Michael A. Forbes
M.forbes@illinois.edu
2024-08-29.1
CS579

CS579 Computational Complexity : Lecture 2 (2024-08-29)

logistics : pset 1 out, due 09-12
- logistics
last lecture : - motivation and goals
- background

today : time complexity - def
- examples
- robustness

[[move to front]]

[[recall]]

def : a Turing Machine (TM) is M [[def by picture]]
- read tape at head position
each step : - write
- update finite control
- move tape head

[[finite control]] [[head]]
start, acc, rej
[[input]] [[blanks]]

vs 2023 : def by pair.

M accepts input x if q_{acc} reached before q_{rej}, else rejects [[reach q_{rej} & not halt]]

$L(M) = \{x : M \text{ acc } x\}$ [[language]]

[[power?]]

prop : exist TM M s.t. $L(M) = \{0^n 1^n : n \in \mathbb{N}\}$, and M always halts

vs 2023 : (2) n vs n

pf : [[Q]]

idea : 0 1 0 ⊔⊔ ... $\notin 0^+1^+$ [[DFA, last time]] ⇒ reject
[[cross off]]
0 0 X X 1 ⊔⊔ ... ⇒ reject
0 0 0 1 1 X ⊔⊔ ... ⇒ accept

vs 2023 :

[[3 components]] [[algo]]
algo : [[TM]] M = "on input x:
[[CT : write TM in words]]
[[start at left, end at blank]]
- pass over x, check if $x \in 0^+1^+$, reject if not [[halts]] O(n)
- repeat ≤ n loops
- pass over input
- cross off single 0 ↩ 0 ↦ ∅ O(n)
- cross off single 1 ↩ 1 ↦ X
- reject if 0's finish before 1's
- 1's finish before 0's
- accept

vs 2023 :

complexity : [[complexity]] [[halts]] DFA, so halts

correctness : [[correctness]] [[$L(M)$]] induction ⇒ halts [[finite # symbols]]
each loop crosses off ≥ 1 symbol [[uses read more details]]
clear [[# steps]]

Q : efficiency?

prop : [[same TM]] , halts in $O(n^2)$ steps [[$n = |x|$]]

sketch : ≤ $O(n) + n \cdot O(n) = O(n^2)$ steps [[asymptotically]]

Q : can we do better?

A : intuitively need ≥ n steps ↩ "MUST read all of input"

fact [Sipser §7.47] : cannot achieve $O(n)$ steps

[[Q : n^2 vs n]]

ex

Michael A. Forbes
miforbes@illinois.edu
2024-08-29.2
cs579

vs2023 - n vsm

prop: exists TM M w/ $-L(M) = \{0^n 1^n : n \in \mathbb{N}\}$
- halts in $O(n \lg n)$ steps on length n inputs

pf: idea.
- check for $0^* 1^*$
- given if $0^a 1^b$, check if $a = b$ ($\lg n$ bits)
  - ↳ check one bit at a time
    - ↳ takes $O(n)$ time

vs 2023

algo: M = " on input x:
- $O(n)$ — check if $x \in 0^* 1^*$, reject if not
- ($\leq \lg n$) — repeat
  - $O(n)$ — count parity of # of 0's [not crossed off]
  - — " — 1's — reject if unequal
  - $O(n)$ — cross off every other 0, eg $\not{0}\,0$, $\not{0}\,0\,\not{0}$
  - — " — 1
- accept"

vs 2023

correctness [$L(M)$]: given $0^a 1^b$ [wlog $x \in 0^* 1^*$] w/ [division w/ remainder]
$a = 2a' + a''$
$b = 2b' + b''$, $a'', b'' \in \{0,1\}$ — unique decomposition

$a' = \lfloor a/2 \rfloor$ [parity]
$a'' = a \bmod 2 \in \{0,1\}$
$b' = \lfloor b/2 \rfloor$
$b'' = b \bmod 2 \in \{0,1\}$

demo rules up

if $a'' \neq b''$: reject [uniqueness] $a \neq b$ — correct!

$a'' = b''$: [ignore crossed off] produces $0^{\lfloor a/2 \rfloor} 1^{\lfloor b/2 \rfloor} = 0^{a'} 1^{b'}$
- if $a = b$: $\Rightarrow a' = b' \Rightarrow$ accept [recursively]
- $a \neq b$: $\Rightarrow a' \neq b' \Rightarrow$ reject [recursively]

correct! [correct!]

base case [recursion]: empty string is accepted

vs 2023

complexity [# steps]: [accounting]
$$\#\text{rounds}: R(n) \leq 1 + R(\lfloor n/2 \rfloor) \leq \text{[hw]} \in O(\lg n)$$
$$\Rightarrow O(n \lg n) \text{ steps.}$$
∎

= [Q]

Q: do better?

ex

fact [Sipser 7.47]: no $o(n \lg n)$ algo [asymptotic optimality] [amazing!] [test-taking skills]

vs 2023

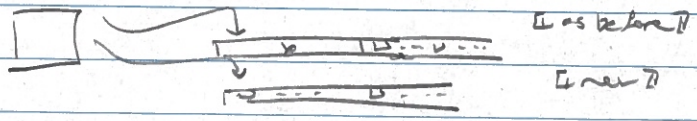
Q: are we done?

n vsm

prop: exists [2-tape] TM M w/ $-L(M) = \{0^n 1^n : n \in \mathbb{N}\}$
- runs in $O(n)$ steps [!]

Michael A. Forbes
MForbes@cs.illinois.edu
2024-08-29.3
CS579

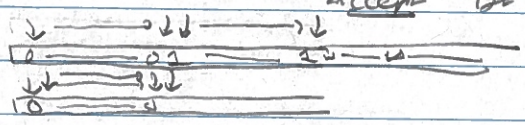
↳ M



[[both]] heads get updated in each step, independently [[state, read/write, movement]]

pf: algo $M =$ " on input x:
- check if $x \in 0^* 1^*$, reject if not
- copy all 0's on first tape onto second tape
- accept iff # 1's on first tape = # 0's on second tape"

[VS 2023]

eg 

correctness: clear

complexity: $O(n) + O(n) = O(n)$ ∎

rmk: - time complexity changed w/ model
- ↳ but not by much

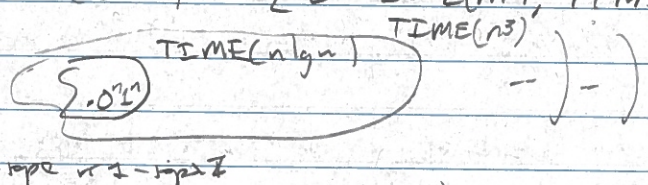
want: - theory of computation independent of specific model
- specific model of study [[have to choose]] = one tape TM
⇝ universality of specific model

def: TM M runs in time $t(n): \mathbb{N} \to \mathbb{N}$ if for all inputs $x \in \Sigma^n$ [[input length n]] M halts in $\le t(n)$ steps

[VS 2023]

$$\mathrm{TIME}(t(n)) = \{ L : L = L(M),\ M \text{ runs in time } O(t(n)) \}$$

[[ignoring]] ↳ ignore constant factor

eg: 

prop: $\mathrm{TIME}_{2\text{-tape TM}}(t(n)) \subseteq \mathrm{TIME}\left(t(n)^2 + n^2\right)$ [[2-tape vs 1-tape]] [[one tape]] [[a bound gap in power]]

pf: idea: simulate each step of 2-tape TM by $O(t(n))$ steps of 1-tape TM
$\le t(n)$ steps

$O(n^2)$ initialization $\Rightarrow O(t^2 + n^2) \subseteq O(t^2)$

↳ encode 2-tape into 1-tape via interleaving [[↳ $t(n) \ge n$ to read input]]

$a_1 a_2 a_3 \cdots$ ⟼ $a_1 b_1 a_2 b_2 a_3 b_3 \cdots$
$b_1 b_2 b_3 \cdots$

[VS 2023]

- encode head positions into tape

[VS 2023]

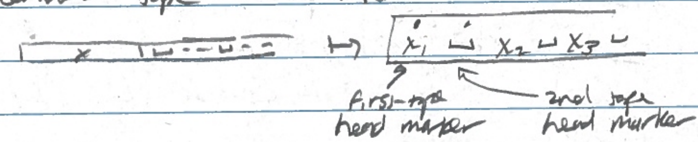
2-tape TM w/ tape alphabet Γ
⟼ 1-tape TM w/ tape alphabet $\Gamma \cup \{ \dot{\gamma} : \gamma \in \Gamma \}$
↑ marker for head position

Michael A. Forbes
mforbes@illinois.edu
2024-08-29.4
CS579

M 2-tape TM

claim: M' = "on input x:

$O(n^2)$ [insert takes n steps] [& insert n blanks] — convert tape to interleaved form



t steps to simulate, $O(t)$ tape cells $\Rightarrow O(t)$ steps — simulate each transition of M
- scan tape to find head marker of first tape, read location
- " second
- " first, write and move
- " second, "
- update finite control of M "

• vs 2021

correctness: "clear"

complexity: [accounting] $\leq O(n^2) + t \cdot O(t) \leq O(t^2+n^2)$ ∎

• vs 2023

rmk: - 2 tape TM ≈ 1-tape TM, up to polynomial factors in runtime
- ↳ true of all known deterministic (realistic) models of computation [hence 1-tape is good]

Extended Church-Turing thesis: ↳ true for all realistic models of computation

rmk: ↳ may be false due to quantum computing

def: $P = \bigcup_k \text{TIME}(n^k) = \text{TIME}(\text{poly}(n))$ is polynomial time

rmk: - P is model invariant
- roughly corresponds to "practical solvability" - n^{1000} not practical - $2^{n/1000}$ can be practical
- closed under subroutines, $\text{poly}(\text{poly}(n)) = \text{poly}(n)$

Q: what is P?

• check

Today: time complexity - def - examples - robustness

next lecture: nondeterminism

logistics: pset 1 due 09-12