CS 340 – lecture 31 – Nov 27

3 things to prove:

(i) red is poly time in the size of its input

here the size of the input is size(G) $(=|V|+|E|)$

(ii) if G has an HC then G' has a TSP tour of length $(\leq)$ k ✓

(iii) if G has no HC then G' has no TSP tour of length $(\leq)$ k ✓

Definition 20. A decision problem $\mathcal{P}$ is in (the complexity class) NP if there is a (deterministic) algorithm with runtime in $O(p(N))$, for some polyn. p, which, for any problem instance x of $\mathcal{P}$ and any "potential witness" y test whether or not y witnesses a "yes"- answer to x. Here N is the size of x.

S. Cook, 1971: There are hardest problems in NP!

Definition 21. A dec. problem is NP-complete if

(i) it is in NP and

(ii) every problem in NP is polynomially reducible to it.

Such problems seem intractable, since no poly-time alg. finding solutions to them are known.

NOTE If $\mathcal{P}$ is $\mathbb{NP}$-complete and

$\mathcal{P}'$ is in $\mathbb{NP}$ and

$\mathcal{P}$ is polyn. reducible to $\mathcal{P}'$

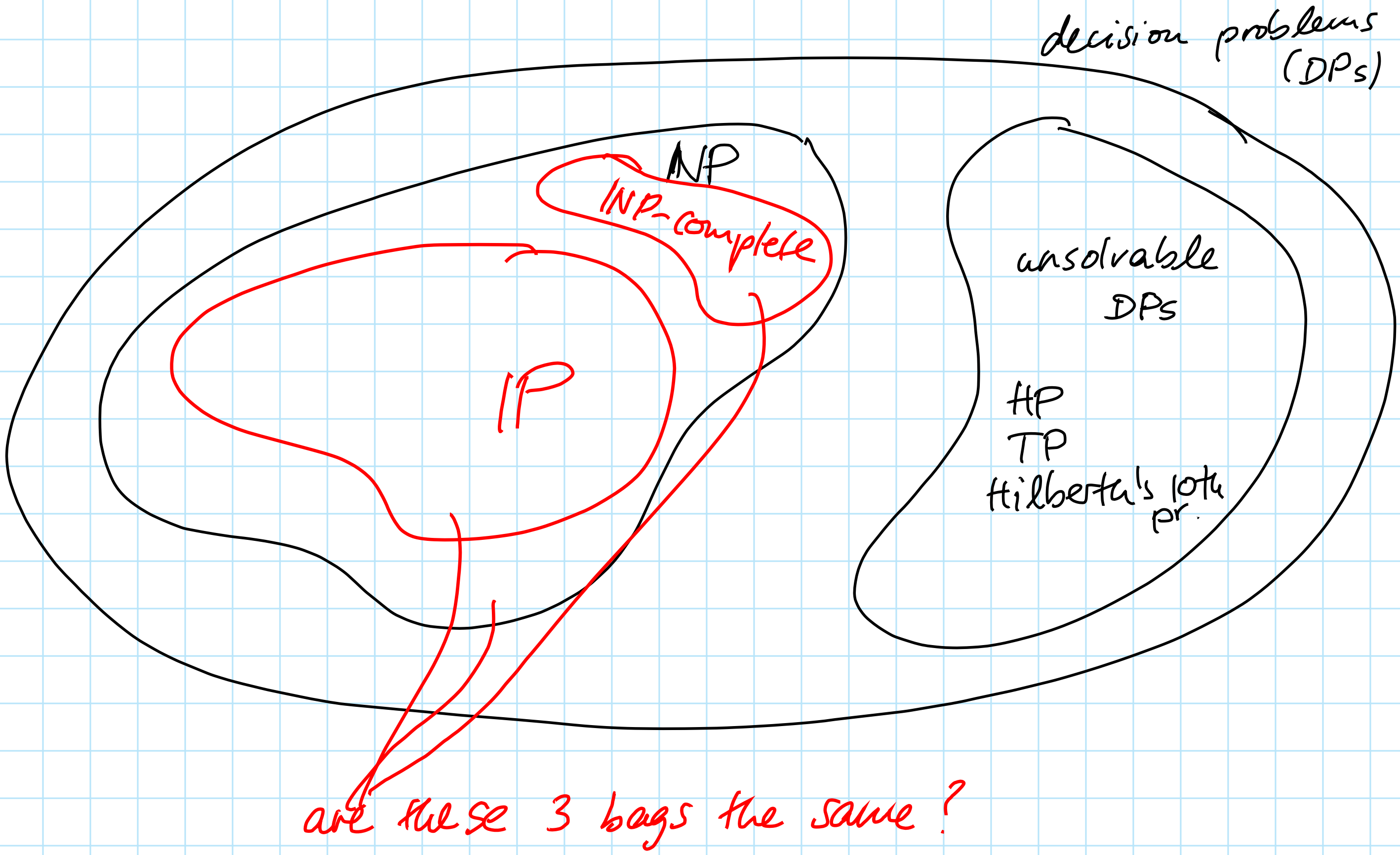
then $\mathcal{P}'$ is also $\mathbb{NP}$-complete.

Example 53. HCP is $\mathbb{NP}$-complete.

Since TSP is in $\mathbb{NP}$ (why?)

and HCP is polyn reducible to TSP,

thus also TSP is NP-complete.



open problem: is $\mathbb{P} = \mathbb{NP}$?

so far, for all known $\mathbb{NP}$-complete problems, we do know exponential-time algorithms, but not poly-time alg. $\rightsquigarrow$ it seems $\mathbb{P} \neq \mathbb{NP}$...?

6. ALGORITHM DESIGN TECHNIQUES

6.1. PARALLEL ALGORITHMS:

see posted notes

6.2. GREEDY ALGORITHMS

proceed in stages; in each stage make the choice that locally looks best

see e.g., Dijkstra, Kruskal, Prim

very natural, see e.g., giving change in coins

40 cents in fewest possible coins

NOT a good approach for every problem!