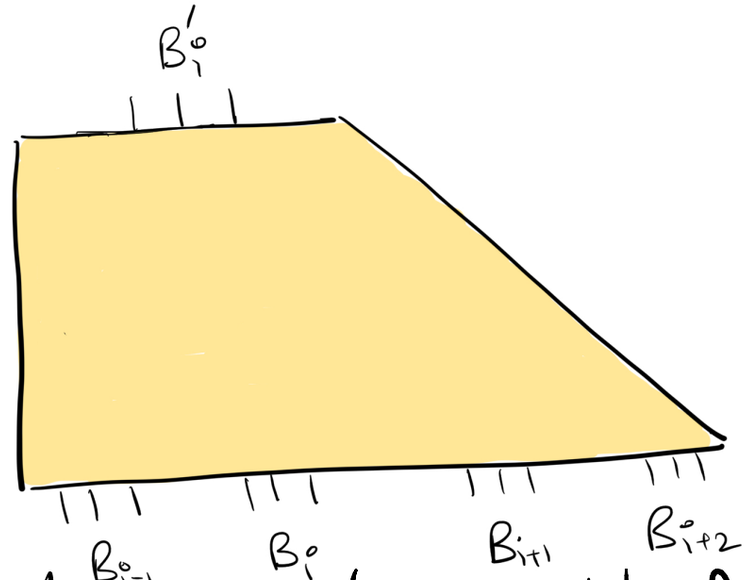


Recap: Simulating Turing Machines as Circuits

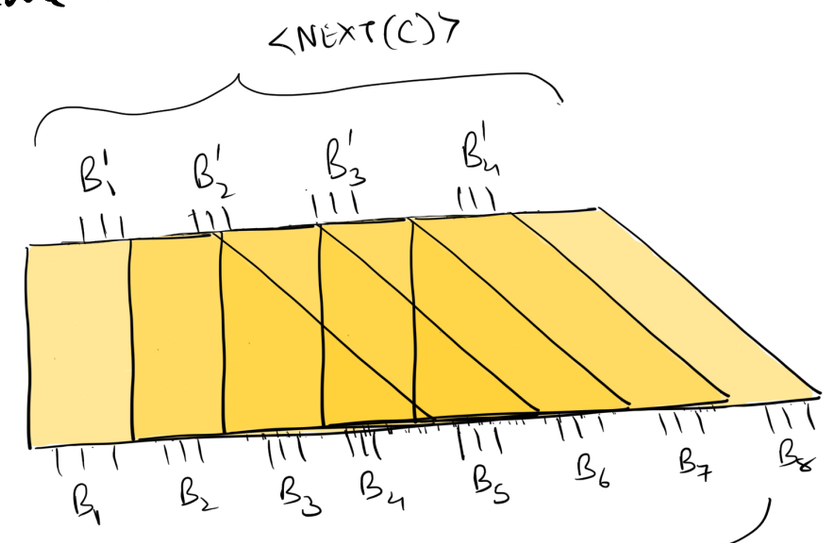
$\Delta_M : \{0,1\}^{qr} \longrightarrow \{0,1\}^{r}$

Time for computing the circuit for Δ_M:





⇒ Construct a boolean circuit for each bit of $\langle B_i' \rangle$.
Takes $O(1)$ time.

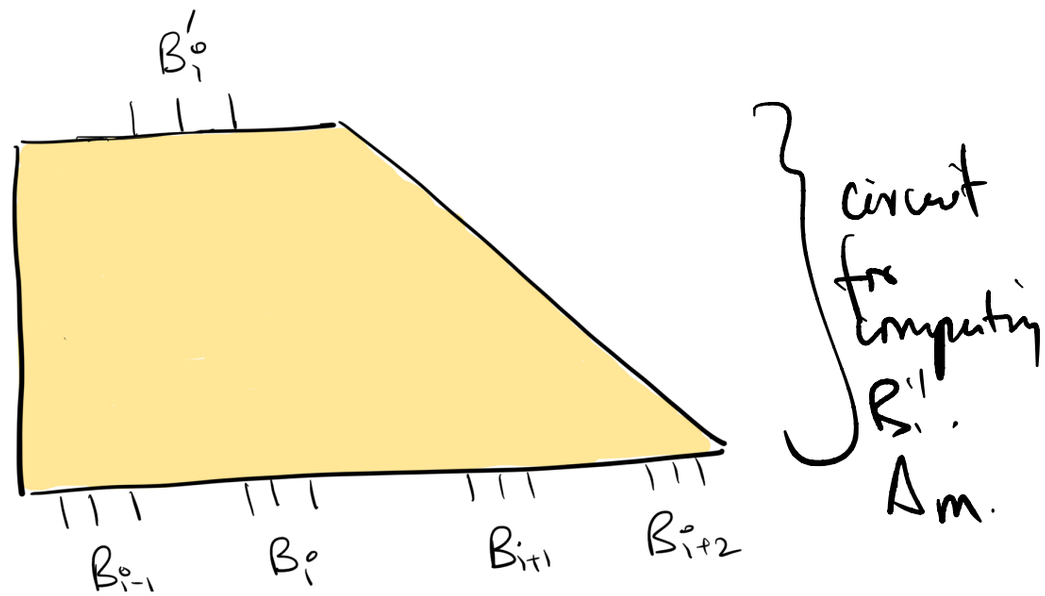


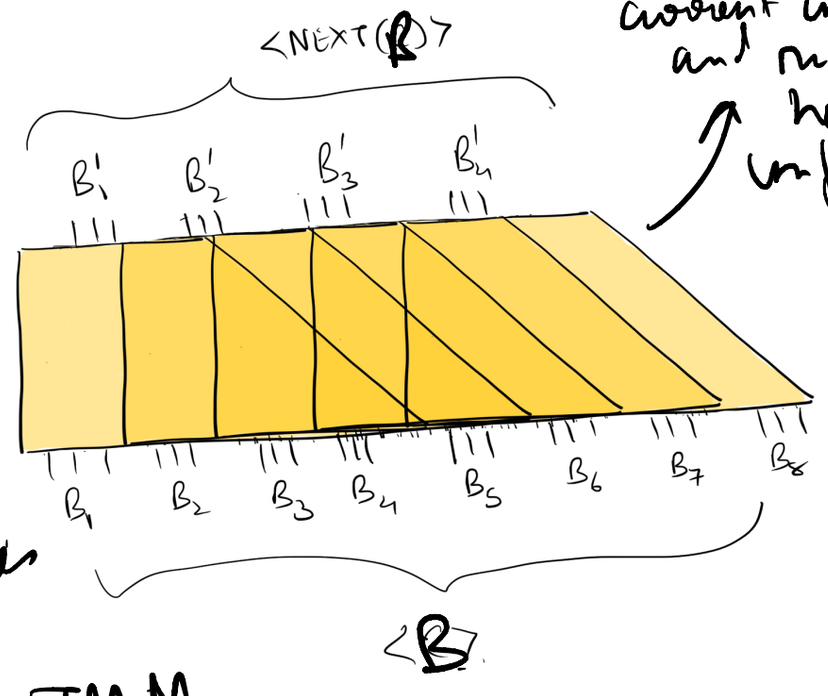
$O(2^r)$ is the # ones.
The length of this truth table is $O(1)$.
Using δ from M this table can be populated in $O(2^r)$ time.
$= O(1)$ time.

Recap: Simulating Turing Machines as Circuits

Time for computing the circuit for Δ_M:

$\langle B_{i-1}\rangle$	$\langle B_i\rangle$	$\langle B_{i+1}\rangle$	$\langle B_{i+2}\rangle$	$\langle B_i'\rangle$
0^r	0^r	0^r	0^r	101^{r-2}
0^r	0^r	0^r	$0^{r-1}1$	$1^{r-2}01$
0^r	0^r	0^r	$0^{r-2}10$	1^r
⋮	⋮	⋮	⋮	⋮
1^r	1^r	1^r	$1^{r-1}0$	1^r
1^r	1^r	1^r	1^r	1^r

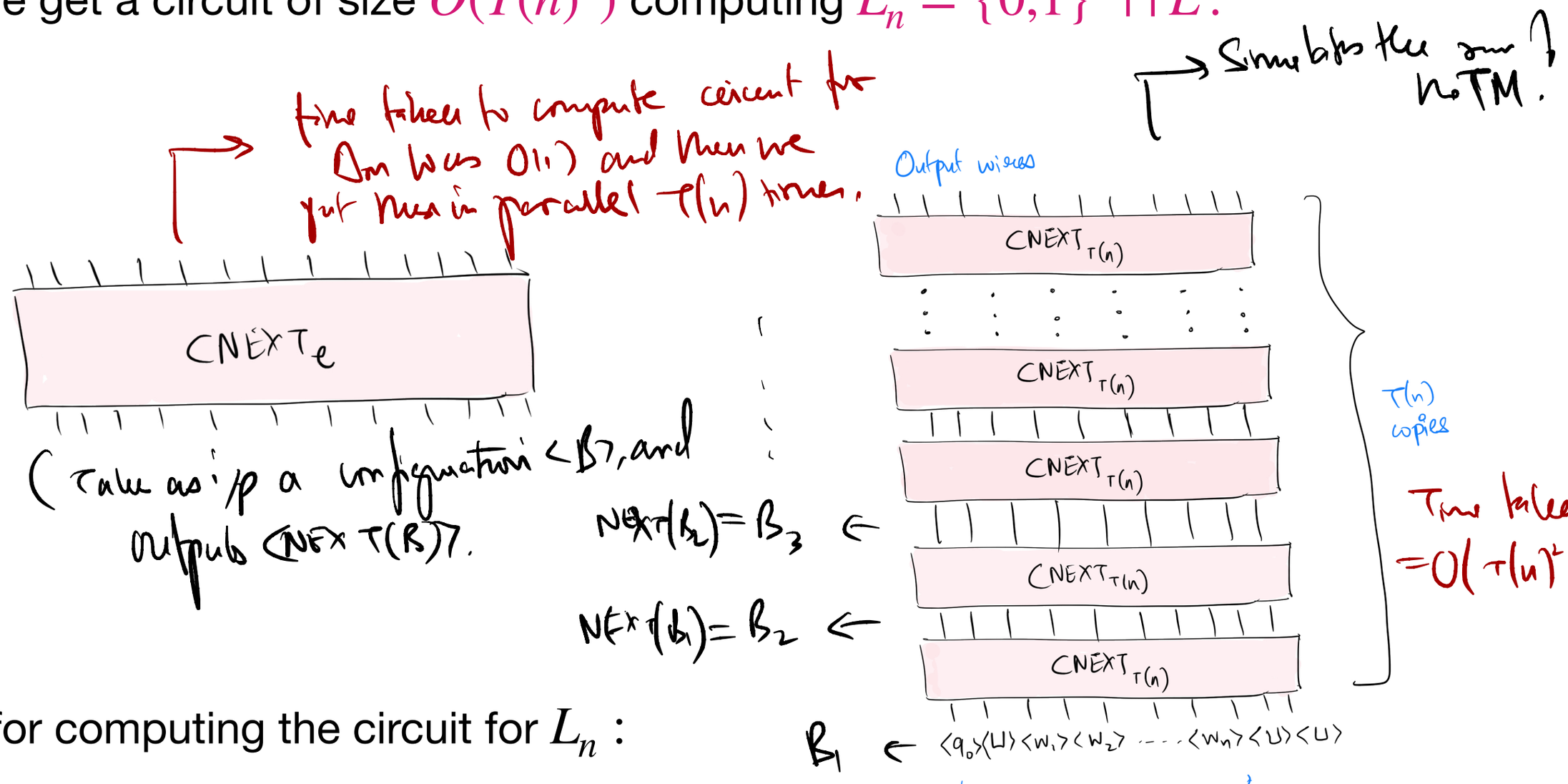




Obtain a circuit for computing $NEXT(B)$ from B, by the circuit in parallel, $T(n)$ copies of them, where $T(n)$ is the runtime of TM M.

Recap: Simulating Turing Machines as Circuits

- Let M be a Turing Machine that decides L with time complexity $T(n)$.
- We get a circuit of size $O(T(n)^2)$ computing $L_n = \{0,1\}^n \cap L$.



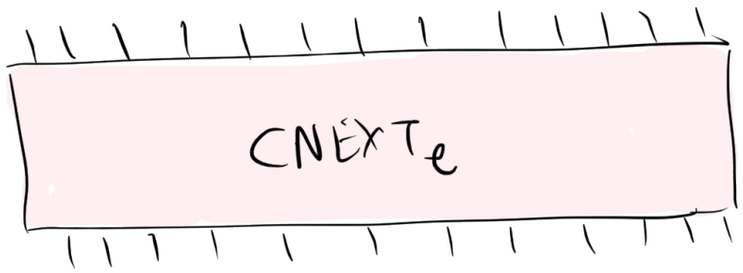
Time for computing the circuit for L_n :

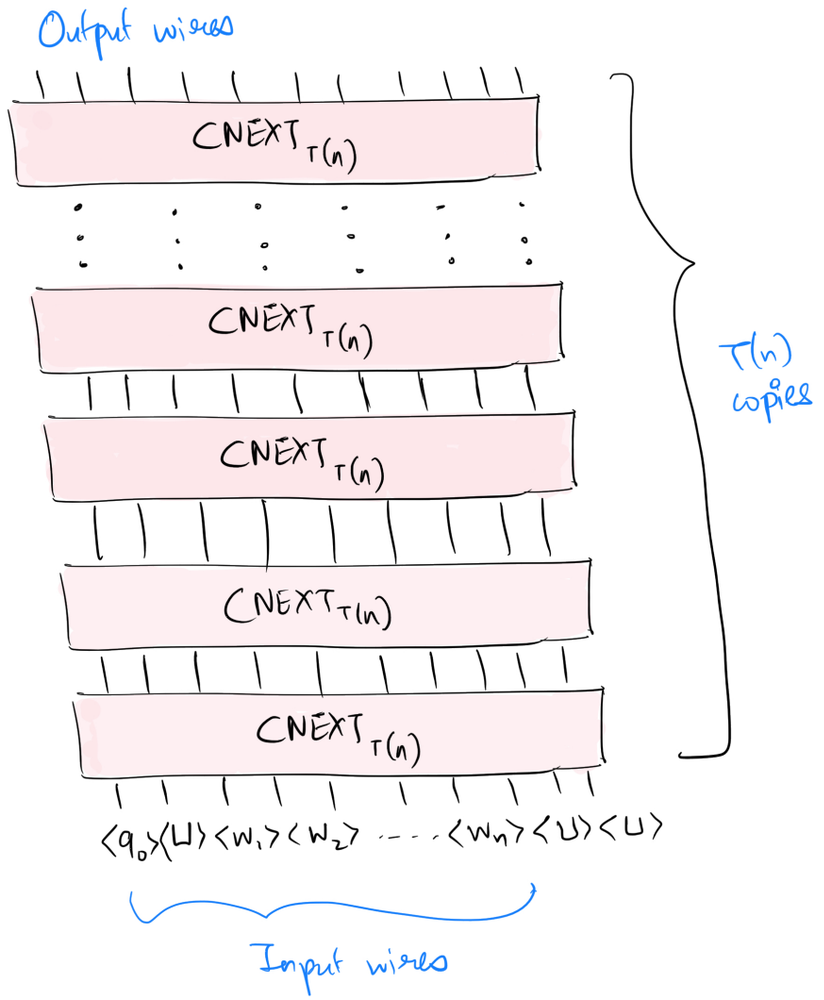
As the runtime of M on any i/p of length n is $\leq T(n)$, one of $B_1, B_2, \ldots, B_{T(n)}$ is halting. Add auxiliary circuits to test which of these are halting. If they are A/R.

Summary: In time $O(T(n)^2)$ we can obtain a circuit that decides L_n and the size of the circuit is $O(T(n)^2)$.

Recap: Simulating Turing Machines as Circuits

- Let M be a Turing Machine that decides L with time complexity $T(n)$.
- We get a circuit of size $O(T(n)^2)$ computing $L_n = \{0,1\}^n \cap L$.





Time for computing the circuit for L_n :

CIRCUIT-SAT is NP-Complete

a) CIRCUIT-SAT $\in$ NP

b) CIRCUIT-SAT is NP-hard.

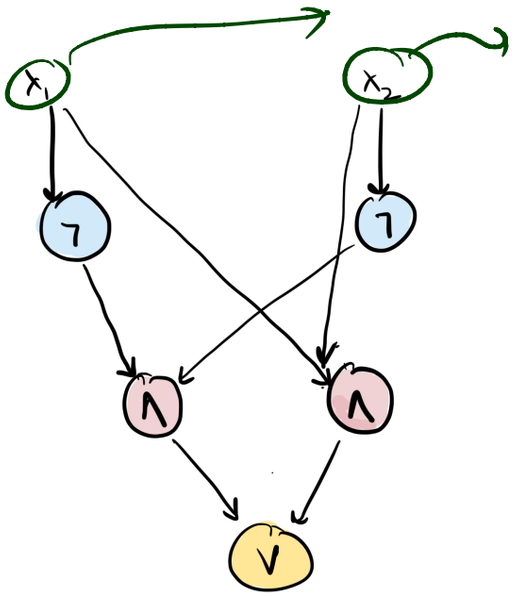
Circuit Satisfiability

We say that circuit C (on n inputs) is satisfiable if there exists $x \in \{0,1\}^n$ such that $C(x) = 1$

CIRCUIT-SAT $= \{\langle C \rangle \mid \exists x \text{ s.t. } C(x) = 1\}$

CIRCUIT-SAT $\in NP$

a) Verifier : on i/p tape $\langle C \rangle$ and the certificate is a satisfying assignment x.



y_1, y_2 are i/p variables and they are set to
$y_1 = x_1 = 0$
$y_2 = x_2 = 1$.

b) Soundness & completeness: $\exists x$ s.t. $C(x) = 1 \Leftrightarrow \langle C \rangle \in$ CIRCUIT-SAT

c) $|x| \leq |\langle C \rangle|$

d) to evaluate C on x, we showed a polynomial time algorithm.

The circuit gets some values, and the # boolean values it gets as i/p is $\leq$ # input variables.

CIRCUIT-SAT is NP-Hard

$Y \in NP$, $Y \leq_p$ CIRCUIT-SAT.

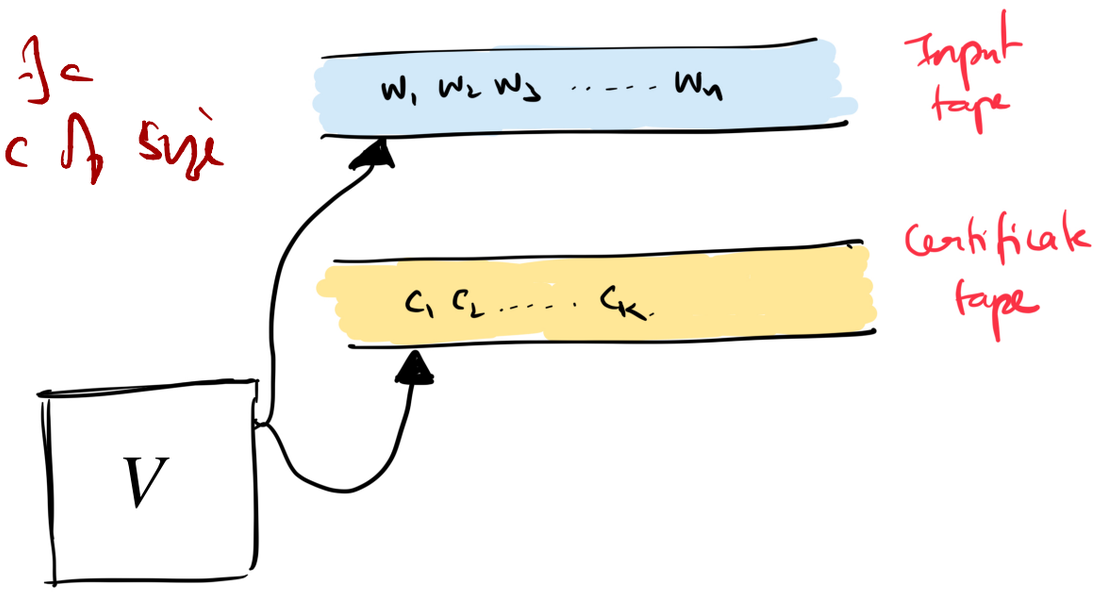
CIRCUIT-SAT is NP-Hard

- Consider $Y \subseteq \{0,1\}^*$ such that $Y \in NP$:
  - There exists verifier V such that $w \in Y \iff \exists k_1, k_2 \geq 0$ such that $V(w, c) = 1$ for some $|c| \leq |w|^{k_1}$ and V runs in time at most $|w|^{k_2}$.

a) Completeness: $\forall w \in Y$, $\exists c$ certificate c of size $\leq |w|^k$. s.t. $V(w,c)$

b) Soundness: $\forall w \notin Y$, there is no such certificate.

c) The verifier runs in $|w|^{k_2}$. for $k_2 \geq 0$.



CIRCUIT-SAT is NP-Hard

Theorem: CIRCUIT-SAT is NP-Hard.

- **Proof:** Given any $Y \in$ NP, show that $Y \leq_P$ CIRCUIT-SAT.

1) $\exists$ a function f s.t. $x \in Y \Leftrightarrow f(x) \in$ CIRCUIT-SAT

2) $\exists$ a TM F s.t. on i/p x, F writes $f(x)$ on the tape & halts in time $|x|^k$.

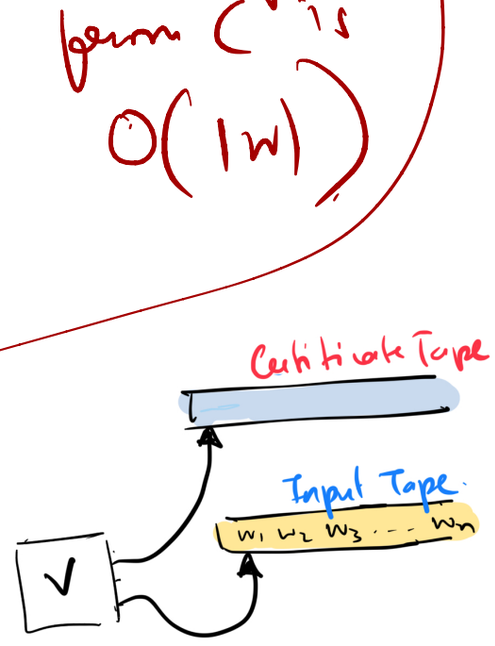
CIRCUIT-SAT is NP-Hard

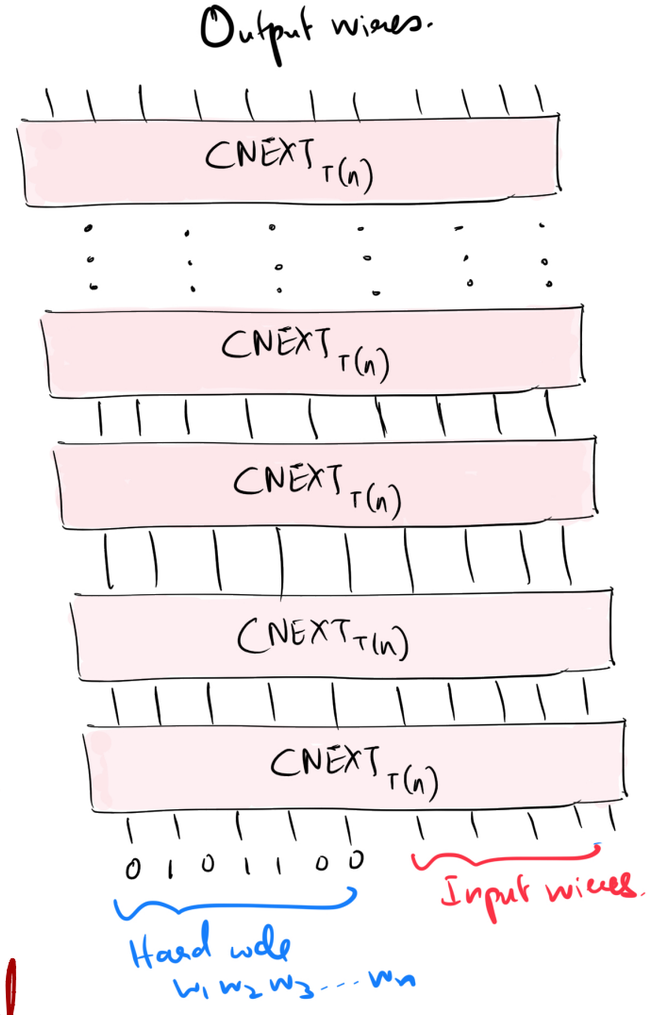
Claim: $w \in Y \Leftrightarrow \exists\, c$ s.t. $V(w,c) = 1 \Leftrightarrow C_w^V(x) = 1 \Leftrightarrow C_w^V \in$ CIRCUIT-SAT.

Theorem: CIRCUIT-SAT is NP-Hard.

- **Proof:** Given any $Y \in$ NP, show that $Y \leq_P$ CIRCUIT-SAT.

a) Time taken to compute $C^V = |w|^k$,
b) to compute C_w^V from C^V is also $O(|w|)$





i/p an instance $w \in Y$,
a) f will hardcode w in the i/p wires.
b) leave the certificate wires as it is.
c) C_w^V is obtained.

f: 1) Construct the circuit corresponding to the verifier V. The circuit is called C^V.

2) C^V takes w i/p a string x, and a corresponding certificate c.

Show: $Y \leq_P$ CIRCUIT-SAT.

Cook-Levin Theorem

SATISFIABILITY

↳ ∃x s.t. F(x)=1.

Definition: SAT $= \{\langle F \rangle \mid F$ is a satisfiable boolean formula$\}$.

CIRCUIT-SAT ≥ SAT ⇒ SAT is NP-complete.

SAT ≤p CIRCUIT-SAT.

⇓

CIRCUIT-SAT is NP-complete.

(we are doing the opposite direction)

Cook-Levin Theorem: SAT is NP-Complete.

↳ SAT is in NP

↳ SAT is NP-hard.

a) The polytime verifier for SAT.

(1) The certificate for any $\langle F \rangle \in$ SAT is a satisfying assignment for F

(2) The evaluation algorithm for CIRCUIT-SAT also works here and it runs in polytime.

Chaining Reductions

Claim: If $Y_1 \leq_P Y_2$ and $Y_2 \leq_P Y_3$ then $Y_1 \leq_P Y_3$.

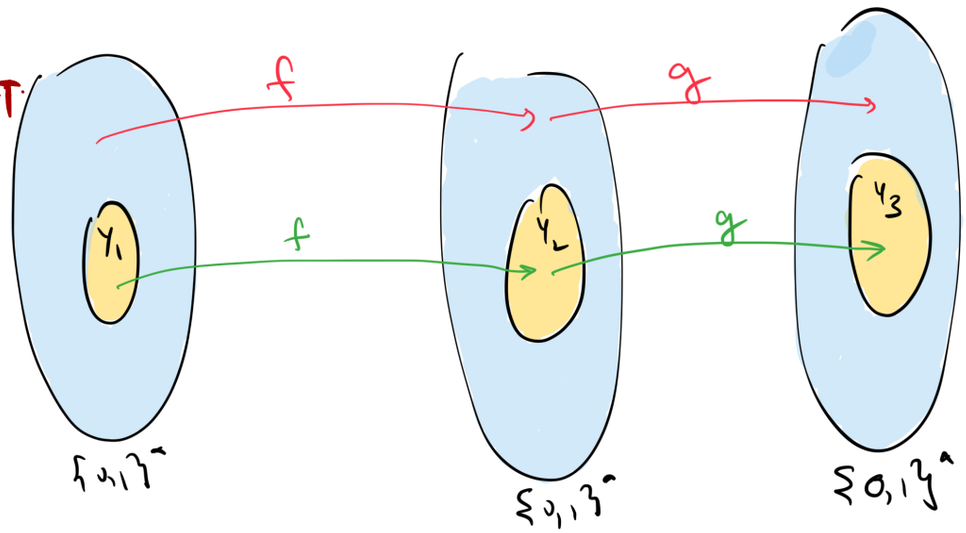
(a) How to use this to show SAT is NP-hard:

(i) CIRCUIT-SAT $\leq_p$ SAT

(ii) Any $Y \in NP$, $Y \leq_p$ CIRCUIT-SAT

So by claim, $Y \leq_p$ SAT

$\Rightarrow$ SAT is NP-hard.



Book: $Y_1 \leq_p Y_2 \Rightarrow$ (i) $\exists$ a f s.t. $x \in Y_1 \Leftrightarrow f(x) \in Y_2$,

(ii) f is computable in time $|x|^{k_1}$.

$Y_2 \leq_p Y_3 \Rightarrow$ (i) $\exists$ a function g s.t. $y \in Y_2 \Leftrightarrow g(y) \in Y_3$

(ii) g is computable in time $|y|^{k_2}$

Chaining Reductions

Claim: If $Y_1 \leq_P Y_2$ and $Y_2 \leq_P Y_3$ then $Y_1 \leq_P Y_3$.

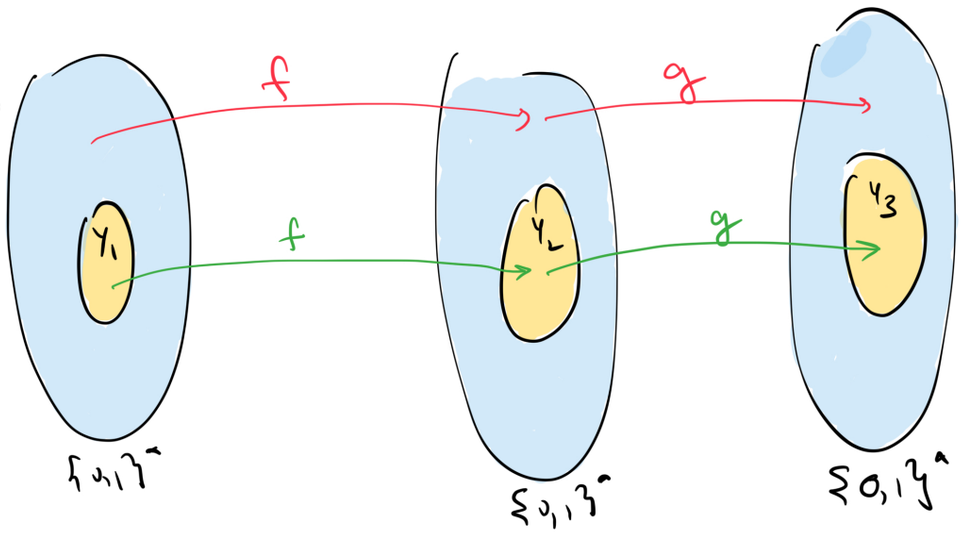
$Y_1 \longrightarrow Y_3$:

$h(x) = g(f(x))$

(i) $x \in Y_1 \Leftrightarrow f(x) \in Y_2 \Leftrightarrow g(f(x)) \in Y_3$

(Yes instances of Y_1 are mapped to Yes instances of Y_3)

No instances of Y_1 are mapped to No instances of Y_3



(ii) h can be computed in polynomial time:

(1) given x, time $f(x) = |x|^{k_1}$

$\Rightarrow |f(x)| \leq |x|^{k_1}$

(2) $g(f(x))$ takes $|f(x)|^{k_2} = |x|^{k_1 k_2}$

SATISFIABILITY

Definition: SAT $= \{\langle F \rangle \mid F$ is a satisfiable boolean formula$\}$.

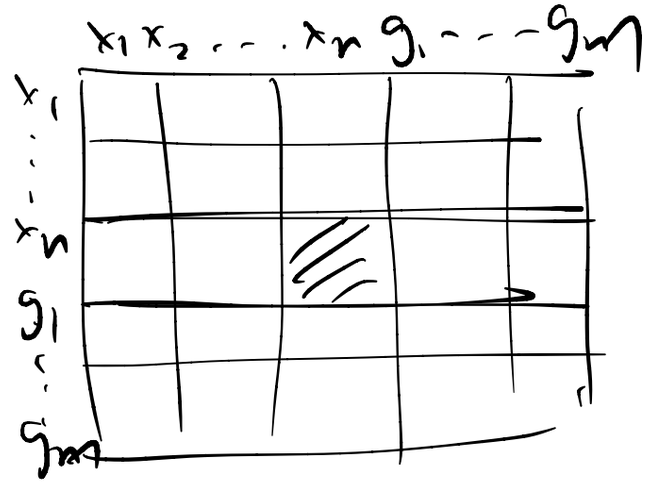
Cook-Levin Theorem: SAT is NP-Complete.

(1) CIRCUIT-SAT $\leq_p$ SAT.

(i) $\langle C \rangle \longrightarrow \langle F_C \rangle$ s.t. $\langle C \rangle \in$ CIRCUIT-SAT $\Leftrightarrow \langle F_C \rangle \in$ SAT.

(ii) $|\langle F_C \rangle| \leq |\langle C \rangle|^{k_1}$ for $k_1 \geq 0$.

$\langle C \rangle = M =$



$M(x, g) = 1 \Leftrightarrow x_i$ feeds in r to g_j.

SATISFIABILITY

Definition: SAT $= \{\langle F \rangle \mid F$ is a satisfiable boolean formula$\}$.

Cook-Levin Theorem: SAT is NP-Complete.

given a circuit C, the formula F_C will do the following:

(i) Introduce a variable x_i corresponding to every x_i and y_{g_j} corresponding to each gate g_j.



(m-n) variables

$y_g = x_1 \wedge x_2$ $\equiv$ $(\neg y_g \vee (x_1 \wedge x_2)) \wedge (\neg(x_1 \wedge x_2) \vee y_g)$

$\equiv$ $(\neg y_g \vee x_1) \wedge (\neg y_g \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee y_g)$

Identity: $P = Q$ (where P, Q are boolean variables)

$(\neg P \vee Q) \wedge (P \vee \neg Q)$

If y_g is an AND gate w/ k i/p; then, F_C will have (k+1) clauses corresponding to g.

SATISFIABILITY

$P = Q$

$\equiv (\neg P \vee Q) \wedge (P \vee \neg Q)$

Definition: SAT $= \{\langle F \rangle \mid F$ is a satisfiable boolean formula$\}$.

Cook-Levin Theorem: SAT is NP-Complete.



$y_{g'} = (x_1 \vee x_2)$

$\equiv (\neg y_{g'} \vee x_1 \vee x_2) \wedge (y_{g'} \vee \neg(x_1 \vee x_2))$

$= (\neg y_{g'} \vee x_1 \vee x_2) \wedge (y_{g'} \vee \neg x_1) \wedge (y_{g'} \vee \neg x_2)$

If an OR-gate has 6 inputs, then, in F_c you include the 7 clauses corresponding to this gate g'.

C_g = set of clauses corresponding to gate g.
(→ a variable for the o/p gate.)

$F_c = y_0 \wedge \bigwedge_{C \in C_g} C$.
g is a gate.

Check: (1) $|\langle F_c \rangle|$ poly($|\langle C \rangle|$).

(2) $\langle F_c \rangle \in SAT \Leftrightarrow \langle C \rangle \in$ CIRCUIT-SAT.

SATISFIABILITY

Definition: SAT $= \{\langle F \rangle \mid F$ is a satisfiable boolean formula$\}$.

Cook-Levin Theorem: SAT is NP-Complete.

$x_1 = v_1, x_2 = v_2 \ldots x_n = v_n$ is the satisfying assignment for C

then satisfying assignment for F_C:

a) $x_i = v_i$

b)



$y_g = x_1 \wedge x_2 \wedge \ldots \wedge x_k$

$= \wedge v_i$

c) with these values, F_C will evaluate to 1.

Summary

- We saw some explicit NP-Complete Languages:
  - CIRCUIT-SAT is NP-Complete
  - SAT is NP-Complete
  - In HW, you will prove other languages such as 3SAT, CLIQUE, etc. are NP-Complete.