18
tikz collab
erase
1109

Michael A. Forbes
miforbes@illinois.edu
2023-08-21.1

cs579 Computational Complexity: Lecture 1 (2023-08-22)

today: - logistics
- motivation and goals
- background

[[move to front]]

class: cs579 Computational Complexity
TR 15:30 - 16:45, Siebel 1302
courses.engr.illinois.edu/cs579/fa2023 [[optional]]

instructor: Prof. Michael A. Forbes
miforbes@illinois.edu
[[office hrs]] Siebel 3220 T17, [or] by appt [[we can make it at]]

TA: Zander Kelley
awk2@illinois.edu
TBD

grades: - psets (70%): - 6 - biweekly - 1 ← starts Thurs [[lighter in 2nd half to accomodate project]]
- project (30%) - groups of size 2+ε [[ε = rounding]] [[late policy online]]
- paper → you - 30 min presentation [[end of semester]]
- short report

ref: - "Introduction to the Theory of Computation", Sipser, [2nd] [3rd] edition
- first half of course
- "Computational Complexity - A modern Approach", Arora, Barak
- second half of course
[[scanned]]
- course notes [[~illegible]]
- course videos [[previous year ≈ this year]]
↳ @illinois.edu only

preq: - discrete math 173
- models of computation 374, 475 [[will overlap]]
- algorithms 473
- mathematical maturity [[(!)]]

[[Q: in-person!]]

Q: why are we here?

A: cryptography [[encryption is everywhere]]

Alice ⟷ Bob

Eve

secret key k secret key k [[pre-shared]]

message M → Enc(m,k) →

Michael A. Forbes
mikeforbes@illinois.edu
2023-08-21.2

want: decryption: $Dec(Enc(m,k),k) = m$ [[efficient algo]]

encryption: $Crack(Enc(m,k))$ "reveals nothing" [[modeling]]
about m if Crack is efficient algo
[[lack of efficient algo]]

$\Rightarrow$ crypto requires - easy problems
- hard problems
} related

Q: are there hard problems? [[all problems easy?]]
which problems are hard? [[classify]]
why are problems hard? [[understanding]]
} this course

A: are hard questions [[start somewhere]]

Q: what is "convincing evidence" a problem is hard?

[[Q]]

someone on the internet [[ChatGPT]] said it was hard
we thought for a long time and found no efficient algo
no algo of specific/natural form can efficiently solve the problem
"similar" problems can be proven to be hard
unconditional mathematical proof of hardness
relate to other "seemingly hard" problems
} also this course [[non-trivial!]]

goals: identify important computational problems
- shortest paths in graphs
- primality testing
- satisfiability of boolean formula

identify important computational resources
- time
- space
- ability to solve a specific computational problem

[[leads to Q]]

Q: does using more of a resource give more computational power? [[caffeine to thms]]
how do different types of resources compare? [[sleep is caffeine, to thms]]
problems vs resources? [[thms into caffeine]]

this course - [[22/28]] - structural complexity (3/4)
- theory of Turing machines, different resources
- few unconditional results
- concrete complexity (~1/4)

Michael A. Forbes
miforbes.illinois.edu
2023-08-21.3

- theory of finite computational models, eg circuits
- "more" unconditional results

← [[QE]]
[[background]]

def: a language is a set $L \subseteq \Sigma^*$ ← all finite strings over Σ
← alphabet, often $\{0,1\}$

Q: given $x \in \Sigma^*$, is $x \in L$? [[prototypical question, often decision problem. Captures variants]]

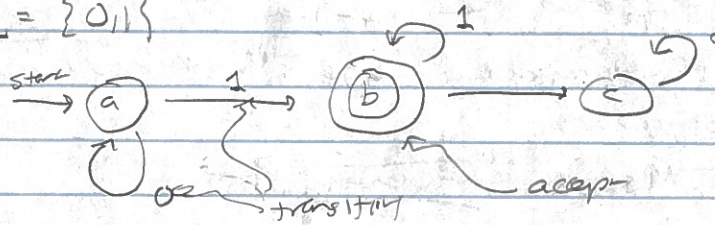
[[model]]
def: a deterministic finite automaton (DFA) is a 5-tuple
$M = (Q, \Sigma, \delta, q_0, F)$ where:
- Q is a finite set of states
- Σ alphabet
- $\delta: Q \times \Sigma \to Q$ transition function
- $q_0 \in Q$ start state [[single]]
- $F \subseteq Q$ accept states [[multiple]]

A DFA accepts $x \in \Sigma^*$, if - start on q_0
- at ith step, transition $q \mapsto q' = \delta(q, x_i)$ [[start on x_1]] 2019, 2021
- after n-th step - accept if $q \in F$
- reject else

$n = |x|$ [[universal]] [[ith symbol / after i steps]]

denote $L(M) = \{x : M \text{ accepts } x\}$ ← language of M ← regular language [[some M]]

eg $\Sigma = \{0,1\}$



2019 Σ vs Σ^*

$L(M) = \{0^* 1 1^*\}$ is regular
[[includes empty word]]
[Eg 3.7]

fact: $\{0^n 1^n : n \in \mathbb{N}\}$ is not regular
[[amazing!]] [[DFA's well understood]] [[need stronger models]]

Michael A. Forbes
miforbes@illinois.edu
2023-08-21.4

def: a Turing machine (TM) is a [DFA w/ a tape to store all]

[finite control] [infinite tape]

formally - Q set of states, $\{q_{start}, q_{acc}, q_{rej}\} \subseteq Q$

Γ tape alphabet, $\Gamma \supseteq \Sigma$

• σ vs δ

δ transition function

$$\delta(\text{current symbol under tape head}, \text{current state}) = (\text{head movement}, \text{tape symbol to write}, \text{new state})$$

TM computes by - tape initialized to $x \sqcup \sqcup \cdots$ ($\sqcup \in \Gamma \setminus \Sigma$ [blank])

- head placed at start of tape
- [iterate δ until reach q_{acc} or q_{rej}] halts

the language L of TM M is: M on x reaches $q_{acc} \Rightarrow x \in L$

——//—— $q_{rej} \Rightarrow x \notin L$

[never] halt $\Rightarrow x \notin L$

fact: - exists TM M w/ $L(M) = \{0^n 1^n : n \in \mathbb{N}\}$ [! TM $\not\to$ DFA]

- any language of any [known] computational process is also the language of some TM [! TM $\geq$ CPUs] → [possible]

Church-Turing Thesis: [philosophical point]

fact [cs374] $L = \{\langle M, x \rangle$ (← encoding) : M does not halt on $x\}$

M ~ TM, x ~ input

is [not] the language of any TM that [always] halts

L is undecidable

Q: What can TMs do [efficiently]?

today: - log visits
- motivation and goals
- background

next lecture: time complexity