erase !
$11 \to ?$
loop until no symbols
t is $O(t)$
worst case

Michael A. Forbes
miforbes@illinois.edu
2023-08-24-1

CS579 Computational complexity: Lecture 2 (2023-08-24)

logistics: - pset 1 out, due 09-07

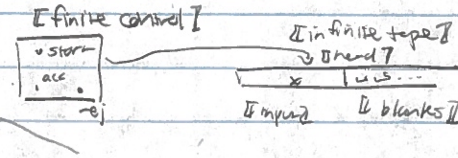
last lecture: - logistics
- motivation and goals
- background

today: time complexity
- def
- examples
- robustness

[move to front]

[recall]

def: Turing machine (TM) is [picture] M



eachstep: - read tape at head position
- write tape at head position
- update finite control
- move tape head

moving off tape • 2021 •

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

[language]

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

[prove?]

n vs m •

prop: exist TM M w/ $L(M) = \{0^m 1^m : m \in \mathbb{N}\}$, and M always halts

pf: [Q] $0 1 0 \sqcup \sqcup \ldots \notin 0^* 1^* \Rightarrow$ reject [DFA, last time]

change last example? •

idea: $\cancel{0}\cancel{0}\cancel{1}11 \sqcup \sqcup \ldots \Rightarrow$ reject

$\cancel{0}\cancel{0}\cancel{0}\cancel{1}\cancel{1}\cancel{1} \sqcup \sqcup \ldots \Rightarrow$ accept

[3 components]

algo: [algo] M = "on input x:
[TM]
[CT: write TM in words]

- pass over x, check if $x \in 0^* 1^*$, reject if not [start at left, end at blank] [DFA] [and halt] — $O(n)$
- repeat — n loops
  - pass over input
  - cross off single 0 — $0 \mapsto \cancel{0}$
  - cross off single 1 — $1 \mapsto \cancel{1} \in \Gamma$ — $O(n)$
- reject if - 0's finish before 1's
  - 1's finish before 0's
- accept

complexity: [complexity] [halts] — DFA, so halts
each loop crosses off ≥ 1 symbol $\Rightarrow$ halt [induction] [finite # symbols]

correctness: [correctness] [L(M)] "clear" [need more details in pset]

Q: efficiency? [# steps]

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

sketch: $\leq O(n) + n \cdot O(n) = O(n^2)$ steps

Q: can we do better? [asymptotically]

A: intuitively need $\geq n$ steps ← "must read all of input" •

fact [Sipser #7.47] cannot achieve $O(n)$ steps •

[Q: n^2 vs n]

Michael A. Forbes
mforbes@illinois.edu

2023-08-24.2

d.n w/ counters

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

pf. idea:
- check for $0^* 1^*$
- given $0^a 1^b$, check if $a = b$

• $\lg n$ vs length(n)

$\lg n$ bits
check one bit at a time
takes $O(n)$ time

algo: M = [TM] " on input x:
- $O(n)$ — check if $x \in 0^* 1^*$, reject if not
- 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 $\cancel{0}0$, $\cancel{0}0\cancel{0}$
  - — " — 1
- accept "

correctness [$\mathcal{L}(M)$]: given $0^a 1^b$ w/ $a = 2a' + a''$ [division w/ remainder], $a' = \lfloor a/2 \rfloor$, $a'' = a \bmod 2 \in \{0,1\}$ [parity]

$b = 2b' + b''$, $b' = \lfloor b/2 \rfloor$, $b'' = b \bmod 2 \in \{0,1\}$ [unique decomposition]

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

[recursion] base case: empty string is accepted [correct!]

complexity [# steps]:

$O(n)$ work for DFA

$O(n)$ work per round

rounds $R(n) \le 1 + R(\lfloor n/2 \rfloor) \le \cdots \le O(\lg n)$ [hw]

$\Rightarrow O(n \lg n)$ steps ☐

Q: do better?

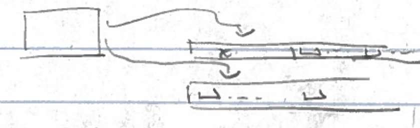
vs NP^cc 16?

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

Q: are we done?

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

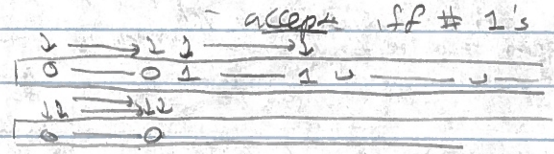
Michael A. Forbes
miforbes@illinois.edu
2023-08-24.3

↳ M  [as before] [new]

$\boxed{\text{both}}$ 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"



correctness: clear

complexity: $O(n) + O(n) = O(n)$ ∎

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

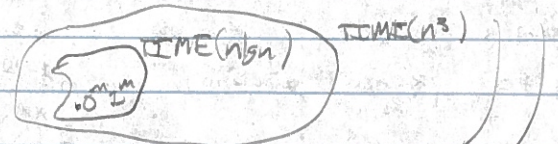
want: - theory of computation $\boxed{\text{independent}}$ of specific model
- specific model to study [to have to choose] = one tape TMs
- ⟹ $\boxed{\text{universality}}$ of specific model

t vs $O(t)$.

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

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

↳ ignore constant factors [granularity]

eg: 

[2 tape vs 1 tape]

prop: $\mathrm{TIME}_{2\text{-tape TM}}(t(n)) \subseteq \mathrm{TIME}(t(n)^2 + n^2)$ [one tape] [bounds gap in power]

pf. idea: simulate $\underbrace{\text{each step}}_{\le t(n) \text{ steps}}$ of 2-tape TM by $O(t(n))$ steps of 1-tape TM

$O(n^2)$ initialization ⟹ $O(t^2 + n^2) = O(t^2)$ ← $t(n) \ge n$ often [read input]

↳ - encode 2 tapes into 1 tape via interleaving

$a_1\ a_2\ a_3 \ldots$ / $b_1\ b_2\ b_3 \ldots$ ⟼ $a_1 b_1\ a_2 b_2\ a_3 b_3 \ldots$

- encode head position into tape

2-tape TM w/ tape alphabet Γ

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

M 2-tape TM

algo M' = " on input x:

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

x | ⊔ ⊔ --- ⊔ --- $\mapsto$ $\dot{x}_1$ $\dot{⊔}$ x_2 ⊔ x_3 ⊔ ---

first tape head marker — 2nd tape head marker

t steps, $O(t)$ 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 "

• self

• conv self

• details vs steps

correctness: "clear"

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

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 $P = \bigcup_k \mathrm{TIME}(n^k) = \mathrm{TIME}(\mathrm{poly}(n))$ 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, $\mathrm{poly}(\mathrm{poly}(n)) = \mathrm{poly}(n)$

Q: what is P?

today: time complexity - def
- examples
- robustness

next lecture: nondeterminism

logistics: pset 1 due 09-07