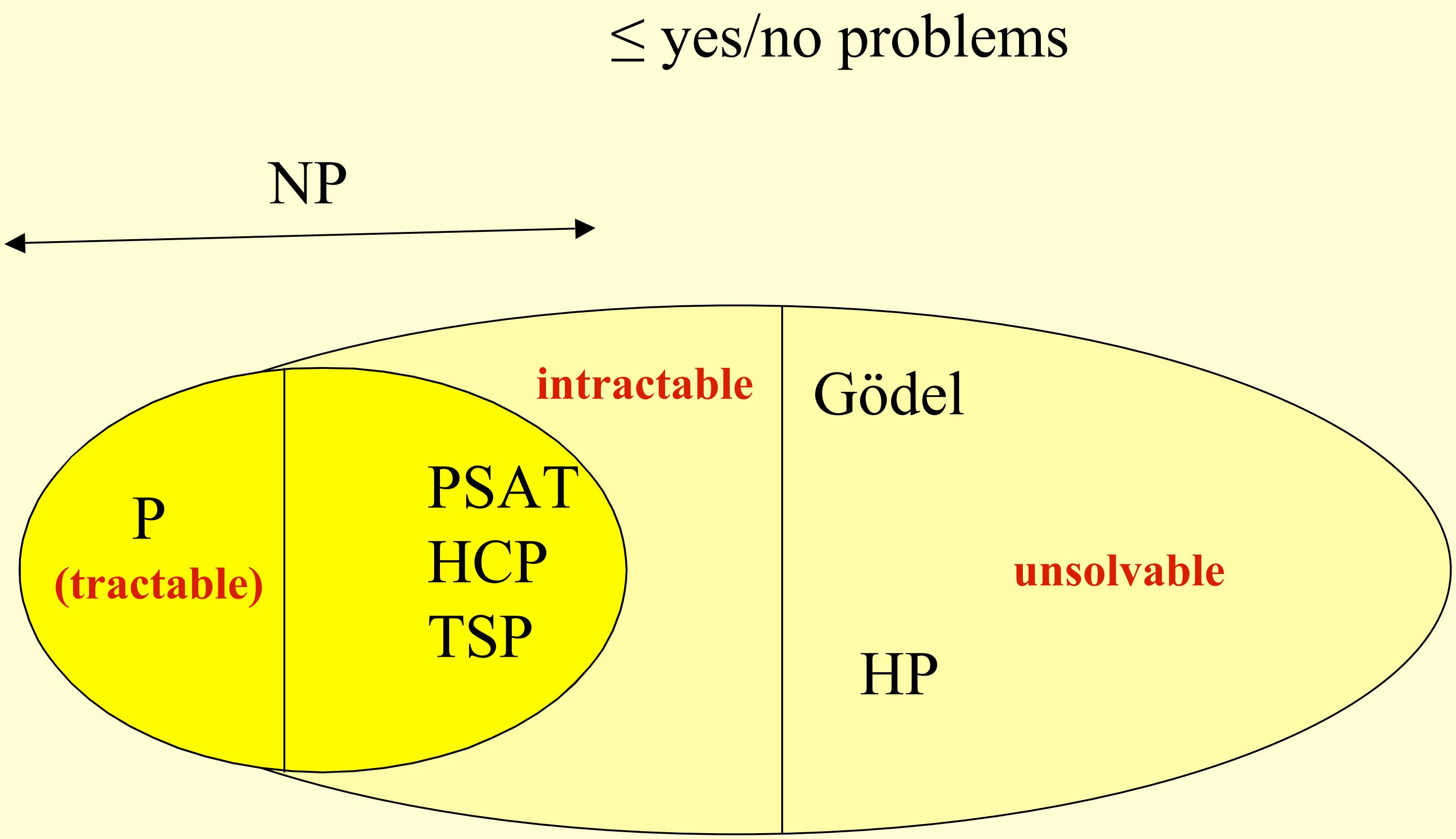


Complexity classes P, NP.

NP-complete problems..NPC

is there a ≤-hardest problem in NP. Or set of hardest problems?
or..a sequence ≤ harder ≤ harder ≤ harder ≤…

there are hardest problems in NP: the NP-complete problems.
Definition of NP-complete:

A yes/no problem A is NP-complete if:

1. $A \in NP$
2. $B \leq A$ for all problems $B \in NP$

i.e. NP-complete problems are problems in NP to which all other NP Problems can be reduced in p-time.

NPC - the class of NP-complete problems.

If A, B are NP-complete then A~B:

A is NP-complete $\Rightarrow$ A $\in$ NP

for all C $\in$ NP, C≤A (including B)

B is NP-complete $\Rightarrow$ B $\in$ NP

for all C $\in$ NP, C≤B (including A)

..so A≤B and B≤A .. **A~B**.

If A is NP-complete and A~B then B is NP-complete:

A is NP-complete $\Rightarrow$ A $\in$ NP

$\Rightarrow$ for all C $\in$ NP, C≤A

A~B $\Rightarrow$ A≤B and B≤A.

NP is closed downwards under ≤ $\Rightarrow$ B $\in$ NP

For any C $\in$ NP, C≤A and A≤B so as ≤ is transitive, C≤B.

so **B is NP-complete**.

A yes/no problem A is NP-complete if:

1. A ∈ NP
2. B ≤ A for all problems B ∈ NP

if we know that another problem C, is NP-complete, we can show

2*. C ≤ A (instead of 2. above)

(remember 1. must be shown: that A ∈ NP.

This permits extension of the set NPC by proving that a known NP-complete problem reduces in p-time to an NP problem thought to be in NPC…but is there a "first" problem in NPC?

to use 1.+and 2*) as proof, we need an existing NP-complete problem

are there any NP-complete problems?

Cook's Theorem proved that PSAT is NP-complete

so NPC ≠ ∅

Cook's Theorem..PSAT is NP-complete

Proved by showing that

1. PSAT $\in$ NP (already seen..see example 10.7 of the notes)
2. any problem in NP can be reduced in p-time to PSAT
   ie that for any problem A $\in$ NP, A $\leq$ PSAT.

let A $\in$ NP $\Rightarrow$ $\exists$ a non-deterministic TM, $N = (Q, \Sigma, I, q_0, \delta, F)$ which solves A

$\Rightarrow$ $\exists$ $p(n)$: $time_N(n) \leq p(n)$

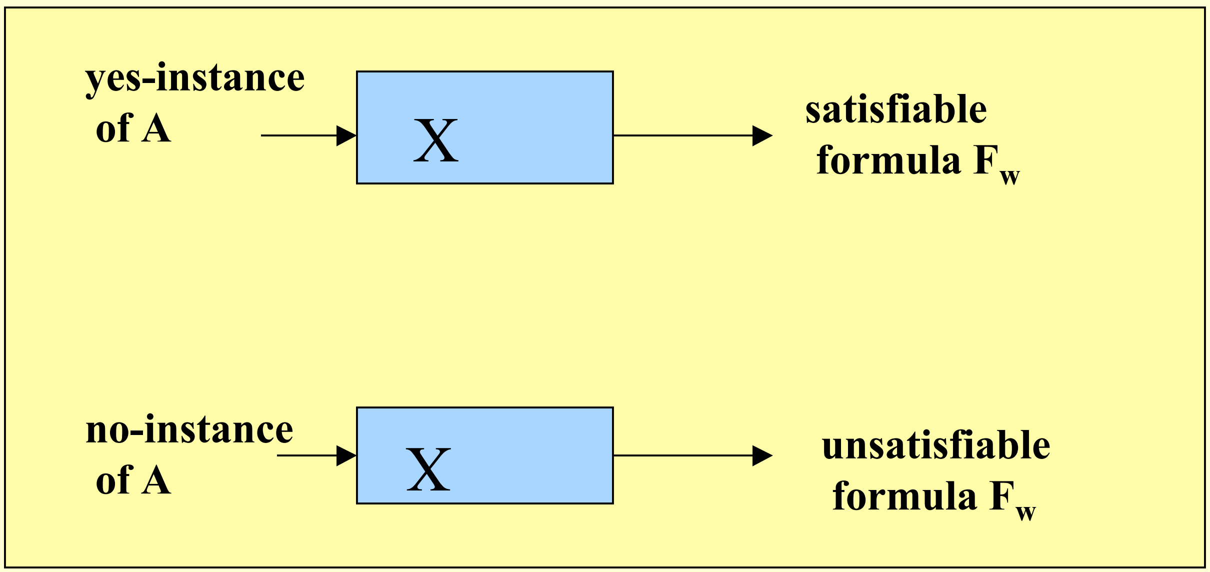
[no run of N on input with length n takes more than p(n) steps.]

for input w a yes-instance of A: N has an accepting run
no-instance of A: N has no accepting runs

we want to reduce A to PSAT in p-time

If we can build a deterministic TM, X which

- given a yes-instance of A, outputs a satisfiable formula F_w of propositional logic (a yes-instance of PSAT)
- given a no-instance of A, outputs an unsatisfiable formula F_w of propositional logic (a no-instance of PSAT)



..and X runs in polynomial time, then X reduces A to PSAT in p-time and so PSAT is NP-complete

The formula F_w where w is a yes-instance of A describes the conditions for N to have an accepting run: for N to Halt and Succeed.

So the formula is satisfiable iff there is an accepting run of N on w…

F_w is a yes-instance of PSAT iff w is a yes-instance of A.
F_w is a no-instance of PSAT iff w is a no-instance of A.

Description of runs of N as a formula of propositional logic:

represent the configuration (state, tape contents, head position) of N at each step of N's run.

…at most p(n) of these for input w of length n.

not all possible configurations

configuration sequences are possible in a run:

Constraints for a sequence of configuration boxes to represent an accepting run of N:

1. For each $t \leq p(n)$ the box for time t must represent a genuine configuration
2. C(0) must be the initial configuration:
   state q_0, head in sq 0, w on the tape

3. The sequence of configurations must represent a complete run of N, as defined by the δ function

4. The run corresponding to the sequence must be an accepting run:
   ..it must end in a halting state.

does N accept w?
= can we complete a sequence subject to the 4 constraints?

Representing the configuration sequence for a run as a Formula in propositional Logic:

- all the entries in the sequence are boolean values (shaded or not).
- describe the constraints for an accepting run:

  F_w - formula expressing constraints on the truth values of atoms

a valuation of atoms which makes F_w true

≡ a valid table/sequence of configurations

≡ an accepting run for w.

a yes-instance of A corresponds to a valuation of the atoms giving F_w the value true

this is reduction of A to PSAT..

does X run in p-time?

X is deterministic and runs in p-time

F_w depends on w and the □-function of N. It can be derived by a deterministic algorithm from w and □..
..so can be done by a deterministic Turing Machine. This is X.

running time of X..

Suppose $Q = \{q_0, q_1, q_2, q_3, \ldots q_s\}$, $\sum = \{a_0, a_1, a_2, \ldots a_r)$
number of atoms X writes in F_w
$\leq$ (p(n)+1) (maximum number of steps in a run of N solving A)

x ((r+1)(p(n)+1) (max length of tape x size of alphabet)
+ p(n) +1 (head position)
+ (s+1)) (number of states)
= (p(n) + 1)(p(n)+1)(r+2)+s+1)..a polynomial.

Summary

We have defined:

non-deterministic Turing machines (and shown their equivalence to ordinary TMs).

time functions of TMs and NDTMs, $time_X(n)$ for TM X, where n is the length of the input.

polynomial-time Reduction: $time_X(n) \leq p(n)$, p a polynomial.

complexity classes of yes/no problems:

P..solvable by a p-time deterministic TM

NP.. solvable by a p-time non-deterministic TM

NPC..(the ≤ hardest problems in NP)..problems in NP to which all other problems in NP can be reduced in p-time.

Cook's Theorem proves that NPC ≠∅

≤ is a pre-order: a reflexive, transitive, binary relation

P and NP are closed downwards under reduction (≤)