Circuits
TM: A single program that works for every input length

Circuits: A program tailored to a specific input length

Motivation:

- that's what computers really are
- cryptography: attackers focus on specific key length
- more combinatorial, should be easier to understand (?)
Circuit definitions:

Gates basis (typically AND, OR, NOT)

Input and output gates

Fan-in, Fan-out

Size = number of gates (sometimes wires)

Depth = length of longest input-output path
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function computed by a circuit with \( s \) gates and fan-in \( h \). Then \( f \) is computed by a circuit with \( O(s) \) gates and fan-in 2.

Proof:

?
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function computed by a circuit with \( s \) gates and fan-in \( h \). Then \( f \) is computed by a circuit with \( O(s) \) gates and fan-in 2.

Proof:
Replace AND / OR gates with fan-in \( h \) with a binary tree of AND / OR gates.

Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function.
(1) Computable with \( s \) gates \( \Rightarrow \) computable with \( s^2 \) wires
(2) Computable with \( s \) wires \( \Rightarrow \) computable with \( O(s) \) gates

Proof:
(1) ?
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function computed by a circuit with \( s \) gates and fan-in \( h \). Then \( f \) is computed by a circuit with \( O(s) \) gates and fan-in 2.

Proof:
Replace AND / OR gates with fan-in \( h \) with a binary tree of AND / OR gates.

Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function.
(1) Computable with \( s \) gates \( \Rightarrow \) computable with \( s^2 \) wires
(2) Computable with \( s \) wires \( \Rightarrow \) computable with \( O(s) \) gates

Proof:
(1) \( s^2 \) is maximum number of wires
(2) ?
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function computed by a circuit with \( s \) gates and fan-in \( h \). Then \( f \) is computed by a circuit with \( O(s) \) gates and fan-in 2.

Proof:
Replace AND / OR gates with fan-in \( h \) with a binary tree of AND / OR gates

Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function.
(1) Computable with \( s \) gates \( \Rightarrow \) computable with \( s^2 \) wires
(2) Computable with \( s \) wires \( \Rightarrow \) computable with \( O(s) \) gates

Proof:
(1) \( s^2 \) is maximum number of wires
(2) Each wire touches \( \leq 2 \) gates
Claim: Let $f : \{0,1\}^n \to \{0,1\}$ be a function. 

$f$ is computable by a circuit of size $O(2^n)$ gates.

Proof:
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function.

\( f \) is computable by a circuit of size \( O(2^n) \) gates

Proof:

\[ V_a : f(a) = 1 \land \bigwedge_i x_i = a_i \]

There are \( \leq ? \) AND gates
Claim: Let \( f : \{0,1\}^n \rightarrow \{0,1\} \) be a function. 
\( f \) is computable by a circuit of size \( O(2^n) \) gates.

Proof:

\[ V_a : f(a) = 1 \land_i x_i = a_i \]

There are \( \leq 2^n \) AND gates.

\( x_i = a_i \) takes \( O(1) \) gates. ■

Exercise: \( \exists f : \{0,1\}^n \rightarrow \{0,1\} \) requiring circuits of size \( 2^{\Omega(n)} \)
● How do circuits compare to TM?

● Exercise: Exhibit a function $f : \{0,1\}^* \rightarrow \{0,1\}$ that is not decidable but has circuits of polynomial size.

● What about the other way around? Can poly-time TM compute more than poly-size circuits?
Poly-size circuits are at least as powerful as poly-size TM

**Theorem:** Let \( f \in \text{TIME}(t(n)) \). Then \( \forall n, f \) on inputs of length \( n \) computable with \( t^2(n) \) gates

**Corollary:** \( P \) has polynomial-size circuits (\( P \subseteq P/poly \))

**Beginning of proof of theorem:**
Assume w.l.o.g. TM for \( f \) writes output on 1st cell.

We encode configs of TM using symbols which encode a tape symbol, whether the head is there, and the state

So we think of \( 0 0 q_5 1 2 \) as \( 0 0 (q_5 1) 2 \)
where \( (q_5 1) \) is one symbol
Fact: \( \exists \) circuit of \( O(t(n)) \) gates which given
n symbols of a configuration \( C \) produces
the \( n \) symbols of the next configuration \( C' \).

Proof: A variant of locality of computation

Each symbol of \( C' \) is a function of ?
Fact: \( \exists \) circuit of \( O(t(n)) \) gates which given \( n \) symbols of a configuration \( C \) produces the \( n \) symbols of the next configuration \( C' \).

Proof: A variant of locality of computation

Each symbol of \( C' \) is a function of three symbols of \( C \). As we saw, that function is doable by a circuit of size \( ? \).
Fact: ∃ circuit of $O(t(n))$ gates which given $n$ symbols of a configuration $C$ produces the $n$ symbols of the next configuration $C'$.

Proof: A variant of locality of computation

Each symbol of $C'$ is a function of three symbols of $C$. As we saw, that function is doable by a circuit of size $O(1)$. ■

Proof of theorem:
Fact: \( \exists \) circuit of \( O(t(n)) \) gates which given \( n \) symbols of a configuration \( C \) produces the \( n \) symbols of the next configuration \( C' \).

Proof: A variant of locality of computation

Each symbol of \( C' \) is a function of three symbols of \( C \). As we saw, that function is doable by a circuit of size \( O(1) \).

Proof of theorem:

Pile up \( t(n) \) copies of circuit from Fact

Total size = \( O(t^2(n)) \)

- Size can be improved to \( O(t(n) \log^c t(n)) \)
• **Def:** Circuit-SAT := \{ C : C is a circuit : \exists y : C(y) = 1 \}

• **Claim:** Circuit-SAT is NP-complete

• **Proof:** Circuit-SAT ∈ NP because ?
• Def: Circuit-SAT := \{ C : C is a circuit : \exists y : C(y) = 1 \}
• Claim: Circuit-SAT is NP-complete
• Proof: Circuit-SAT \in NP because given C and y we can compute C(y) in time polynomial in |C|

Suppose now Circuit-SAT \in P. We show P = NP.

Let L \in NP with corresponding machine M(x,y).

Here's a polynomial-time algorithm for L: Given x,
**Def:** Circuit-SAT := \{ C : C is a circuit : \exists y : C(y) = 1 \}

**Claim:** Circuit-SAT is NP-complete

**Proof:** Circuit-SAT \( \in \) NP because given \( C \) and \( y \) we can compute \( C(y) \) in time polynomial in \( |C| \)

Suppose now Circuit-SAT \( \in \) P. We show P = NP.

Let \( L \in \) NP with corresponding machine \( M(x,y) \).

Here's a polynomial-time algorithm for \( L \): Given \( x \),

1. Construct following previous theorem circuit \( C \) for the function \( y \rightarrow M(x,y) \).
2. This circuit has size poly(|\( x \)|) because ?
Def: Circuit-SAT := \{ C : C is a circuit : \exists y : C(y) = 1 \}

Claim: Circuit-SAT is NP-complete

Proof: Circuit-SAT \in NP because given C and y we can compute C(y) in time polynomial in |C|

Suppose now Circuit-SAT \in P. We show P = NP.

Let L \in NP with corresponding machine M(x,y).

Here's a polynomial-time algorithm for L: Given x,
Construct following previous theorem circuit C for the function y \rightarrow M(x,y).
This circuit has size \text{poly}(|x|) because M runs in polynomial time and |y| = \text{poly}(|x|)
Use poly-time algorithm for Circuit-SAT on C.
Corollary: 3SAT is NP-complete.

Proof:

We just need to reduce Circuit-SAT to 3SAT.

Idea: replace each gate in the circuit with $O(1)$ clauses

Exercise.
• Recall $P \subseteq \text{poly-size circuits (aka } P/\text{poly})$

• Believed $\text{NP NOT } \subseteq P/\text{poly}$, which implies $P \neq \text{NP}$.

• Leading goal: prove $\text{NP NOT IN P/poly} \Rightarrow P \neq \text{NP}$

• We cannot even show $\text{NP NOT in circuits of size } O(n)$

• We cannot even show $\text{EXP NOT in P/poly}$
Exercise:

● Prove \( \exists c \ \forall k, \sum_c P \) does not have circuits of size \( n^k \)

● Prove \( PH \subseteq EXP \)

● So \( \forall k, EXP \) does not have circuits of size \( n^k \)

Open:

● Does NP have circuits of size \( O(n) \)?
Exercise:

- Def.: \( E := \text{TIME}(2^{O(n)}) \)
- Open: Does \( E \) have circuits of size \( O(n) \)?
- Prove \( E \subseteq \text{P/poly} \leftrightarrow \text{EXP} \subseteq \text{P/poly} \)
• **Theorem:** NP ⊆ P/poly → PH = \( \sum_2 P \)

• **Proof:** We'll show the \( \Pi_2 P \)-complete problem

\[
L := \{ \phi : \forall u \in \{0,1\}^{|\phi|} \exists v \in \{0,1\}^{|\phi|} : \phi(u,v) = 1 \} \in \text{????}
\]

Where do we need to place this, to get PH = \( \sum_2 P \)?
• **Theorem:** $\text{NP} \subseteq \text{P/poly} \rightarrow \text{PH} = \sum_2 \text{P}$

• **Proof:** We'll show the $\prod_2 \text{P}$ - complete problem

$L := \{ \varphi : \forall u \in \{0,1\}^{\lvert \varphi \rvert} \exists v \in \{0,1\}^{\lvert \varphi \rvert} : \varphi(u,v) = 1 \} \in \sum_2 \text{P}$

$\text{NP} \subseteq \text{P/poly} \rightarrow \{ (\varphi, u) : \exists v \in \{0,1\}^{\lvert \varphi \rvert} : \varphi(u,v) = 1 \} \in \Sigma$
• Theorem: \( \text{NP} \subseteq \text{P/poly} \rightarrow \text{PH} = \sum_2 \text{P} \)

• Proof: We'll show the \( \prod_2 \text{P} \) - complete problem

\( L := \{ \varphi : \forall u \in \{0,1\}^{|\varphi|} \exists v \in \{0,1\}^{|\varphi|} : \varphi(u,v) = 1 \} \in \sum_2 \text{P} \)

\( \text{NP} \subseteq \text{P/poly} \rightarrow \{ (\varphi, u) : \exists v \in \{0,1\}^{|\varphi|} : \varphi(u,v) = 1 \} \in \text{P/poly} \)

We can guess this circuit, but is it the right one?

How do you turn the circuit into one whose output you can check by yourself, i.e., in poly-time?
• **Theorem:** $\text{NP} \subseteq \text{P/poly} \rightarrow \text{PH} = \sum_2 \text{P}$

• **Proof:** We'll show the $\prod_2 \text{P}$ - complete problem

$L := \{ \phi : \forall u \in \{0,1\}^{\lvert \phi \rvert} \exists v \in \{0,1\}^{\lvert \phi \rvert} : \phi(u,v) = 1 \} \in \sum_2 \text{P}$

$\text{NP} \subseteq \text{P/poly} \rightarrow \{ (\phi, u) : \exists v \in \{0,1\}^{\lvert \phi \rvert} : \phi(u,v) = 1 \} \in \text{P/poly}$

We can guess this circuit, but is it the right one?

Note $\text{NP} \subseteq \text{P/poly} \rightarrow$ in $\text{P/poly}$ can compute a satisfying assignment $v$ if one exists.

$\phi \in L \leftrightarrow \exists$ poly-size circuit $C$ : ?
• Theorem: $\text{NP} \subseteq \text{P/poly} \rightarrow \text{PH} = \sum_2 \text{P}$

• Proof: We'll show the $\prod_2 \text{P}$ - complete problem

$L := \{ \varphi : \forall u \in \{0,1\}^{|\varphi|} \exists v \in \{0,1\}^{|\varphi|} : \varphi(u,v) = 1 \} \in \sum_2 \text{P}$

$\text{NP} \subseteq \text{P/poly} \rightarrow \{ (\varphi, u) : \exists v \in \{0,1\}^{|\varphi|} : \varphi(u,v) = 1 \} \in \text{P/poly}$

We can guess this circuit, but is it the right one?

Note $\text{NP} \subseteq \text{P/poly} \rightarrow$ in $\text{P/poly}$ can compute a satisfying assignment $v$ if one exists.

$\varphi \in L \leftrightarrow \exists \text{ poly-size circuit } C : \forall u \in \{0,1\}^{|\varphi|}, \varphi(u, \text{ ??????? }) = 1$
• **Theorem:** $\text{NP} \subseteq \text{P/poly} \rightarrow \text{PH} = \sum_2 \text{P}$

• **Proof:** We'll show the $\prod_2 \text{P}$-complete problem

$L := \{ \phi : \forall u \in \{0,1\}^{\text{|}\phi\text{|}} \exists v \in \{0,1\}^{\text{|}\phi\text{|}} : \phi(u,v) = 1 \} \in \sum_2 \text{P}$

$\text{NP} \subseteq \text{P/poly} \rightarrow \{ (\phi, u) : \exists v \in \{0,1\}^{\text{|}\phi\text{|}} : \phi(u,v) = 1 \} \in \text{P/poly}$

We can guess this circuit, but is it the right one?

Note $\text{NP} \subseteq \text{P/poly} \rightarrow$ in $\text{P/poly}$ can compute a satisfying assignment $v$ if one exists.

$\phi \in L \iff \exists \text{poly-size circuit } C : \forall u \in \{0,1\}^{\text{|}\phi\text{|}}, \phi(u, C(\phi, u)) = 1$

Note $\phi(u, C(\phi, u))$ is computable in poly-time.