f and NE

So far in the course we studied algorithms that solve computational problems efficiently. We saw liveau fine alg (O(n)), quasi-theor $(O(n\log n))$, quadratic $O(n^2)$ and $cubic (O(n^2))$. (common, to all these alg. is that they take polynomial time to run as function of the input size.

Complexity theory tries to enleastered and classify computational problems. For this In particular we wish to understand which problems can be solved efficiently. One can define reflicient in many ways. E.y. as being computable in linear than But clearly, some problems take more than then then. (although we can only prove this in the very special cases). A more natural definition is that of being solvable in polynomial time.

Clearly an olg that takes the now is not very officient and could not be run in practice, but the definition still makes sense as

I in many cases we do get better algorithms and

2. it is closed under composition for example it a poly-time algorithm algorithm poly-time algorithm algorithm algorithm algorithm algorithm and the result is still poly-time algorithm.

3. Different whitectures can solve back problems more efficiently and jet speedup, but when we study the asymptotic complexity as the input size grows, it is still polytime on both different machines or an neither.

The class of all compitational problems that have poly-time alg. is denoted P.

Notice that when we speak of a comp. prob. we man to say that it is a problem

that wakes source for growing imput sizes. E.g. sorting constatingful at an exotherhole

makes source for any input size. In matrix-multiplication the input size is 2n2 (for

multiplify the even metrices and again we can stoly what happens when now.

However many important problems det us une not known to be in P nor are believed to be there.

For example consider the vertex cover problem:

Input: A graph (undirected) and a number K.

boal: determine if I very vie sit VEEE one of its end point is in luned

In other words, we wish to find a minimal set of notices that fouch all edges.

P (and NP) concern decision problem, i.e., YES/NO problems, this is why we don't usk for searchly the minimum site, but rather ask whether it is sk.

(with binary search we can find best site-n).

This is important as it abstracts many scenarios. They we don't know at an efficient alg. for solving this difficulty problem. On the other hand, notice that if one would claim that such a cover exists and would present us with a solution the we'll be able to verity it efficiently.

Another example is independent-set (IS).

An ind. set. in a graph 6 is a subset I = V s.t. there are no edges between vertices in I.

Inpt: G. E

Goal: determine whether 6 my IS of size >K.

Again, no officient alg. Known, but a solution can be resified efficiently.

Another important problem is 3SAT. Here we study boolean formulas of the form CIACIA ... 11 Cm where each Ci is a clewe on 3 variables XVYVE. Recall' each var can take a T/F val-e. XVYVZ Vs evalutes to T it X=T or X=T or Z=T (=) Z=F). Example: (X vyvz) A (X v y vz) A (x v y vz) A (y v t v x) X=T, Y=F, Z=T gives a true value. Thus, it is a satistying wsignment. Input: a 35AT formula on h vars Goal: determine whether I a sat, arrigurant. No officient ulp. known, but a solution can be veritized easily. So we saw thme diff, problems that have no efficient alg. but i. h. a solution for them can be nextitled efficiently, Sadulco Cray non size one sums! is another such problem. NP is the class of election problems for which a solution can be veritial etticiently. NP stands for pondeteraristic poly-time, where the nealeterminism comes from the need to "gress" a solution or from being handed one "nor deterministically" The amoring fact about the problems that we neethered is that it on it then

is in P the all of NP is!

The famous Prs. Ne problem was exactly this! is P=NP!

Such problem, whose containment in P implies P=NP, are collect NP-hord problems. An NP-hand problem that is also in NP is called an NP-amplife problem.

Thur: 354T, IS, Vertex Cover was NP complete.

We will not prove this both as it requires more formal definitions than what a have given (and more time) but rather explain how one prover such a thing. The important concept is that of a REDUCTION hetumen computelyan) problem. That is, it is a poly-thun computable function that takes inputs to one problem and transforms them to imputs to a second problem.

We already seen a vedection between SSSP and SDSP. Given an input to SDSP we reverse the edge directles (can be done easily) and solve the SSSP problem on the resulting graph.

Here is another reduction! SEV is a vertex cover iff VIS is ind, set.

Thus, it we can solve IS(GIC) then we causoline VC(G, h-k),

and vice versa.

Showling that 35ATSIS is more tricky. We will show glasm a formula MC; how to get an instance of IS.

how to get an instance of IS

XVYVE XVEVE

XVYVE

XVYVE

XVEVE

X

Each classe years a triangle. In abbition each appearance of v is compted to v

bused on it.

13.5 Charles We get a graph on 3m restices. claims G has Is of size in iff MC: is sat. Pf! A sat. ussgn. pides (at least) one var from each clause. we never pick both v mel v. (set ass. => Is) (IS => SAT). AT IS much contain exactly one vor from each \ This gives an ass. In the natural way (no ambiguity because of v-D The general pf. Hat 35AT is NP complete follows from showing that the steps of any poly-time olg. can be encoded using a beclear formula. What about problems between P and NP: Import: An a digit number N, integer 1c. where Pak. Goal: Determine whether N=P.Q Hearter grin number P.Q. This is in NP since them is a (nontrivial!) alg. for deciding primality.
(we can give Ms & prime fecture and multiplicities as preef). It is not believed to be NP complete as it will imply other things that are not believed (b.t it P=NP than it's complete ...). Yet, it is believed to be hard and the famous RSA encryption schoene is

hutter for foot Sacurity of crypto currency, situin, ethereum atc., is bread on PTND Lesson When you can't find on ethicient algo, try to fill graf of NI completeness. The gradices many NP problems was important and becausely so other. I nerristics (algo with no proof, or that make her some importal other are useed to proofice. There is a formal SAT-solving congretation between SAT-solvers. They well in practice, but we lesson how to come up. The head higher for the	13.6	
hesson's when you can't find an efficient algorithm to find proof of NE completeness. In practice: many NP problems we important and necessary so that a new important and necessary so that imports) that are used in practice. There is a formas SAT-solving competables between SAT-solvers. They well in practice, but we know how to came up with heard higher has the		
The practices many NP problems are important and hecessary so that a herristics (also with no pract, or that work has some important that work has some important that are used in practice. There is a famous SAT-solving computation between SAT-solvers. They well in practice, but we know how to came up with head higher has the		based on PFNPI
The practices many NP problems are important and becassary so that a herristics (also with no pract, or that make her some imports) that are used in practice. There is a former SAT-solving computation between SAT-solvers. They well in practice, but we know how to came up with head higher har then		
The practices many NP problems are important and hecessary so that a herristics (also with no pract, or that work has some important that work has some important that are used in practice. There is a famous SAT-solving computation between SAT-solvers. They well in practice, but we know how to came up with head higher has the	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	hesson's when you can't find on etticient algority to find proof of NE
nerristics (als, with no prof, or that work har some inputs) that are used in practice. There is a formas SAT-solving computation between SAT-solvers. They a well in practice, but we know how to came up with bond higher for them		
nerristics (als. with no prof. or Heat work how some inputs) that are used in practice. There is a formus SAT-solving computation between SAT-solvers. They well in practice, but we know how to came up with bond higher has the		
Thur is a famous SAT-solving competation between SAT-solvers. They awall in practice, but we know how to come up with bent highs for the		<i>'</i>
Thur is a famous SAT-solving computable between SAT-solvers. They well in practice, but we know how to come up with hour highs for the		·
well in practice, but we laway how to come up with hand higher tree		inputs) that are used in practice
well in practice, but we laway how to come up with heart hapts for the		
well in practice, but we laway how to come up with hand higher tree	na (mana at 1999) (mana at 1999)	There is a famous SAT-solving computation between SAT-solvers. They c
	an yana manala ara san ta mara ara a ara a Ana a an	
	1,11,11,11,11,11,11,11,11,11,11,11,11,1	
	ennesialis lineates en semesa, en en en eterro	
	25,111111111111111111111111111111111111	
	ariini ahaa ahaa ahaa ahaa ahaa ahaa ahaa	
	ana katamatan ka kata ka kata ka	
	ga garannan ga mailgar na chara na mhhair a t-thrachail	
	والمناوات	
	11 - 12 - 14 - 14 - 14 - 14 - 14 - 14 -	