Lecture 1 - Complexity, Turing Machines, Time Hierarchy

Rafael Oliveira
rafael.oliveira.teaching@gmail.com
University of Waterloo

CS 860 - Graduate Complexity Theory
Fall 2022

Overview

- What is Complexity Theory
- Turing Machines Refresher
- Time Hierarchy

What is Complexity Theory?

- Objectives of Theoretical Computer Science:
  1. understand how different models of computation relate to one another
  2. classify computational problems according to the amount of resources[1] needed to solve them

[1] time, memory, communication, randomness, entanglement, ...

What is Complexity Theory?

- Objectives of Theoretical Computer Science:
  1. understand how different models of computation relate to one another
  2. classify computational problems according to the amount of resources[1] needed to solve them
- 3 basic tasks
  1. upper bounds (algorithms)
  2. lower bounds (impossibility)
  3. reductions (hierarchy/equivalence of problems/models)

[1] time, memory, communication, randomness, entanglement, ...

What is Complexity Theory?

- Objectives of Theoretical Computer Science:
  1. understand how different models of computation relate to one another
  2. classify computational problems according to the amount of resources[1] needed to solve them
- 3 basic tasks
  1. upper bounds (algorithms)
  2. lower bounds (impossibility)
  3. reductions (hierarchy/equivalence of problems/models)
- Many connections!

See Math and Computation book!

[1] time, memory, communication, randomness, entanglement, ...

Uniform Model: Turing Machines

"one algorithm to handle them (the inputs) all"

Deterministic computation.

Uniform Model: Turing Machines

Definition (Deterministic Turing Machines)

A Deterministic Turing Machine (TM) M is described by a tuple $(\Sigma, \Gamma, \mathcal{Q}, \delta)$ such that:

- $\Sigma, \Gamma, \mathcal{Q}$ are finite sets
- $\mathcal{Q}$ is the set of states, with $q_{start}, q_{halt} \in \mathcal{Q}$
- Σ is the input (and output) alphabet $(\triangleright, \Box \notin \Sigma)$
- Γ is the tape alphabet, with $\triangleright, \Box \in \Gamma, \Sigma \subset \Gamma$
  - $\Box$ is the blank symbol
  - $\triangleright$ is the start symbol
- $\delta : \mathcal{Q} \times \Gamma \to \mathcal{Q} \times \Gamma \times \{L, R\}$ (transition function)

Uniform Model: Turing Machines

Definition (Deterministic Turing Machines)

A Deterministic Turing Machine (TM) M is described by a tuple $(\Sigma, \Gamma, \mathcal{Q}, \delta)$ such that:

- $\Sigma, \Gamma, \mathcal{Q}$ are finite sets
- $\mathcal{Q}$ is the set of states, with $q_{start}, q_{halt} \in \mathcal{Q}$
- Σ is the input (and output) alphabet $(\triangleright, \Box \notin \Sigma)$
- Γ is the tape alphabet, with $\triangleright, \Box \in \Gamma, \Sigma \subset \Gamma$
  - $\Box$ is the blank symbol
  - $\triangleright$ is the start symbol
- $\delta : \mathcal{Q} \times \Gamma \to \mathcal{Q} \times \Gamma \times \{L, R\}$ (transition function)

Since can simulate multiple tapes with one tape, let's assume we have 3 tapes: input tape (read-only), work tape, output tape.

$$\text{Then } \delta : \mathcal{Q} \times \Gamma^3 \to \mathcal{Q} \times \Gamma \times \Sigma \times \{L, R\}^3.$$

Uniform Model: Turing Machines

Definition (Non-deterministic Turing Machines)

A Non-deterministic Turing Machine (NTM) M is described by a tuple $(\Sigma, \Gamma, \mathcal{Q}, \delta_0, \delta_1)$ such that:

- $\Sigma, \Gamma, \mathcal{Q}$ are finite sets
- $\mathcal{Q}$ is the set of states, with $q_{start}, q_{halt} \in \mathcal{Q}$
- Σ is the input (and output) alphabet $(\triangleright, \Box \notin \Sigma)$
- Γ is the tape alphabet, with $\triangleright, \Box \in \Gamma, \Sigma \subset \Gamma$
  - $\Box$ is the blank symbol
  - $\triangleright$ is the start symbol
- $\delta_b : \mathcal{Q} \times \Gamma \to \mathcal{Q} \times \Gamma \times \{L, R\}$ (transition functions)

Uniform Model: Turing Machines

Definition (Non-deterministic Turing Machines)

A Non-deterministic Turing Machine (NTM) M is described by a tuple $(\Sigma, \Gamma, \mathcal{Q}, \delta_0, \delta_1)$ such that:

- $\Sigma, \Gamma, \mathcal{Q}$ are finite sets
- $\mathcal{Q}$ is the set of states, with $q_{start}, q_{halt} \in \mathcal{Q}$
- Σ is the input (and output) alphabet $(\triangleright, \square \notin \Sigma)$
- Γ is the tape alphabet, with $\triangleright, \square \in \Gamma, \Sigma \subset \Gamma$
  - $\square$ is the blank symbol
  - $\triangleright$ is the start symbol
- $\delta_b : \mathcal{Q} \times \Gamma \to \mathcal{Q} \times \Gamma \times \{L, R\}$ (transition functions)

We can similarly assume that we have 3 tapes here.

For NTM Mn input x, and $y \in \{0,1\}^*$, let $M(x, y)$ be the execution of M on input x where at the i^{th} step we select transition function δ_{y_i}.

Computing a Function and Running Time

Definition

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a TM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, we have $M(x)$ outputs $f(x)$
- M computes f in T-time if for all $x \in \{0,1\}^*$, $M(x)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

Computing a Function and Running Time

Definition

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a TM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, we have $M(x)$ outputs $f(x)$
- M computes f in T-time if for all $x \in \{0,1\}^*$, $M(x)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

1. Given function $T : \mathbb{N} \to \mathbb{N}$, define DTIME$(T)$ as the set of languages $L \subseteq \{0,1\}^*$ such that
   - there is TM M and c constant such that $M(x)$ halts in $c \cdot T(|x|)$ time
   - $M(x) = 1 \Leftrightarrow x \in L$ (M decides L)

Computing a Function and Running Time

Definition

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a TM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, we have $M(x)$ outputs $f(x)$
- M computes f in T-time if for all $x \in \{0,1\}^*$, $M(x)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

1. Given function $T : \mathbb{N} \to \mathbb{N}$, define DTIME$(T)$ as the set of languages $L \subseteq \{0,1\}^*$ such that
   - there is TM M and c constant such that $M(x)$ halts in $c \cdot T(|x|)$ time
   - $M(x) = 1 \Leftrightarrow x \in L$ (M decides L)
2. $$\mathsf{P} := \bigcup_{c \in \mathbb{N}} \mathsf{DTIME}(n^c)$$

Computing a Function and Running Time

Definition (Non-deterministic setting)

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a NTM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x,y)$ outputs $f(x)$
- M computes f in T-time if for each $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x,y)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

Computing a Function and Running Time

Definition (Non-deterministic setting)

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a NTM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x,y)$ outputs $f(x)$
- M computes f in T-time if for each $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x,y)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

1. Given $T : \mathbb{N} \to \mathbb{N}$, define NTIME$(T)$ as the set of languages $L \subseteq \{0,1\}^*$ such that
   - there is NTM M and c constant such that for all $x, y \in \{0,1\}^*$, $M(x,y)$ halts in $c \cdot T(|x|)$ time
   - $\exists y \in \{0,1\}^*$ s.t. $M(x,y) = 1 \Leftrightarrow x \in L$ (M decides L)

Computing a Function and Running Time

Definition (Non-deterministic setting)

Given functions $f : \{0,1\}^* \to \{0,1\}^*$ and $T : \mathbb{N} \to \mathbb{N}$, and a NTM M, we say that:

- M computes f if for all $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x, y)$ outputs $f(x)$
- M computes f in T-time if for each $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ s.t. $M(x, y)$ takes at most $T(|x|)$ steps and outputs $f(x)$.

1. Given $T : \mathbb{N} \to \mathbb{N}$, define $\mathsf{NTIME}(T)$ as the set of languages $L \subseteq \{0,1\}^*$ such that
   - there is NTM M and c constant such that for all $x, y \in \{0,1\}^*$, $M(x, y)$ halts in $c \cdot T(|x|)$ time
   - $\exists y \in \{0,1\}^*$ s.t. $M(x, y) = 1 \Leftrightarrow x \in L$ (M decides L)
2. $$\mathsf{NP} := \bigcup_{c \in \mathbb{N}} \mathsf{NTIME}(n^c)$$

Time Constructible Functions

Definition

A function $T : \mathbb{N} \to \mathbb{N}$ is time constructible if $T(n) \geq n$ and there is a TM M which computes $x \mapsto b(T(|x|))$ in time $T(|x|)$.

Time Constructible Functions

Definition

A function $T : \mathbb{N} \to \mathbb{N}$ is time constructible if $T(n) \geq n$ and there is a TM M which computes $x \mapsto b(T(|x|))$ in time $T(|x|)$.

- Examples: $n, n \log n, n^{10}, 2^n$

Time Constructible Functions

Definition

A function $T : \mathbb{N} \to \mathbb{N}$ is time constructible if $T(n) \geq n$ and there is a TM M which computes $x \mapsto b(T(|x|))$ in time $T(|x|)$.

- Examples: $n, n \log n, n^{10}, 2^n$
- restriction $T(n) \geq n$ is to allow algorithm to read its input

Universal Turing Machines

"one algorithm to rule them (algorithms) all"

Theorem ([Hennie Stearns, 1966])

There is a TM $\mathcal{U}$ such that for every $\alpha, x \in \{0,1\}^*$

$$\mathcal{U}(\alpha, x) = M_\alpha(x),$$

moreover, there is a function $C := C_\alpha$ (depending only on M_α's description), such that if M_α halts on x within T steps, then $\mathcal{U}(\alpha, x)$ halts within $C \cdot T \log T$ steps.

Universal Turing Machines

"one algorithm to rule them (algorithms) all"

Theorem (**[Hennie Stearns, 1966]**)

There is a TM $\mathcal{U}$ such that for every $\alpha, x \in \{0,1\}^*$

$$\mathcal{U}(\alpha, x) = M_\alpha(x),$$

moreover, there is a function $C := C_\alpha$ (depending only on M_α's description), such that if M_α halts on x within T steps, then $\mathcal{U}(\alpha, x)$ halts within $C \cdot T \log T$ steps.

- Note that C_α depends on the description of M_α, not necessarily the string α
  For instance α and $\beta := \alpha \circ 1^3$ (by our discussion) are such that $M_\alpha = M_\beta$ and thus $C_\alpha = C_\beta$.

1. Can also prove that the above is true for NTMs
2. For NTMs, one can actually get simulation runtime of $C \cdot T$

- What is Complexity Theory
- Turing Machines Refresher
- Time Hierarchy

Deterministic Time Hierarchy

Theorem (**[Hartmanis Stearns, 1965]**)

If f, g are time-constructible functions such that $f(n) \log f(n) = o(g(n))$ for all $n \in \mathbb{N}$, then

$$\mathit{DTIME}(f(n)) \subsetneq \mathit{DTIME}(g(n))$$

Deterministic Time Hierarchy

- We will prove: $\mathsf{DTIME}(n) \subsetneq \mathsf{DTIME}(n^2)$

Deterministic Time Hierarchy

- We will prove: DTIME$(n) \subsetneq$ DTIME(n^2)

1. Let $\mathcal{U}$ be UTM from **[Hennie Stearns, 1966]**
2. Consider TM D: on input x
   - run $M_x(x) := \mathcal{U}(x, x)$ for $|x|^{1.5}$ steps
   - if $M_x(x)$ halts, output $1 - M_x(x)$
   - else, output 0

Deterministic Time Hierarchy

- We will prove: DTIME$(n) \subsetneq$ DTIME(n^2)

1. Let $\mathcal{U}$ be UTM from **[Hennie Stearns, 1966]**
2. Consider TM D: on input x
   - run $M_x(x) := \mathcal{U}(x, x)$ for $|x|^{1.5}$ steps
   - if $M_x(x)$ halts, output $1 - M_x(x)$
   - else, output 0
3. $D \in$ DTIME(n^2), since it always halts in $\leq |x|^{1.5}$ steps

Deterministic Time Hierarchy

- We will prove: $\mathsf{DTIME}(n) \subsetneq \mathsf{DTIME}(n^2)$

1. Let $\mathcal{U}$ be UTM from **[Hennie Stearns, 1966]**
2. Consider TM D: on input x
   - run $M_x(x) := \mathcal{U}(x, x)$ for $|x|^{1.5}$ steps
   - if $M_x(x)$ halts, output $1 - M_x(x)$
   - else, output 0
3. $D \in \mathsf{DTIME}(n^2)$, since it always halts in $\leq |x|^{1.5}$ steps
4. If $D \in \mathsf{DTIME}(n)$, let M be TM and $c \in \mathbb{N}$ constant s.t.
   - $M(x)$ halts in $\leq c \cdot |x|$ time
   - $M(x) = D(x)$ for all $x \in \{0, 1\}^*$

Deterministic Time Hierarchy

- We will prove: $\mathsf{DTIME}(n) \subsetneq \mathsf{DTIME}(n^2)$

1. Let $\mathcal{U}$ be UTM from **[Hennie Stearns, 1966]**
2. Consider TM D: on input x
   - run $M_x(x) := \mathcal{U}(x, x)$ for $|x|^{1.5}$ steps
   - if $M_x(x)$ halts, output $1 - M_x(x)$
   - else, output 0
3. $D \in \mathsf{DTIME}(n^2)$, since it always halts in $\leq |x|^{1.5}$ steps
4. If $D \in \mathsf{DTIME}(n)$, let M be TM and $c \in \mathbb{N}$ constant s.t.
   - $M(x)$ halts in $\leq c \cdot |x|$ time
   - $M(x) = D(x)$ for all $x \in \{0,1\}^*$
5. Let $y \in \{0,1\}^*$ such that $M_y = M$ and $|y|$ large enough

Deterministic Time Hierarchy

- We will prove: DTIME$(n) \subsetneq$ DTIME(n^2)

1. Let $\mathcal{U}$ be UTM from **[Hennie Stearns, 1966]**
2. Consider TM D: on input x
   - run $M_x(x) := \mathcal{U}(x, x)$ for $|x|^{1.5}$ steps
   - if $M_x(x)$ halts, output $1 - M_x(x)$
   - else, output 0
3. $D \in$ DTIME(n^2), since it always halts in $\leq |x|^{1.5}$ steps
4. If $D \in$ DTIME(n), let M be TM and $c \in \mathbb{N}$ constant s.t.
   - $M(x)$ halts in $\leq c \cdot |x|$ time
   - $M(x) = D(x)$ for all $x \in \{0, 1\}^*$
5. Let $y \in \{0, 1\}^*$ such that $M_y = M$ and $|y|$ large enough
6. Simulate $M(y)$ with UTM $\mathcal{U}(y, y)$
   - $D(y)$ obtains $M(y) = D(y)$ in $C \cdot |y| \log |y|$ time
   - in this case $D(y) = 1 - M(y)$ (by definition) contradiction

Non-deterministic Time Hierarchy

Theorem (**[Cook 1972]**)

If f, g are time-constructible functions such that $f(n+1) = o(g(n))$ for all $n \in \mathbb{N}$, then

$$\mathit{NTIME}(f(n)) \subsetneq \mathit{NTIME}(g(n))$$

Non-deterministic Time Hierarchy

We will prove $\mathsf{NTIME}(n) \subsetneq \mathsf{NTIME}(n^3)$.
Cannot use previous proof, since unclear "flip the output" of a NTM. Idea: lazy diagonalization.

1. Let $\mathcal{U}$ be UNTM

Non-deterministic Time Hierarchy

We will prove NTIME$(n) \subsetneq$ NTIME(n^3).
Cannot use previous proof, since unclear "flip the output" of a NTM. Idea: lazy diagonalization.

1. Let $\mathcal{U}$ be UNTM
2. Let $f : \mathbb{N} \to \mathbb{N}$ be such that $f(1) = 2$, $f(i+1) = 2^{f(i)^2}$.
   Given 1^n, not hard to find $i \in \mathbb{N}$ such that $f(i) < n \leq f(i+1)$ in $O(n^3)$ time.

Non-deterministic Time Hierarchy

We will prove $\mathsf{NTIME}(n) \subsetneq \mathsf{NTIME}(n^3)$.
Cannot use previous proof, since unclear "flip the output" of a NTM. Idea: lazy diagonalization.

1. Let $\mathcal{U}$ be UNTM
2. Let $f : \mathbb{N} \to \mathbb{N}$ be such that $f(1) = 2$, $f(i+1) = 2^{f(i)^2}$.
   Given 1^n, not hard to find $i \in \mathbb{N}$ such that $f(i) < n \leq f(i+1)$ in $O(n^3)$ time.
3. Consider NTM D: on input x, if $x \notin 1^*$, reject.
   - If $x = 1^n$, compute $i \in \mathbb{N}$ s.t. $f(i) < n \leq f(i+1)$
   - If $f(i) < n < f(i+1)$, simulate NTM $M_i(1^{n+1}) := \mathcal{U}(i, 1^{n+1})$ in $n^{1.5}$ time and output its answer (if M_i hasn't halted, then halt and accept)
   - If $n = f(i+1)$, accept 1^n iff $M_i(1^{f(i)+1}) = 0$ in $(f(i)+1)^{1.5}$ time

Non-deterministic Time Hierarchy

We will prove $\mathsf{NTIME}(n) \subsetneq \mathsf{NTIME}(n^3)$.
Cannot use previous proof, since unclear "flip the output" of a NTM. Idea: lazy diagonalization.

1. Let $\mathcal{U}$ be UNTM
2. Let $f : \mathbb{N} \to \mathbb{N}$ be such that $f(1) = 2$, $f(i+1) = 2^{f(i)^2}$.
   Given 1^n, not hard to find $i \in \mathbb{N}$ such that $f(i) < n \leq f(i+1)$ in $O(n^3)$ time.
3. Consider NTM D: on input x, if $x \notin 1^*$, reject.
   - If $x = 1^n$, compute $i \in \mathbb{N}$ s.t. $f(i) < n \leq f(i+1)$
   - If $f(i) < n < f(i+1)$, simulate NTM $M_i(1^{n+1}) := \mathcal{U}(i, 1^{n+1})$ in $n^{1.5}$ time and output its answer (if M_i hasn't halted, then halt and accept)
   - If $n = f(i+1)$, accept 1^n iff $M_i(1^{f(i)+1}) = 0$ in $(f(i)+1)^{1.5}$ time
4. $L_D \in \mathsf{NTIME}(n^3)$

Non-deterministic Time Hierarchy

We will prove NTIME$(n) \subsetneq$ NTIME(n^3).
Cannot use previous proof, since unclear "flip the output" of a NTM. Idea: lazy diagonalization.

1. Let $\mathcal{U}$ be UNTM
2. Let $f : \mathbb{N} \to \mathbb{N}$ be such that $f(1) = 2$, $f(i+1) = 2^{f(i)^2}$.
   Given 1^n, not hard to find $i \in \mathbb{N}$ such that $f(i) < n \leq f(i+1)$ in $O(n^3)$ time.
3. Consider NTM D: on input x, if $x \notin 1^*$, reject.
   - If $x = 1^n$, compute $i \in \mathbb{N}$ s.t. $f(i) < n \leq f(i+1)$
   - If $f(i) < n < f(i+1)$, simulate NTM $M_i(1^{n+1}) := \mathcal{U}(i, 1^{n+1})$ in $n^{1.5}$ time and output its answer (if M_i hasn't halted, then halt and accept)
   - If $n = f(i+1)$, accept 1^n iff $M_i(1^{f(i)+1}) = 0$ in $(f(i)+1)^{1.5}$ time
4. $L_D \in$ NTIME(n^3)
5. $L_D \in$ NTIME(n) then get similar contradiction to previous proof.

References I

Cook, Stephen (1972)
A hierarchy for nondeterministic time complexity
Proceedings of the fourth annual ACM symposium on Theory of computing

Hartmanis, J. and Stearns, R.E. (1965)
On the computational complexity of algorithms
Transactions of the American Mathematical Society

Hennie, Fred C and Stearns, Richard Edwin (1966)
Two-tape simulation of multitape Turing machines
Journal of the ACM (JACM)