Complexity, P and NP

EECS 477

Lecture 21, 11/26/2002

Last week

- Lower bound arguments
  - Information theoretic (12.2)
    - Decision trees (sorting)
  - Adversary arguments (12.3)
    - Maximum of an array
    - Graph connectivity
    - Median

Linear reductions

- A is linearly reducible to B (A<=B) if the existence of a O(t(n)) algorithm for B implies the existence of O(t(n)) algorithm for A
- When both ways we get linear equivalence
  - Ex: SQR and MULT
    - $x^2 = x*x$
    - $x*y = ((x+y)^2 - (x+y)^2)/4$

Smoothness matters: see the book f(bN) = O(f(N)), for all integer b>=2

Polynomial vs non-polynomial

- Linear reduction works for polynomial time algorithms
- Polynomial = efficient
- Distinguish efficient from the rest
  - Allow polynomial reduction
  - Polynomial number of polynomial-time operations takes polynomial time
  - Versus exponential time

Decision problems

- Technically easier to handle decision problems
  - Answer is “yes” or “no”
- Example:
  - TSP
    - Find the tour of the minumum cost
  - TSPD: decision version
    - For K, is there a tour of cost <=K
    - We can **verify** given an example

Hamiltonian cycles

- Hamiltonian path
  - Goes through every node once
  - Hamiltonian cycle
- HAM
  - Find Hamiltonian path if one exists
- HAMD
  - Is a given graph Hamiltonian?
    - Do not have to present the path

Complexity classes

- Two classes
  - P
    - The class of decision problems that can be solved by a polynomial-time algorithm
  - NP
    - The class of decision problems that admit a proof system $F \subseteq X \times Q$, poly-time algorithm A
      - 1: $(\forall x \in X)(\exists q \in Q)$ s.t. $(x,q) \in F$ and $\#q \le p(\#x)$
      - 2: $(\forall (x,q))$ algorithm A can verify whether $(x,q) \in F$
    - Q - certificates (there are q for for "yes" instances x only), X - "yes" instances

In other words,

- P – class of decision problems that can be solved by a polynomial-time algorithm
- NP – non-deterministic polynomial time
  - Given a solution, it can be checked in polynomial time
    - given a cycle/tour – check?
    - Composite number: given a factor easy to check (but to find one?)

Theorems, conjectures

- Theorem: $P \subseteq NP$
  - If we can solve a problem then we can surely check it
- The central open question:

  Is P=NP or not?
- Conjecture: $P \neq NP$
  - Look at the hardest problems in NP
    - As hard as any other problem in NP

Polynomial reduction

Polynomial reductions

- Two problems A and B
- $A \leq^p B$:
  - A is polynomially reducible to B
  - There is an algorithm for solving A in time that would be polynomial if we could solve arbitrary instances of B in unit time
  - If both ways then they are polynomially equivalent: $A \equiv^p B$
  - Transitive: if $A \leq^p B$ and $B \leq^p C$ then $A \leq^p C$

Example

- HAM $\equiv^p$ HAMD
  - HAMD $\leq^p$ HAM
    - Trivial: if HAMalgo finds a cycle then yes
  - HAM $\leq^p$ HAMD
    - First check if HAMDalgo gives yes for the original graph
    - Start considering edges for removal one by one
      - Apply HAMDalgo to the remaining
      - If still Hamiltonian without an edge then remove it
      - Otherwise remove the edge and keep going
      - Stop when a cycle is left, return it

Reduction function

- Two <u>decision</u> problems $X \subseteq I$ and $Y \subseteq J$
- F: map $I \rightarrow J$

  such that $F(x) \in Y$ if and only if $x \in X$

  Theorem: If F is computable in polynomial time then $X \leq^p Y$

```
bool DecideX(x) {
    y = F(x);
    if(DecideY(y)) return true;
    else return false;
}
```

Example

- HAMD $\leq^p$ TSPD
  - Given a graph G=(N,A), need to see if it is Hamiltonian
  - Define F(G) be the TSPD instance with a complete graph (N,N$\times$N)
    - Cost = 1 if the edge in A and 2 otherwise
    - TSPD bound being N
    - If TSPD yes then that is a Hamiltonian cycle
    - If TSPD no then no Hamiltonian cycle

NP-completeness

- Decision problem X is NP-complete
  1. X is in NP
  2. Y $\leq^p$ X for every problem Y in NP
- X is polynomially harder than any other NP problem
- If we know that X is NP-complete and X $\leq^p$ Z then Z is NP-complete
- If we could only find one such X

SAT: satisfiability

- Given a boolean formula
  - Is it satisfiable?
    - is there an assignment of values to variables that will make it true?
    - e.g. $(p \wedge q) \Rightarrow (p \vee q)$ is satisfiable via (p=q=true)
    - No efficient algorithm known
  - CNF: conjunctive normal form
    - e.g. $(p+q+\neg r)(p+\neg q+\neg t)(\neg p+q+\neg r+t)p$
    - SAT-CNF satisfiability for boolean expressions in CNF form

Cook's theorem

- For any NP problem Y, $Y \leq^p$ SAT-CNF
  - Proof:
    - Any decision problem in NP has a decision algorithm A_y that checks a certificate
    - A_y is given by a non-deterministic one-tape Turing machine program
    - Can construct polynomial size boolean CNF formula from that program
    - "Formula is satisfiable" = "Instance y is in Y"
    - No more details here

Some NP-complete problems

- SAT
- 3SAT: clauses have three variables
- 3DM: 3D matching
- HAMD: hamiltonian circuit
- PARTITION: set A and $s:A \rightarrow Z^+$
  - Partition A into two equally sized parts
- CLIQUE: clique of size J
- VERTEX COVER: of size K
- K-COL: graph colorability with K colors

NP-hard

- X is NP-hard
  - if there is an NP-complete problem Y that can be polynomially reduced to X
    - $Y \leq^p X$
  - Does not have to be a decision problem
  - Decision problem can be NP-hard but not in NP, for instance exact K-colorability
    - Any K-coloring is a certificate for K-COL but not for K-COLE(exact)