W11L2: the class P

Wednesday, April 15, 2020 9:22

<u>Announcements:</u>

- quiz 8 is live on gradescope

We want to measure the runtime of a Turing machine as a function of the length of the input string, and we are interested in worst-case runtime.

<u>Why bother measuring TM runtime, when they are so miserably slow?</u>

- with reasonable arguments we can say that multitape TMs are a model for real computational speed (in the real world)
- the trick will be to figure out how much "wiggle room" we have when comparing TMs to real computers
- we'll start by comparing TMs to other TMs using $O(\cdot)$ notation

Let's discuss an example TM for $\{ w\#w \mid w \in \{0,1\}^* \}$.
(Refresher: this is not regular, not context-free, but yes: decidable.)

High-level decider for this language:

1. Scan the input to check for exactly one `#`, and if we find any different number, reject.
   Return to the beginning of the string.
2. Starting at the leftmost character, begin zigzagging, crossing off the first character in each half (if they match); if they don't match, reject immediately.
3. When everything in one half of the string is crossed off,
   - if any 0 or 1 characters are left, reject
   - if every 0 and 1 is crossed off, accept

When we talk about the runtime of a Turing machine, we always want a worst-case upper-bound.

Line 1 will take $O(n)$ because it requires reading the entire string (n steps), and reset to the beginning (n steps), plus maybe one or two more steps for off-by-one moves at the end. $2n + c$ is $O(n)$

Line 2: need to scan to find the first character, so read the first half of the string and then go back --- $\frac{n}{2} + \frac{n}{2} + c = n$.

$\cancel{1}000000\ 0\#\cancel{1}$

in the worst case, scanning for one symbol to cross off takes n; so scan, cross off, return to beginning: $n + n + c = 2n$

In the worst case, we will run this once per character for ~$\frac{n}{2}$ times, so we multiply (time for one cross-off)*(# of iterations) = $(2n) * \left(\frac{n}{2}\right) = n^2$

Line 3: need to scan the entire string so n steps

Overall the runtime is (runtime of line 1)+(runtime of line 2)+(runtime of line 3)
$= O(n) + O(n^2) + O(n) = O(n^2)$

<u>Theorem:</u> If f, g, h are all functions and $f \in O(h)$ and $g \in O(h)$ then the function $f + g \in O(h)$.
$(f + g)(n) = f(n) + g(n)$ is $O(h(n))$.

<u>Def:</u> Let $t\colon \mathbb{N} \to \mathbb{R}^+$ be a function.

The **time complexity class** $TIME(t(n))$ is the collection of all languages decidable by a $O(t(n))$-time one-tape deterimistic Turing machine.

e.g. we can now talk about:
- linear time problems $TIME(n)$ which are problems solvable in $O(n)$-time by a one-tape deterministic Turing machine.
- quadratic time problems $TIME(n^2)$ which are problems solvable in $O(n^2)$-time by a one-tape deterministic Turing machine.

<u>Def:</u> The class P is the set of all languages decidable in **polynomial** time (on a one-tape deterministic Turing machine).

$$P = \bigcup_{\substack{f(n) \\ \text{which is} \\ \text{a polynomial}}} TIME(f(n))$$