NP-Complete and NP-Hard Problems

• A problem is an the class NP-Complete (NPC) of it is in NP and is as hard as any problem in NP.

A problem is NP-hard if all the problems in NP are polynomial time reducible to it, even though it may not be in NP itself.

· Every problem in NP can be reduced to another NP problem, in a	
POMII MADILLI DI MADALI	
. If a problem in NP-Complete class can be solved using a polynomial	
o If a problem in NP-Complete class can be solved using a polynomial deterministic algorithm, then all problems in NP can be solved in	
polynamial time.)
Mil-complete	187
Reduction	1
· A problem 'A' can be reduced to another problem B' if	
and sutous of A can be rephrased transformed into any	
instance of B, the solution to which provides the solution to the	
instance of A.	
Intuitively: If A reduces in polynomial time to B, A is "no have	der to solve
than B.	w. 9 7 4
In other words:	
· Suppose we already know how to solve a decision problem & in poly	no mial time
· suppose we have a procedure that transforms an austance & of A	nto some
instance & of B with the foll-characteristics.	• In
as The transformation takes polynomial time, ie,	
there exists a function of that converts the superts of A to SIPS of B in	eolynomial
(is A(i) = YES (B(i) = YES, ic,	time.
the answers are same; the ans. for a is "yes" iff the ans. for pe	also yes"
· This procedure is called polynomial time reduction algorithm, and	we
say that A is polynomially reducible to B. (A ≤ p B)	A A T
instance & polynamial-time instance & polynamial time yes	
of A > reduction algorithm of B algo to decide B 20	
polynomial time algorithm to decide A	
· It provides us a way to solve problem A in polynomial time:	1
1. buven an sustance of of A, use polynomial time reduction who to to	uslorn it
to an instance & of problem B.	
/ Company of the comp	Β.
2. Rure the polynomial time decision algorithm for B on the distance. 3. Use the answer for B as the answer for a.	
:3TAO	
PAGE NO.	

· By "reducing" rolving problem A to solving prob. B, we use the "easin	us" of B
to prove the "earners" of A.	-
· · · · · · · · · · · · · · · · · · ·	
NP-Completences (formally)	
· A problem B is NP-Complete if:	
(i) B E NP	
(ii) A ≤ p B for all A ∈ NP.	
· If B satisfies only property in, but not necessarily property is,	
Bis known as NP-Hard	
· Informally, a search publis B is NP-Hard of there exists some NP-co	and to
problem A that turing reduces to B.	maraca
· No polynomial time algorithm has been discovered for an NP-Comple	to broblem
· The problem in NP-Hard cannot be solved until P=NP.	<u> </u>
The probleme that cannot be solved by any algorithms are called	
Unde cidable Problème	
egs:	
NP-Hard Matrix Permanent / Halting problem	
Hamiltonian Cycle	
steiner tree NP-Complete	
Sotiefiability No factoring Factoring Fraph isomorphism	
Man dique Proph isomorphism	
2 Stricture sage	
Graph connectivity	
Primality testing Motrix determinant	
Linear programming	