L21 2022-11-06

Time Complexity

Want to understand which problems can be solved efficiently

This can mean quickly / little memory / parallelizable / ...

We start with running time.

def Running time of a (deterministic) TM.

$t(x) = \#$computation steps on input x.

$t(n) = \max_{|x|=n} t(x)$.

* obs we are measuring "worst-case" time. Can also measure "average-case".

We don't care about exact runtime, so we use $O(\cdot)$ notation.

Recall $f \in O(g)$ if $\exists c, n_0 \quad f(n) \leq c \cdot g(n) \ \forall n \geq n_0$.

f is a poly if $\exists c$ st $f \in O(n^c)$

is exp if $\exists c$ st $f \in O(2^{(n^c)})$.

e.g. M deciding palindromes:

```
M: input x
   for i = 1..n/2:
       if x[i] ≠ x[n-i-1]: reject
   accept
```

$t(x) = O(|x|^2)$ no matter the input, hence $t(n) = O(n^2)$.

obs a RAM machine only needs time $O(n)$.

this difference is not too large:

th a 1-tape TM can simulate

→ a multi-tape TM in time $O(t^2)$

→ a RAM machine in time poly(t, n).

We only consider 1-tape TMs from now on.

def $DTIME(g) = \{L : \exists M$ deciding L in time $O(g)\}$.

e.g. PALINDROMES $\in DTIME(n^2)$.

(and also $\in DTIME(n^3)$, and $DTIME(2^n)$, ...)

Nondeterministic Time

def $t(x) = \min_{y:\text{ accepts } x} \#$steps of y.

$t(n) = \max_{|x|=n} t(x)$.

e.g. NTM deciding $\neg$PALINDROMES

```
N: input x
   guess i ∈ [1, n/2]
   if x[i] ≠ x[n-i-1]: accept
   else reject.
```

$t(n) = O(\log n + n) = O(n)$.

def $NTIME(g) = \{L : \exists N$ deciding L in time $O(g)\}$.

e.g. $\neg$PALINDROMES $\in NTIME(n)$.