unmatched [

Michael A. Forbes
miforbes@illinois.edu
2023-09-05.1

CS579 Computational Complexity: Lecture 5 (2023-09-05)

logistics: pset 1 due 09-07 12pm [gradescope]

last lecture: reductions [elementary properties] [A ≤ B, use B to solve A]
NP-completeness [def via reductions] [hardest prob in NP]
SAT, 3SAT [def] [Cook-Levin: 3SAT NP-complete]
3SAT ≤ CLIQUE [⟹ CLIQUE NP-complete]

3SAT vs CNF-SAT vs 2SAT in two

today: Cook-Levin theorem

thm [Cook 71, Levin 73] SAT = {⟨φ⟩ : φ boolean formula [AND, OR, NOT] $\exists \alpha \in \{0,1\}^n$ [# vars], $\varphi(\alpha)=1$} [satisfiable]

3SAT = {⟨φ⟩ : φ 3CNF [CNF] satisfiable}

$\varphi = C_1 \wedge \cdots \wedge C_m$ [clause] $C = l_1 \vee l_2 \vee l_3$ [literal] ← x or $\bar{x}$

3SAT is NP-complete [∈ NP]

pf idea: (a) A ∈ NP $A \le_p$ CNF-SAT [unbounded # literals per clause], via computation [tableau] [main work]

(b) CNF-SAT $\le_p$ 3SAT [direct reduction] [[now compose]] reread space

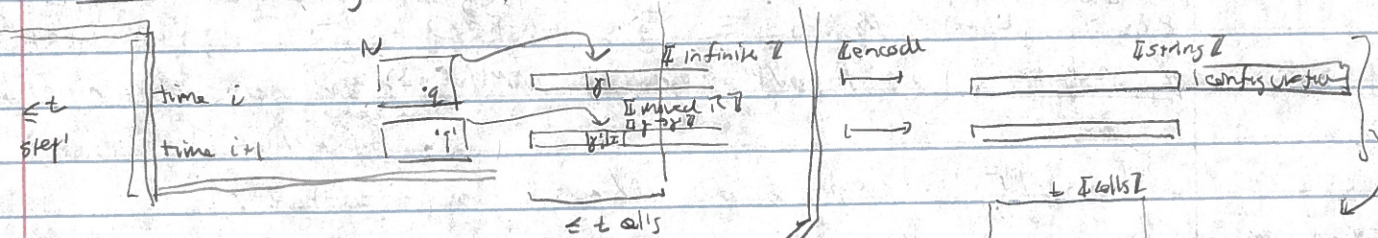
A decided by NTM N time $t(n) \le poly(n)$

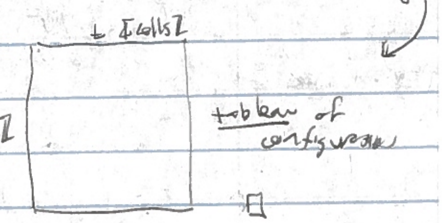
design $f : \Sigma^* \to \Sigma^*$ polytime computable
$x \mapsto \langle \varphi_x \rangle$ [encoding] CNF

$x \in A$ iff N acc x
iff $\langle \varphi_x \rangle \in$ CNF-SAT

↳ φ_x simulates N on x
- variables: encode nondeterministic guesses
- clauses: guesses lead to N accepting [constraints]

want: encoding of computation into boolean logic





N acc x iff exists tableau [details]
- first config is initial config
- i-th config → (i+1)st config is valid [nondet]
- last config is accepting [reach qacc]

def: for NTM N w/ states Q, tape alphabet Γ [disjoint], a configuration is a string over alphabet $C := Q \sqcup \Gamma \sqcup \{\#\}$, where: [marker]

- '#' is the first and last symbol, appearing nowhere else
- exactly one symbol in Q appears, located before a Γ symbol

encodes - state of NTM
- position of head in tape, reading

Michael A. Forbes
Miforbes@illinois.edu
2023-09-05.2

it uses t cells if t symbols of Γ appear. An accepting config has $q_{acc} \in Q$

eg: NTM N on x has initial configs using t cells ($\# q_{start} x_1 \dots x_n \sqcup \dots \sqcup \#$) [$t-n$ blanks]

rejecting — " — q_{rej} — " ; halting config

[clear]
lem - snapshot of NTM computation using $\leq t$ cells can be bijectively encoded [exer] into a configuration with $t+3$ symbols
- t tape cells [rest blank]
- 1 head
- 3 markers

def - t-step tableau of N on x is a sequence of t configurations using t cells
where: - configuration 1 is t-cell init config of N on x
- config i - encodes tape using $\leq t$ cells [no overflow]
- non-deterministically follows from computation of configuration $i-1$ [Q] [or] equals $(i-1)$st config if computation has halted

$t+3$ symbols [$Q \cup \Gamma \cup \{\#\}$] $\Rightarrow t \times (t+3)$ matrix

tableau is accepting if t-th config is accepting [has q_{acc}]
— " — rejecting — " — rejecting

lem: NTM N accepts x on $\leq t$ steps iff $\exists$ accepting t step tableau
[by def]
pf $\Rightarrow$: time $\leq t \Rightarrow \begin{cases} \leq t \text{ configs (row)} \\ \leq t \text{ cells per config (col)} \end{cases} \equiv$ tableau

[some] branch accepts $\Rightarrow$ [some] tableau is accepting

$\Leftarrow$: $t \times (t+3)$ size accepting tableau $\Rightarrow$ initialize NTM
repeatedly transition validly [no overflow]
accepted & halt in $\leq t$ step ; non-det guess
$\Rightarrow N$ acc x in $\leq t$ steps ☐

[QZ]
Q: what is $t(n)$? [poly time] [want [simple] form, for concreteness]

[technical, easy] [leading term] [hidden constants] [explicit ubd]
lem: $t(n) \in \mathrm{poly}(n) \equiv \exists c \; t(n) \leq O(n^c) \equiv \exists d, \forall n \; t(n) \leq n^d + d$

pf (Cook-Levin): $A \in NP$ accepted by NTM N - runs in time $t(n) = n^c + c$ [lem]
- states Q
- tape alphabet Γ

idea: NTM N accepts x iff N acc x in $\leq t(n)$ steps
iff — " — , [some] branch
[how?] iff [exists] t-step accepting tableau ← $t \times (t+3)$ size matrix
iff φ_x has satisfying assignment in vars (encodes)
(we construct) (encodes)

most hard line

[vars]
construction - φ has vars $y_{i,j,a}$ over $\{0,1\}$ [boolean]
$1 \leq i \leq t$
$1 \leq j \leq t+3$
$a \in Q \cup \Gamma \cup \{\#\} = C$

'want'
want: $y_{i,j,a} = 1$ iff (i,j)th cell in tableau is symbol a

[constraints]
$\varphi = \varphi_{cell} \wedge \varphi_{start} \wedge \varphi_{acc} \wedge \varphi_{transition}$
[well-defined] [init] [accept] [transition]
[most interesting]

Michael A. Forbes
Mforbes@illinois.edu
2023-09-05.3

φ_{cell} := each cell is assigned [exactly] one symbol in C

$= \bigwedge_{\substack{1\le i\le t \\ 1\le j\le t+3}}$ [[and]] $\left(\left(\bigvee_{a\in C} y_{ija} \right) \wedge \left(\bigwedge_{a\neq b\in C} \left(\overline{y_{ija}} \vee \overline{y_{ijb}} \right) \right) \right)$ [[clause]] [[and]] [[pairs]]

for each cell; ≥ 1 symbol; $\equiv \overline{y_{ija} \wedge y_{ijb}}$; ≤ 1 symbol

a ∨ b ≡ b ∨ a.

clm: φ_{cell} is - CNF $C_1 \wedge \dots \wedge C_m$ [[AND of clauses]]
- [not] 3CNF, as has many literals

— says

= initialization to $\# q_{start}\, x_1 \dots x_n \sqcup \dots \sqcup \#$

$\varphi_{start} := y_{11\#} \wedge y_{12q_{start}} \wedge y_{13x_1} \wedge \dots \wedge y_{1(n+2)x_n} \wedge y_{1(n+3)\sqcup} \wedge \dots \wedge y_{1(t+2)\sqcup} \wedge y_{1(t+3)\#}$ [[m]]

clm: φ_{start} is 1-CNF, says

= last config is accepting

$\varphi_{acc} = \bigvee_{1\le j\le t+3} y_{tjq_{acc}}$

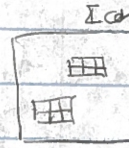
clm: φ_{acc} CNF, says

$\varphi_{transition}$:= each row in tableau [legally] follows from previous row

= - '#' stays '#'
- if cell [not] overwritten by head, it stays same
- " [is] " , is properly written [[transitions forced]]
- head moves correctly, stays within markers #
- state transitions correctly, or remains halted

[[clm]] = every 2x3 window inside tableau is [valid]
is proper action of [some] nondeterministic choice

[[constraints]]



eg:

q_5	a	b
c	q'	b

[[a ↦ c]] [[b ↦ b]] [[q → q']] [[move R]]

$\delta(q,a) \ni (q', c, R)$

q	a	b
q	d	b

[[a ↦ d]] [[stay]]

$\delta(q,a) \ni (q, d, S)$

vs is [nondet]

a	b	c
a	d	c

[[not in window]] [[not unique]]

a	b	c
a	b	q

[[TM moves left]]

vs : correctness forced by [other] window

eg/not:

a	b	c
a	d	c

[[no head]]

a	b	c
a	q	c

[[improper move]]

$= \bigwedge_{\substack{1\le i\le t-1 \\ 1\le j\le t+1}}$ (2x3 window starting at (i,j) is legal)

$= \bigvee_{\text{legal}} \big(y_{ija} \wedge y_{i(j+1)b} \wedge y_{i(j+2)c} \wedge y_{(i+1)jd} \wedge y_{(i+1)(j+1)e} \wedge y_{(i+1)(j+2)f} \big)$

a	b	c
d	e	f

legal

rmk: [not] CNF yet, as is AND ∘ OR ∘ AND

clm: has equivalent CNF via distributive law ← $\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$ [[CNF]]

cor: φ_{trans} has equivalent CNF, computable efficiently

Michael A. Forbes
miforbes@illinois.edu
2023-09-05.4

· ≤

reduction . f = "on input x : – output $\langle \varphi_x \rangle$ "

complexity = [encoding length] $|\varphi_x| \approx |\varphi_{cell}| + |\varphi_{start}| + |\varphi_{acc}| + |\varphi_{move}|$

[Q] [cells][index]
$\Theta(t^2 \lg t)$ $\Theta(t \lg t)$ $\Theta(t \lg t)$ $\Theta(t^2 \lg t)$

can construct in time poly $|\varphi_x|$ = poly n [O(1) time per window] [φ_x highly regular]

correctness =

clm : correct start config and all 2x3 windows of tableau legal
⇒ encodes valid tableau of some branch

· sketch

pf : by induction on time

config 1 : clear

config i : head moves ≤ 1 distance per transition

–

#	·	·
#	·	·

boundaries [and right boundary]

–

a	b	c
·	b	·

static cells

–

a	q	b
·	·	·

$\delta(q, b) \ni (q', c, R)$, (q', c, L), (q', c, S) transitions are observed [no overrun]

⇒ $x \in A$ iff N acc x iff some branch accepts
iff ∃ accepting tableau
iff ∃ satisfying assign to φ_x
iff $\langle \varphi_x \rangle \in$ CNF-SAT [reduction] ☐

thm [Cook-Levin] : CNF-SAT is NP-complete [see textbook]

fact : CNF-SAT $\leq_p$ 3SAT [will another proof later]

idea : introduce auxiliary variables to simulate long clauses by short ones