Computational Complexity

Lecture 7: Non-Uniform Complexity

Ronald de Haan
me@ronalddehaan.eu

University of Amsterdam

April 23, 2024
What will we do today?

- Non-uniform complexity
- Circuit complexity
- TMs that take advice
- The Karp-Lipton Theorem
Non-uniformity

- “Uniform”: the algorithm is the same, regardless of the input size

  vs.

- “Non-uniform”: there can be different algorithms for different input sizes
- Boolean circuits are very similar to propositional formulas

- Directed acyclic graphs (instead of trees)

- We view binary strings as truth assignments

- Example: \((\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3), x = 010,\) and \(\alpha_x = \{ x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0 \}\)
Boolean circuits are very similar to propositional formulas.

- Directed acyclic graphs (instead of trees)

We view binary strings as truth assignments.

Example: \((\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3), x = 010,\) and \(\alpha_x = \{x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0\}\)
Boolean circuits are very similar to propositional formulas

Directed acyclic graphs (instead of trees)

We view binary strings as truth assignments

Example: \((\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3), x = 010, \) and \(\alpha_x = \{x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0\}\)
Boolean circuits are very similar to propositional formulas.

Directed acyclic graphs (instead of trees)

We view binary strings as truth assignments.

Example: $(\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3)$, $x = 010$, and $\alpha_x = \{x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0\}$
Boolean circuits are very similar to propositional formulas.

Directed acyclic graphs (instead of trees)

We view binary strings as truth assignments.

Example: \((\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3), x = 010, \) and \(\alpha_x = \{ x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0 \} \)
Definition (Circuits)

An \( n \)-input single-output Boolean circuit \( C \) is a directed acyclic graph with:
- \( n \) sources (nodes with no incoming edges), labelled 1 to \( n \), and
- one sink (a node with no outgoing edges).

All non-source vertices are called gates, and are labelled with \( \land \), \( \lor \), or \( \neg \):
- \( \land \)-gates and \( \lor \)-gates have in-degree 2 (exactly two incoming edges),
- \( \neg \)-gates have in-degree 1 (exactly one incoming edge).

If \( C \) is an \( n \)-input single-output Boolean circuit and \( x \in \{0, 1\}^n \) is a string, then the output \( C(x) \) of \( C \) on \( x \) is defined by plugging in \( x \) in the source nodes and applying the operators of the gates, and taking for \( C(x) \) the resulting value in \( \{0, 1\} \) of the sink gate.
Definition (Circuit families)

Let $t : \mathbb{N} \to \mathbb{N}$ be a function. A $t(n)$-size circuit family is a sequence $\{C_n\}_{n \in \mathbb{N}}$ of Boolean circuits, where each $C_n$ has $n$ inputs and a single output, and $|C_n| \leq t(n)$ for each $n \in \mathbb{N}$.

Definition (SIZE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{N}$ be a function. A language $L \subseteq \{0, 1\}^*$ is in SIZE($t(n)$) if there exists a constant $c \in \mathbb{N}$ and a $(c \cdot t(n))$-size circuit family $\{C_n\}_{n \in \mathbb{N}}$ such that for each $x \in \{0, 1\}^*$:

$$x \in L \quad \text{if and only if} \quad C_n(x) = 1, \text{ where } n = |x|.$$
The complexity class $P/poly$

**Definition ($P/poly$)**

$$P/poly = \bigcup_{c \geq 1} \text{SIZE}(n^c).$$

- In other words, $P/poly$ is the class of all decision problems that can be decided by a polynomial-size circuit family.
\( P \subseteq P/\text{poly} \)

- (We consider only decision problems \( L \subseteq \{0, 1\}^* \)—i.e., binary alphabets.)

**Theorem**

\( P \subseteq P/\text{poly} \).

- Main idea:
  - Like in the proof of the Cook-Levin Theorem, we encode polynomial-time computation in logic
  - Instead of using new, fresh variables we use nodes in the Boolean circuit (to encode tape contents, tape head positions, etc)
(We consider only decision problems $L \subseteq \{0,1\}^*$—i.e., binary alphabets.)

**Theorem**

$P \subseteq P/\text{poly}$.

**Main idea:**

- Like in the proof of the Cook-Levin Theorem, we encode polynomial-time computation in logic.
- Instead of using new, fresh variables we use nodes in the Boolean circuit (to encode tape contents, tape head positions, etc).

- In fact, $P \subsetneq P/\text{poly}$ (you will show this in the homework).
Turing machines that take advice

- We can characterize $\mathsf{P/poly}$ (or more generally, non-uniform complexity classes) also using TMs

- The algorithm might differ per input size $n$, so we will have to give the TM something that depends only on the input size

- This is called advice
Advice characterization of $P/poly$

**Definition (TIME($t(n))/a(n)$)**

Let $t, a : \mathbb{N} \to \mathbb{N}$ be functions. The class DTIME($t(n))/a(n)$ of languages decidable by $O(t(n))$-time Turing machines with $a(n)$ bits of advice contains every decision problem $L \subseteq \{0, 1\}^*$ such that:

- there exists a sequence $\{\alpha_n\}_{n \in \mathbb{N}}$ with $\alpha_n \in \{0, 1\}^{a(n)}$ for each $n \in \mathbb{N}$ and an $O(t(n))$-time deterministic Turing machine $M$ such that for each $x \in \{0, 1\}^*$:

$$x \in L \text{ if and only if } M(x, \alpha_n) = 1, \text{ where } n = |x|.$$
Theorem

\[ P/\text{poly} = \bigcup_{c,d \geq 1} \text{DTIME}(n^c)/n^d. \]
Advice characterization of $P/poly$ (ct'd)

**Theorem**

$$P/poly = \bigcup_{c,d \geq 1} \text{DTIME}(n^c)/n^d.$$ 

- Main idea (for “$\subseteq$”):
  - Use a description of $C_n$ as $\alpha_n$, and then compute $C_n(x)$ in polynomial time.
  - The computation of $M(x,\alpha_n)$ on inputs $x \in \{0,1\}^n$ can be encoded as a polynomial-size circuit $D_n(\cdot,\alpha_n)$, using ideas from the proof of the Cook-Levin Theorem.
  - The circuit $C_n$ is $D_n$ with $\alpha_n$ "hardwired in".
Advice characterization of P/poly (ct’d)

Theorem

\[ \text{P/poly} = \bigcup_{c,d \geq 1} \text{DTIME}(n^c)/n^d. \]

- Main idea (for “\( \subseteq \)“):
  - Use a description of \( C_n \) as \( \alpha_n \), and then compute \( C_n(x) \) in polynomial time
Advice characterization of P/poly (ct’d)

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P/poly = \bigcup_{c,d \geq 1} \text{DTIME}(n^c)/n^d.$</td>
</tr>
</tbody>
</table>

- **Main idea (for “⊆”):**
  - Use a description of $C_n$ as $\alpha_n$, and then compute $C_n(x)$ in polynomial time

- **Main idea (for “⊇”):**
Advice characterization of P/poly (ct’d)

### Theorem

\[
P/poly = \bigcup_{c,d \geq 1} \text{DTIME}(n^c)/n^d.
\]

- **Main idea (for “\(\subseteq\)”:**
  - Use a description of \(C_n\) as \(\alpha_n\), and then compute \(C_n(x)\) in polynomial time

- **Main idea (for “\(\supseteq\)”:**
  - The computation of \(M(x, \alpha_n)\) on inputs \(x \in \{0,1\}^n\) can be encoded as a polynomial-size circuit \(D_n(\cdot, \alpha_n)\), using ideas from the proof of the Cook-Levin Thm
  - The circuit \(C_n\) is \(D_n\) with \(\alpha_n\) “hardwired in”
**Definition**

A circuit family \( \{ C_n \}_{n \in \mathbb{N}} \) is **P-uniform** if there exists a polynomial-time deterministic TM that on input \( 1^n \) outputs a description of \( C_n \), for each \( n \in \mathbb{N} \).

**Theorem**

* A decision problem \( L \subseteq \{0,1\}^* \) is in P if and only if decidable by a P-uniform circuit family \( \{ C_n \}_{n \in \mathbb{N}} \).
Question: is SAT decidable by polynomial-size circuits (is it in P/poly)?

- Perhaps by allowing the algorithm to change per input size, this might work.
The Karp-Lipton Theorem

- Question: is SAT decidable by polynomial-size circuits (is it in P/poly)?
  - Perhaps by allowing the algorithm to change per input size, this might work
  - The answer: No (assuming that the PH does not collapse)

**Theorem (Karp, Lipton 1980)**

If \( \text{NP} \subseteq \text{P/poly} \), then \( \Sigma_2^p = \Pi_2^p \).
Proof of the Karp-Lipton Thm

The general argument

- Suppose that $\text{NP} \subseteq \text{P/poly}$.
- We show that then $\Pi^p_2 \subseteq \Sigma^p_2$, by showing $\Pi_2 \text{SAT} \in \Sigma^p_2$. 

We use the following lemma to swap the order of the quantifiers:

**Lemma**

If $\text{NP} \subseteq \text{P/poly}$, then there exists a polynomial-time algorithm that:
- takes polynomial-length advice, and
- given a propositional formula $\phi$:
  - if $\phi$ is unsatisfiable, it outputs $0$;
  - if $\phi$ is satisfiable, it outputs a satisfying truth assignment $\alpha$ for $\phi$.

Idea behind the proof of the lemma: use self-reducibility of SAT.
Proof of the Karp-Lipton Thm

The general argument

- Suppose that \( \text{NP} \subseteq \text{P/poly} \).
- We show that then \( \Pi_2^p \subseteq \Sigma_2^p \), by showing \( \Pi_2\text{SAT} \in \Sigma_2^p \).
- We use the following lemma to swap the order of the quantifiers:

**Lemma**

*If* \( \text{NP} \subseteq \text{P/poly} \), *then there exists a polynomial-time algorithm that:*

- *takes polynomial-length advice, and*
- *given a propositional formula* \( \varphi \):
  - *if* \( \varphi \) *is unsatisfiable, it outputs* 0;
  - *if* \( \varphi \) *is satisfiable, it outputs a satisfying truth assignment* \( \alpha \) *for* \( \varphi \).
Proof of the Karp-Lipton Thm

The general argument

- Suppose that $\text{NP} \subseteq \text{P/poly}$.

- We show that then $\Pi_2^p \subseteq \Sigma_2^p$, by showing $\Pi_2\text{SAT} \in \Sigma_2^p$.

- We use the following lemma to swap the order of the quantifiers:

Lemma

If $\text{NP} \subseteq \text{P/poly}$, then there exists a polynomial-time algorithm that:

- takes polynomial-length advice, and
- given a propositional formula $\varphi$:
  - if $\varphi$ is unsatisfiable, it outputs 0;
  - if $\varphi$ is satisfiable, it outputs a satisfying truth assignment $\alpha$ for $\varphi$.

- Idea behind the proof of the lemma: use self-reducibility of SAT.
Proof of the Karp-Lipton Thm

Completing the proof

Take an arbitrary instance of $\Pi_2$SAT: $\varphi = \forall \overline{u}. \exists \overline{v}. \psi(\overline{u}, \overline{v})$. 
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2$SAT: $\varphi = \forall u. \exists v. \psi(u, v)$.

- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $M$. 
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2$SAT: $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v})$.
- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $M$.
- $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v}) \in \Pi_2$SAT if and only if for all $\bar{z} \in \{0, 1\}^m$, $\psi[\bar{u} \mapsto \bar{z}] \in \text{SAT}$. 
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2\text{SAT}$: $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v})$.

- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $M$.

- $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v}) \in \Pi_2\text{SAT}$ if and only if for all $\bar{z} \in \{0, 1\}^m$, $\psi[\bar{u} \mapsto \bar{z}] \in \text{SAT}$.

- This is the case if and only if:

  there exists some $\bar{w} \in \{0, 1\}^{q(n)}$ such that

  for all $\bar{z} \in \{0, 1\}^m$ $M$ uses $\bar{w}$ as advice to output the assignment $\gamma$

  on input $\psi[\bar{u} \mapsto \bar{z}]$ and $\gamma$ satisfies $\psi[\bar{u} \mapsto \bar{z}]$
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2$SAT: $\varphi = \forall u. \exists v. \psi(u, v)$.
- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $M$.
- $\varphi = \forall u. \exists v. \psi(u, v) \in \Pi_2$SAT if and only if for all $z \in \{0, 1\}^m$, $\psi[u \mapsto z] \in \text{SAT}$.
- This is the case if and only if:

  there exists some $\bar{w} \in \{0, 1\}^{q(n)}$ such that

  for all $z \in \{0, 1\}^m$

  $M$ uses $\bar{w}$ as advice to output the assignment $\gamma$ on input $\psi[u \mapsto z]$ and $\gamma$ satisfies $\psi[u \mapsto z]$  

Key: we check that $\gamma$ is correct; because we don’t know whether $\bar{w}$ is the right advice
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2$SAT: $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v})$.

- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $M$.

- $\varphi = \forall \bar{u}. \exists \bar{v}. \psi(\bar{u}, \bar{v}) \in \Pi_2$SAT if and only if for all $\bar{z} \in \{0, 1\}^m$, $\psi[\bar{u} \mapsto \bar{z}] \in \text{SAT}$.

- This is the case if and only if:
  
  - $\exists$ there exists some $\bar{w} \in \{0, 1\}^{q(n)}$ such that
  
  - $\forall$ for all $\bar{z} \in \{0, 1\}^m$

- $M$ uses $\bar{w}$ as advice to output the assignment $\gamma$ on input $\psi[\bar{u} \mapsto \bar{z}]$ and $\gamma$ satisfies $\psi[\bar{u} \mapsto \bar{z}]$

**Key:** we check that $\gamma$ is correct; because we don’t know whether $\bar{w}$ is the right advice
Proof of the Karp-Lipton Thm

Completing the proof

- Take an arbitrary instance of $\Pi_2\text{SAT}$: $\varphi = \forall \overline{u}. \exists \overline{v}. \psi(\overline{u}, \overline{v})$.

- Let $q$ be the polynomial bounding the size of the advice $\{\alpha_n\}_{n \in \mathbb{N}}$ that can be used to compute satisfying assignments for SAT, in polynomial time with TM $\mathbb{M}$.

- $\varphi = \forall \overline{u}. \exists \overline{v}. \psi(\overline{u}, \overline{v}) \in \Pi_2\text{SAT}$ if and only if for all $\overline{z} \in \{0, 1\}^m$, $\psi[\overline{u} \mapsto \overline{z}] \in \text{SAT}$.

- This is the case if and only if:
  - $\exists$ there exists some $\overline{w} \in \{0, 1\}^{q(n)}$ such that
  - $\forall$ for all $\overline{z} \in \{0, 1\}^m$

  poly $\mathbb{M}$ uses $\overline{w}$ as advice to output the assignment $\gamma$ on input $\psi[\overline{u} \mapsto \overline{z}]$ and $\gamma$ satisfies $\psi[\overline{u} \mapsto \overline{z}]$

  Key: we check that $\gamma$ is correct; because we don’t know whether $\overline{w}$ is the right advice

- Thus, $\Pi_2\text{SAT} \in \Sigma_2^p$, and therefore $\Pi_2^p = \Sigma_2^p$. 
- Non-uniform complexity
- Circuit complexity
- TMs that take advice
- The Karp-Lipton Theorem: if $NP \subseteq P/poly$, then $\Sigma_2^p = \Pi_2^p$
Next time

- A “breather”
- Time to reflect on what we’ve done so far
- Requests for things to recap?