```
// Java el code :

int positive = 0

for( i=0 ; i<n ; i++ )
{
    positive = 1 ;
}
```

Mar 10, 09

+ Thursday 3/12
  - Go over problems
  - Quiz Min test

+ Friday 3/13 TDCM 7:30 ~~pm~~ am / College Eng

Spring Break. T : 3/24
R : 3/26

Turn in

Project Progress Report

Luc Longpre

Problem : We need to separate feasible algorithm from non-feasible ones.

Examples : $t \sim n^2, n^3 \to$ feasible
$t \sim 2^n \to$ not feasible

Definition :

$$t_U^w(n) = \max t_u(x)$$

$$x : len(x) = n.$$

An algorithm u is called feasible if there exists a polynomial $P(n)$ s.t.

$$\forall n \quad t_u^w(n) \leq P(n)$$

+ Important facts

This definition does not fully reflect the notion of feasibility.

- Ex 1. $t_u^w(n) = 10^{300} \cdot n$

→ common sense - viewpoint : not feasible
→ definition - still polynomial : feasible !

- Ex 2. $t_u^w(n) = \exp(0.0000001\, n)$

→ common sense - feasible
→ definition - NOT feasible

Problem : No one knows how to produce a better definition.

We can use the knowingly imperfect definition of feasibility.

what is a problem.

input → [ ] → output

$x \longleftrightarrow y$
condition

Mathimatics :
x - statement
y - proof of x or $\neg x$.

+ comments:

checking whether y is the correct proof of x is relatively easy.

+ The length of the proof should be feasible.

- Condition $R(x, y)$ is feasible.

$$len(y) \leq P_e(len(x))$$

we have

+ feasible predicate $R(x, y) \rightarrow$ returns T / F.
+ Polynomial P_e:

Given X: statement

we want find y such that

$$R(x, y) \text{ and } len(y) \leq P_e(len(X))$$

+ Physics:

X - observations:

y - theory that explains observation.

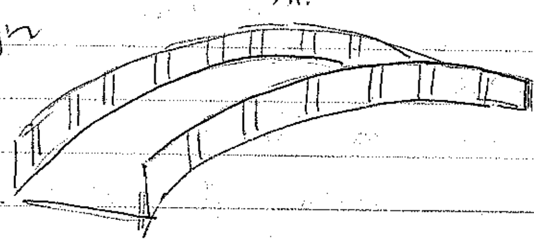
$R(x, y)$: check whether the observations are consistent with the data.

$$len(y) \ll len(x)$$

Engineers

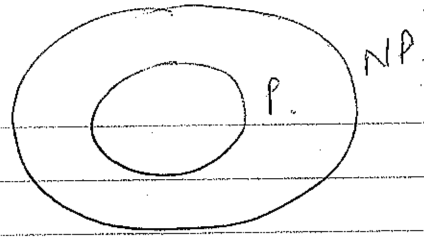
X - specification (cost, ... how much it weight, wind).

y - design



$R(X, y)$:

- feasible.



\+ Humanity X - emotions
y - a poem

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

\+ Problem : We need 2^n time. not feasible.

\+ Good news : if we guess an answer then we can check its correctness in polynomial time

A pr. problem = can be solved on a non-deterministic TM in a polynomial time.

\+ NP = Non-deterministic Polynomial
≡ class of all practical problems.

\- CS problems X - original list
y - sorted list
$R(x, y)$, $\mathrm{len}(y) = \mathrm{len}(x)$

\+ P : class of all problems which can be solved by a feasible algorithm

Is it possible to solve any practical problem in reasonable time ?
→ Open problem / Don't know.

Reduction : if there is an algorithm to solve a problem in class A → also can be used to solve other problems in the same class A.

$NP \stackrel{?}{=} P.$ $ax^2 + bx + c = 0$

$ax + b$