Tuesday.
March 10.

Thursday – Quizz (mini-test)

Friday (3/13) – TCM

Tuesday 3/24 → project report

Thursday 3/26 → Longpre will teach complexity.

Problem:– we need to separate feasible algorithms from non-feasible ones.

Examples :– $t \sim n^2, n^3 \rightarrow$ feasible

$t \sim 2^n \rightarrow$ not feasible.

Definition:– $t_u^w(n) = \max\limits_{x\,:\,len(x)=n} t_u(x)$

An algorithm u is called feasible if there exists a polynomial $P(n)$ such that $\forall n: t_u^w(n) \le P(n)$

Imp. fact:– This definition doesn't fully reflect the notion of feasibility.

<u>Example:</u> $t_u^w(n) = 10^{300} \cdot n$

Common sense viewpoint : not feasible

From our defn : it is feasible (because $10^{300} \cdot n$ is polynomial)

$t_u^w(n) = 10^3 \cdot \exp(\cdot 000001 \cdot n)$

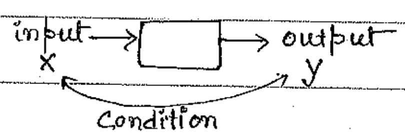
Common sense : feasible [becoz $e^{\cdots}$ is small]

our defn : not feasible [becoz exp is not polynomial]

So, the definition has limitat. No one knows how to produce a better definition.

We can use the knowingly imperfect definin of feasibility.

What is a problem?



<u>In mathematics</u>:–

$X \rightarrow$ statement

$y \rightarrow$ proof of X or $\neg X$.

Comments:– 1. checking whether y is a correct proof of X is relatively easy.
↓
[checking the steps followed in the proof]

2. The length of the proof should be feasible.
↓
[like 4 color problem – length is too much]

bounded by polynomial of (len(x))

condition $R(x, y)$ is feasible means, $len(y) \leq P_l(len(x))$.

we have:—

* feasible predicate relation $R(x, y)$ true or false.
  $\downarrow$ is the step by step stmt of a proof.
* polynomial P_l :

Given:— x binary seq.

we want:— find y such that $P(x, y)$ and $len(y) \leq P_l(len(x))$.

In Physics:—

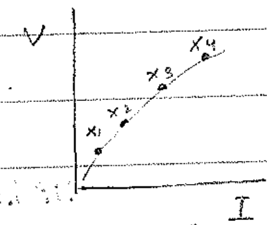
x — observation

y — theory : that explains observations.

R(x, y) — check whether observations are consistent with the data.

here, $len(y) \leq len(x)$ → no of observations

$len(y)$ ↓ The theory.



y: $\boxed{V = R.I}$

x: $x_1, x_2 \ldots x_n$ no of observations

Engineers:—

x — specifications (cost, how much it weight, it tolerate, wind etc)

y — speci' design [like constructing bridge]

R(x, y) — feasible [now-a-days computer simulations help in testing the design like simulating a truck going over bridge]

but coming up with a good design is difficult.

~~len(y) is not so complex here.~~

Humanity:— x — emotions

y — a poem

~~R(x, y) — not formalizable.~~

[how many strings with length $n \to 2^n$
~~with 1 length~~, 2^1 . $[0 \to 0, 1 \to 1]$
2 ", 2^2 $[0, 1 \to 00, 01, 11, 10]$
. .]

In principle, every practical problem can be algorithmically solved by trying all binary strings y of length $\leq P_l(len(x))$.

Problem: we need 2^n time not feasible.

Good news:— If we guess an answer that we can check it's correctness in polynomial time.

A P.R. problem :— can be solved on a non-deterministic TM in polynomial time.

~~practical~~ (P.R. ↓)

Non-deterministic Polynomial = NP

NP = class of all practical problems.

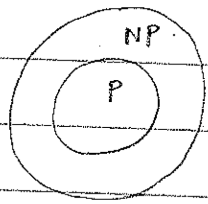
CS problem ÷ X – original list

Y – sorted list.

R(X, Y) ÷ checking if Y is same list as X & n ∈ Y is sorted.
, is easy.

len(Y) = len(X).

P – class of all problems which can be solved by a feasible* algorithm.
example – sorting problem.



Natural question ÷ $NP \stackrel{?}{=} P$

Is it possible to solve any practical problem in resonable time? Ans ÷ we don't know.

NP hard problem ÷ Hardest problems in NP.

By, NP hard we can reduce class of NP to a smaller set of problems.