CS310 : Automata Theory 2019

Lecture 35: Efficiency in computation

Instructor: S. Akshay

IITB, India

02-04-2019

Recap

Turing machines and computability

1. Turing machines
   (i) Definition
   (ii) Variants
   (iii) Decidable and Turing recognizable languages
   (iv) Church-Turing Hypothesis

Recap

Turing machines and computability

1. Turing machines
   (i) Definition
   (ii) Variants
   (iii) Decidable and Turing recognizable languages
   (iv) Church-Turing Hypothesis
2. Undecidability
   (i) A proof technique by diagonalization
   (ii) Via reductions
   (iii) Rice's theorem

Recap

Turing machines and computability

1. Turing machines
   (i) Definition
   (ii) Variants
   (iii) Decidable and Turing recognizable languages
   (iv) Church-Turing Hypothesis
2. Undecidability
   (i) A proof technique by diagonalization
   (ii) Via reductions
   (iii) Rice's theorem
3. Applications: showing (un)decidability of other problems
   (i) A string matching problem: Post's Correspondance Problem
   (ii) A problem for compilers: Unambiguity of Context-free languages
   (iii) Between TM and PDA: Linear Bounded Automata

Animals between PDA and TM

DFA/NFA $<$ PDA $<$ LBA $<$ Algorithms/Halting TM $<$ Semi-algorithms/TM

Animals between PDA and TM

DFA/NFA $<$ PDA $<$ LBA $<$ Algorithms/Halting TM $<$ Semi-algorithms/TM

Regular $\subsetneq$ CFL $\subsetneq$? $\subsetneq$ Decidable $\subsetneq$ Recursively Enumerable $\subsetneq$ All languages

Animals between PDA and TM

DFA/NFA $<$ PDA $<$ LBA $<$ Algorithms/Halting TM $<$ Semi-algorithms/TM

Regular $\subsetneq$ CFL $\subsetneq$? $\subsetneq$ Decidable $\subsetneq$ Recursively Enumerable $\subsetneq$ All languages

Homework problem: Show PDA $<$ LBA

Animals between PDA and TM

DFA/NFA $<$ PDA $<$ LBA $<$ Algorithms/Halting TM $<$ Semi-algorithms/TM

Regular $\subsetneq$ CFL $\subsetneq$? $\subsetneq$ Decidable $\subsetneq$ Recursively Enumerable $\subsetneq$ All languages

Homework problem: Show PDA $<$ LBA

Challenging questions/Find out!

- Why is LBA $<$ Halting TM?
- What is a notion of languages/grammar for LBA?
- Are non-deterministic LBA more powerful than deterministic LBA?

Turing machines with Resource constraints

LBA are an example of resource bounded TMs

Can you think of other resources?

Turing machines with Resource constraints

LBA are an example of resource bounded TMs

Can you think of other resources?

Resources for computation:

- Space/memory
- Time

Turing machines with Resource constraints

LBA are an example of resource bounded TMs

Can you think of other resources?

Resources for computation:

- Space/memory
- Time
- Cost
- Energy
- What else?

Turing machines with Resource constraints

LBA are an example of resource bounded TMs

Can you think of other resources?

Resources for computation:

- Space/memory
- Time
- Cost
- Energy
- What else?number of times a state is visited, number of tape reversals, number of writes

Turing machines with Resource constraints

LBA are an example of resource bounded TMs

Can you think of other resources?

Resources for computation:

- Space/memory
- Time
- Cost
- Energy
- What else?number of times a state is visited, number of tape reversals, number of writes

Why consider resources?

- TMs are algorithms...
- Decidability does not implementability!

Running Time Complexity

Given M a halting TM, running time of M is the function $f(n) : \mathbb{N} \to \mathbb{N}$, which counts the maximum number of steps that M uses on any input of length n.

Running Time Complexity

Given M a halting TM, running time of M is the function $f(n) : \mathbb{N} \to \mathbb{N}$, which counts the maximum number of steps that M uses on any input of length n.

Is this the only notion possible? Any others?

Running Time Complexity

Given M a halting TM, running time of M is the function $f(n) : \mathbb{N} \to \mathbb{N}$, which counts the maximum number of steps that M uses on any input of length n.

- Worst-case complexity - longest running time of all inputs of length n (in this course, we consider this)
- Average-case complexity - average running time over all inputs of length n.

A recap of Big O and small o notation

Let $f, g : \mathbb{B} \to \mathbb{R}^+$, we say $g(n)$ is an (asymptotic) upper bound for $f(n)$, denoted

$f(n) = O(g(n))$ if $\exists c, n_0 \in \mathbb{Z}^+$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$.

- f is less than or equal to g up to a constant factor.
- why use this? to estimate instead of precise.

A recap of Big O and small o notation

Let $f, g : \mathbb{B} \to \mathbb{R}^+$, we say $g(n)$ is an (asymptotic) upper bound for $f(n)$, denoted
$f(n) = O(g(n))$ if $\exists c, n_0 \in \mathbb{Z}^+$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$.

Let $f, g : \mathbb{B} \to \mathbb{R}^+$, we write $f(n) = o(g(n))$ if $\forall c > 0, \exists n_0$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$, i.e., $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

- more like strictly less than (asymptotically)

A recap of Big O and small o notation

Let $f, g : \mathbb{B} \rightarrow \mathbb{R}^+$, we say $g(n)$ is an (asymptotic) upper bound for $f(n)$, denoted

$f(n) = O(g(n))$ if $\exists c, n_0 \in \mathbb{Z}^+$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$.

Let $f, g : \mathbb{B} \rightarrow \mathbb{R}^+$, we write $f(n) = o(g(n))$ if $\forall c > 0, \exists n_0$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$, i.e., $\lim_{n\rightarrow\infty} \frac{f(n)}{g(n)} = 0$.

Exercises: True or false

1. $234n^3 + 345n^2 - 3 = O(n^3)$
2. $84n^3 + 4n^4 + 3 = O(n^5)$
3. $n = o(nloglogn)$
4. $2^{O(n)} = o(n^{242345325})$
5. $O(n) + \frac{n}{2}O(n) + O(n) = O(n^2)$

A recap of Big O and small o notation

Let $f, g : \mathbb{B} \to \mathbb{R}^+$, we say $g(n)$ is an (asymptotic) upper bound for $f(n)$, denoted

$f(n) = O(g(n))$ if $\exists c, n_0 \in \mathbb{Z}^+$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$.

Let $f, g : \mathbb{B} \to \mathbb{R}^+$, we write $f(n) = o(g(n))$ if $\forall c > 0, \exists n_0$ s.t $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$, i.e., $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

Exercises: True or false

1. $234n^3 + 345n^2 - 3 = O(n^3)$
2. $84n^3 + 4n^4 + 3 = O(n^5)$
3. $n = o(nloglogn)$
4. $2^{O(n)} = o(n^{242345325})$
5. $O(n) + \frac{n}{2}O(n) + O(n) = O(n^2)$

polynomial: n^c for $c > 0$, exponential: 2^{n^δ}, $\delta > 0$.

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

Running Time Complexity

Let $t : \mathbb{N} \rightarrow \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Running Time Complexity

Let $t : \mathbb{N} \rightarrow \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject.
2. repeat until there are no 0's:
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject)
4. at end if there are 1's remaining reject, otherwise, accept.

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject. $O(n)$ steps
2. repeat until there are no 0's:
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject)
4. at end if there are 1's remaining reject, otherwise, accept.

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject. $O(n)$ steps
2. repeat until there are no 0's:
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject) $O(n)$ steps
4. at end if there are 1's remaining reject, otherwise, accept.

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject. $O(n)$ steps
2. repeat until there are no 0's: $n/2$ repetitions
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject) $O(n)$ steps
4. at end if there are 1's remaining reject, otherwise, accept.

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject. $O(n)$ steps
2. repeat until there are no 0's: $n/2$ repetitions
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject) $O(n)$ steps
4. at end if there are 1's remaining reject, otherwise, accept. $O(n)$ steps

Running Time Complexity

Let $t : \mathbb{N} \to \mathbb{R}^+$. A language $L \subseteq \Sigma^*$ is said to be in $TIME(t(n))$ if there exists a deterministic (halting) Turing machine M such that $\forall x \in \Sigma^*$ of length n, M halts on x within time $O(t(n))$.

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ TM

Exercise: Is $A = \{0^k 1^k \mid k \geq 0\}$ in $O(n^2)$?

1. scan tape once to check if input is of form 0^*1^*. else reject. $O(n)$ steps
2. repeat until there are no 0's: $n/2$ repetitions
3. scan tape to cross a single 0 and single 1 (if you cant find 1 to cross off, reject) $O(n)$ steps
4. at end if there are 1's remaining reject, otherwise, accept. $O(n)$ steps

Overall: $O(n) + \frac{n}{2}O(n) + O(n) = O(n^2)$ steps

Running Time Complexity (Contd.)

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ Turing machine

Questions

- Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- Is A in $O(n)$?

Running Time Complexity (Contd.)

TIME$(t(n))$ is set of all languages decidable by a $O(t(n))$ Turing machine

Questions

- Is $A = \{0^k1^k \mid k \geq 0\}$ in $o(n^2)$?
- Is A in $O(n)$?

Does crossing two 0s and 1s on every scan instead of just one help?

Running Time Complexity (Contd.)

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ Turing machine

Questions

- Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- Is A in $O(n)$?

- Scan tape and reject, if 0 is to right of 1.
- Scan the 0s and copy them to another tape, until first 1.
- Scan 1s in first tape together with 0s in second tape, if 0's are crossed before 1s are read, reject
- if all 0s are crossed off at the end, accept, else reject.

Running Time Complexity (Contd.)

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ Turing machine

Questions

- Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- Is A in $O(n)$?

- Scan tape and reject, if 0 is to right of 1.
- Scan the 0s and copy them to another tape, until first 1.
- Scan 1s in first tape together with 0s in second tape, if 0's are crossed before 1s are read, reject
- if all 0s are crossed off at the end, accept, else reject.

So, with 2-tape can improve "complexity".

Running Time Complexity (Contd.)

TIME$(t(n))$ is set of all languages decidable by a $O(t(n))$ 1-tape Turing machine

Questions

- (HW) Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- Is A in $O(n)$?

- Scan tape and reject, if 0 is to right of 1.
- Scan the 0s and copy them to another tape, until first 1.
- Scan 1s in first tape together with 0s in second tape, if 0's are crossed before 1s are read, reject
- if all 0s are crossed off at the end, accept, else reject.

So, with 2-tape can improve "complexity". Can we improve with 1-tape?

Running Time Complexity (Contd.)

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ 1-tape Turing machine

Questions

- (HW) Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- (Challenge) Is A in $O(n)$?

So, with 2-tape can improve "complexity".

Running Time Complexity (Contd.)

$TIME(t(n))$ is set of all languages decidable by a $O(t(n))$ 1-tape Turing machine

Questions

- (HW) Is $A = \{0^k 1^k \mid k \geq 0\}$ in $o(n^2)$?
- (Challenge) Is A in $O(n)$?

So, with 2-tape can improve "complexity".

Conclusions: change of model can change complexity, even if it does not change computability.