COMS 4236: Intro to Computational Complexity

Rocco Servedio

Overview of today's lecture:

- quick
- admin overview, web page
- Content: computational problems
  computational model (TMs)
  complexity classes (time, space, nondeterminism)

- Intro to topic + course

- Comput. theory: 1930s. Algorithms.
  Q: what problems can computers solve?

- Complexity theory: 1960. Efficient algs.

  Q: what problems can computers solve with bounded resources?

Examples of q's that cxity theory studies:

(1) How hard is multiplication?
  $m(n)$ = min # atomic steps required by best alg. for multiplying two n-bit #s.

- $m(n) \leq 2^{O(n)}$ (repeated add.)
- $m(n) \leq O(n^2)$

n digits

73142....9

5168.....7

$3^n = 2^{O(n)}$ (n)

- $m(n) \leq O(n^{1.59})$, $O(n^{1.001})$
- 2019: $m(n) \leq O(n \cdot \log n)$
- 2019: under network coding conj., $m(n) = \Theta(n \cdot \log n)$.

(2) Is there an eff. alg. for

- CLIQUE: - input is n-node graph G, param. k;
  - output Y iff G has k-clique

?

- PARTITION: - input is list $x_1, \ldots, x_n$ of n #s
  - output Y iff can split into two disj. subsets w/ same sum.

"NP Completeness": both easy or both hard.

(3) Does randomness make computers more powerful?

(4) Proof - written/static vs interactive.

Admin stuff / web page:

• Unit I: Basics / comput. problems / models
P, NP, time, space, nondet.

Readings: Sipser ch. 3, 7
Arora/Barak: ch. 0, 1.1-1.6
Papad.: 2.1-2.5

Comput. Problems:

→ set of instances (aka # of) : input-output pair.

Ex: mult. Input: two n-bit #s a, b.

Output: # a·b.

We consider time, space, etc. of alg's for our comput. problems as a function of input length n.

$O(n)$? $O(n^2)$? $2^{O(n)}$?

Many types of CP's.

Example: CNF satisfiability.

CNF = AND of ORs of literals (x_i $\overline{x}_i$ Bool vars)

$\cdots\cdots \wedge (\bar{x}_2 \vee \bar{x}_4 \vee x_6) \wedge (x_1 \vee \bar{x}_7 \vee x_3 \vee x_4)$

Input: the CNF formula φ.

$0 = F$, $1 = T$

- Decision problem: is φ satisfiable? (is there an assignment $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ s.t. $\varphi(x) = 1$ (T)?)
- search problem: find an asst. x sat. φ (or report "unsatisfiable").
- counting problem: output # of $x \in \{0,1\}^n$ s.t. $\varphi(x) = 1$.

We'll mostly consider dec. problems.

Any dec. problem is $\equiv$ to a language recog. problem:

$L \subseteq \Sigma^*$ $\quad$ Σ = finite alphabet

Any $L \Leftrightarrow$ a dec. problem: given x, is $x \in L$?

& vice versa.

e.g. dec version of CNF-SAT $\Leftrightarrow$ lang L of all satisfiable CNF formulas.

Need • precise input/output repr. ①
• comput. model. ②

alphabet ↓ Σ

① encode inputs/outputs as strings over Σ;
any reasonable rep. is OK; (not unary).
we won't sweat details.

② Here too details usually not impt;
for concreteness: Turing machines.

TMs

- 1 tape or multitape → fixed # tapes
  $c = 2$ or 5
- finite # internal states ; q_{accept} ; q_{reject}
- Multitape TM:
  - read-only input tape — L/R head movement
  - c R/Write worktapes — " " " "
  - write-only output tapes — R only
  - finite state control, head on each tape.

Local comput's: based on state & contents of tape cells, do update

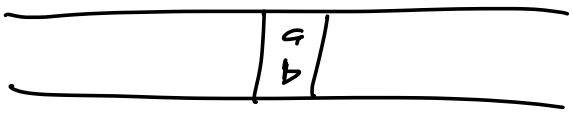
See books for details, conventions, etc.

Recall: Our default model: multitape TM.

- multitape TMs $\equiv$ to 1-tape TMs:

can sim w/ $T(n)$-time multitape $\rightarrow$ $T(n)^2$ time on

one tape machine



- View TM M as computing string-val. function:

x = input string

$M(x) = \begin{cases} \text{contents of output tape if } M \text{ halts on } x \\ \perp \quad \text{o/w.} \end{cases}$

- View TM as acc/dec. languages:

given M, $L = \{x$: on input x, M halts in $q_{acc}\}$.

Def: Lang. L is accepted (recognized) by TM M if $\forall\, x \in \Sigma^*$, M acc. $x \iff x \in L$.

(for $x \notin L$, M may or may not halt).

(L is recognizable / recursively enumerable.)

Def: Lang L is decided by a TM if

M • acc. x if $x \in L$
• rej. x if $x \notin L$.

} M always halts.

(L is decidable / recursive).

Useful TM facts:

① TMs can be listed/enumerated:
every TM can be encoded by a finite string s.t.
(i) every string corr. to some TM;
(ii) " TM occurs ∞ly many times in the enumeration.

② Efficient universal TMs exist: U ($|x|$ = length of x)
on input $\langle M, x \rangle$, U simulates M on x.
If M runs for $T(|x|)$ time steps on x,
then U runs for

$$c_M \cdot \text{poly}(T(|x|)) \text{ steps on } \langle M, x \rangle$$

(even $c_M \cdot T(|x|) \cdot \log T(|x|)$ is doable).

Easy to add "counter": on input $\langle M, x, t \rangle$
U accepts iff M acc. x within t time steps.

We care about resource-bounded comput.

Most basic resources: time + space.

Next time:

- time complexity
  P, EXP, etc.
- space cxity
  L, PSPACE, etc.
- nondeterminism, NP
  NP completeness: Cook-Levin,

few examples.