IE170: Algorithms in Systems Engineering: Lecture 32

Jeff Linderoth

Department of Industrial and Systems Engineering
Lehigh University

April 23, 2007

This Time

- A whirlwind tour of computational complexity
- You are not responsible for this material on the final, but it is stuff that I thought you might like to know.
- It is also covered in Chapter 34 of your textbook.

Computational Complexity

- The ingredients that we need to build a theory of computational complexity for problem classification are the following
  1. A class $\mathcal{C}$ of problems to which the theory applies
  2. A (nonempty) subclass $\mathcal{E} \subseteq \mathcal{C}$ of "easy" problems
  3. A (nonempty) subclass $\mathcal{H} \subseteq \mathcal{C}$ of "hard" problems
  4. A relation $\lhd$ "not more difficult than" between pairs of problems
- Our goal is just to put some definitions around this machinery
  - Thm: $Q \in \mathcal{E}, P \lhd Q \Rightarrow P \in \mathcal{E}$
  - Thm: $P \in \mathcal{H}, P \lhd Q \Rightarrow Q \in \mathcal{H}$

Ingredient #1 — Problem Class $\mathcal{C}$

- The theory we develop applies only to decision problems
- Problems that have a "yes-no" answer.
  - **Opt:** $\max\{c^T x \mid x \in S\}$
  - **Decision:** $\exists x \in S$ such that $c^T x \geq k$?

Example: Hamiltonian Circuit

Instance: Graph $G = (V, E)$
Question: Does G contain a Hamiltonian Circuit?

Example: Traveling Salesperson

Instance: Graph $G = (V, E)$, Integer K
Question: Does G contain a Hamiltonian Circuit of length $\leq K$?

Ingredients #2 and #3

- To define "easy" and "hard", we need to make a few definitions so we can define the running time of an algorithm.
- The running time of an algorithm depends on size of the input. (Duh.)
- A time complexity function specifies, as a function of the problem size, the largest[1] amount of time needed by an algorithm to solve any problem instance.
- How do we measure problem size?
  - The length of the amount of information necessary to represent the problem in a *reasonable* encoding scheme.
  - Example: TSP, N, c_{ij}
  - Example: Knapsack: N, a_j, c_j, b

[1] Here is our "worst case"

What is Reasonable?

- Don't be stupid (pad the input data with unnecessary information)
- Represent numbers in binary notation.
  - That's how computers do it anyway
- An integer $2^n \leq x < 2^{n+1}$ can be represented by a vector $(\delta_0, \delta_1, \ldots, \delta_n)$, where $x = \sum_{i=0}^{n} \delta_i 2^i$
- It requires a *logarithmic* number of bits to represent $x \in \mathbb{Z}$
- We always assume that numbers are *rational*, so they can be encoded with two integers.

- TSP on n cities with costs $c_{ij} \in \mathbb{Z}$, $\max_{i,j} c_{ij} = \theta$, then requires $\leq \log(n) + n^2 \log(\theta)$ bits to represent an instance.

Ready for (Somewhat Formal) Definitions

- Given a problem P, and algorithm A that solves P, and an instance X of problem P.
  - $L(X) \equiv$ The length (in a reasonable encoding) of the instance
  - $f_A(X) \equiv$ the number of elementary calculations required to run algorithm A on instance X.
  - $f_A^*(l) \equiv \max_X \{f_A(X) : L(X) = l\}$ is the *running time* of algorithm A
- If $f_A^*(l) = O(l^p)$ for some positive constant integer p, A is polynomial

More Definitions

- A is strongly polynomial if $f_A^*(l)$ is bounded by a polynomial function that does not involve the data size (magnitude of numbers).
- A is weakly polynomial if it is polynomial and not strongly polynomial. The l in $O(l^p)$ contains terms involving $log\ \theta$
- An algorithm is said to be an exponential-time algorithm if $f_A^*(l) \neq O(l^p)$, for any p

One Last Type of Polynomiality

- A *pseudopolynomial algorithm* A is one that is polynomial in the length of the data when encoded in *unary*.
  - *Unary* means that we are using a one-symbol alphabet. (not binary)
- Practically, it means that A is polynomial in the parameters and the magnitude of the instance data θ—*not* $\log \theta$.
- Example: The Integer Knapsack Problem
  - There is an $O(Nb)$ algorithm for this problem, where N is the number of items and b is the size of the knapsack.
  - This is not a polynomial-time algorithm
  - If b is bounded by a polynomial function of n, then it is

Knapsack In More Detail

- **Knapsack**: N, a_j, c_j, b
- For an instance of **Knapsack** X, what is the length of the input L(X)?
- What are the numbers c_j, a_j, b? Assume they are *rational*.
  - So they can be expressed as the ratio of two integers.
  - Assume $a_j \leq b$
  - $\theta = \max_{j \in N} c_j$
  - $L(X) = \log N + (2N+2)\log b + 2N \log \theta$
- Is $Nb = O(L(X))$?
  - $\exists p \in \mathbb{Z}$ such that $Nb \leq ((2N+2)\log b)^p$?
  - **No!**
  - Note if Nb replaced by $N \log b$, then **Yes!**

The problem class $\mathcal{NP}$

- $\mathcal{NP} \neq$ “Non-polynomial”
- $\mathcal{NP} \equiv$ the class of decision problems that can be solved in polynomial time on a non-deterministic Turing machine.
- What the Heck!!?!?!?!?!?!?!?!?
- $\mathcal{NP} \approx$ the class of decision problems with the property that for every instance for which the answer is “yes”, there is a short certificate
- The certificate is your “proof” that what you are telling me is the truth

$\mathcal{NP}$: Examples

Example: Hamiltonian Circuit

Instance: Graph $G = (V, E)$
Question: Does G contain a Hamiltonian Circuit?

- You say the answer is “Yes”. I say “prove it.”
- You give me the a set of edges $E' \subseteq E$. I check as follows:
  1. Does the degree of each node of $G' = (V, E') = 2$? If not, then return **no**, else go to 2.
     - This takes time $\leq O(|V|^2)$.
  2. Is $G' = (V, E')$ connected. If so, return **yes**, otherwise return **no**.
     - This takes time $O(|E'|)$
- The checking algorithm takes $O(|V|^2 + |E'|)$ time, so it is polynomial. It returns **yes** if and only if the set of edges E' defines a Hamiltonian Circuit in G, so Hamiltonian Circuit $\in \mathcal{NP}$.

$\mathcal{NP}$: Examples

Example: Complement of Hamiltonian Circuit

Instance: Graph $G = (V, E)$
Question: Does G *not* contain a Hamiltonian Circuit?

- You say the answer is "Yes". I say "prove it."
- Equivalently, you say that the answer to Hamiltonian Circuit on G is **no**.
- You give me... ?
  - Careful: Will your answer suffice for *all* graphs G?
  - What you really are giving would be a *characterization* of what graphs are *not* Hamiltonian: G is *not* Hamiltonian if and only if Your Answer.
- No one knows!

$\mathcal{NP}$: Examples

Example: 0-1IP

$\exists x \in \mathbb{B}^n$ such that $Ax \leq b, c^T x \geq K$?

1. You say the answer is "Yes". I say "prove it."
2. You give me the vector x: This is a "short certificate"
3. The 0-1 vector x can be checked such that $Ax \leq b, c^T x \geq K$?
4. If $A \in \mathbb{R}^{m \times n}$, this takes time $O(mn^2)$

The Class co-$\mathcal{NP}$

Example: 0-1IP

$\nexists x \in \mathbb{B}^n$ such that $Ax \leq b, c^T x \geq K$?

1. You say "no." I say "prove it."
2. You give me what? Is this a short (polynomial length) certificate?

co-$\mathcal{NP}$, More examples

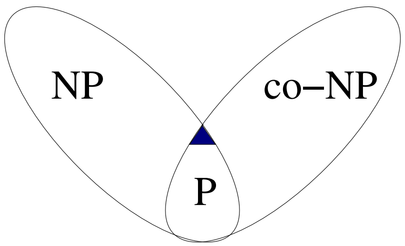
LP

$\exists x \in \mathbb{R}^n_+$ such that $Ax \leq b, c^T x \geq K$?

1. You say "no." I say "prove it."
2. You give me What?
3. Hint: (x, π) is optimal if and only if $Ax \leq b, x \geq 0, \pi^T A \geq c, \pi \geq 0, c^T x = b^T \pi$
4. $\exists \pi \in \mathbb{R}^m$ such that $\pi^T A \geq c, \pi \geq 0, \pi^T b < K \Rightarrow \nexists x \in \mathbb{R}^n$ such that $Ax \leq b, x \geq 0, c^T x \geq K$
5. Is π a short certificate?

The Class $\mathcal{P}$

- $\mathcal{P}$ is the class of problems for which there exists a polynomial algorithm.
- $\mathcal{P} \in \mathcal{NP} \cap$ co-$\mathcal{NP}$: Why?



- It is a (very significant) open question as to whether $\mathcal{P} = \mathcal{NP} \cap$ co-$\mathcal{NP}$.
- There are (very few) problems in $\mathcal{NP} \cap$ co-$\mathcal{NP}$ but not (known) to be in $\mathcal{P}$.
  - LP
  - PRIMES
  - Approximating the shortest and closest vector in a lattice to within a factor of $\sqrt{n}$

Where are we?

- We have our class(es) of problems $\mathcal{P}, \mathcal{NP}$, co-$\mathcal{NP}$
- We know class of "easy" problems. (Problems in $\mathcal{P}$)
- We need our class of "hard" problems.
- We need our relation "not (significantly) more difficult than" ($\lhd$)
  - For this we need the concept of problem reductions.

Polynomial Reduction

- If problems $P, Q \in \mathcal{NP}$, and if an instance of P can be converted *in polynomial time* to an instance of Q, then P is *polynomially reducible* to Q.
  - This is the "not (substantially) more difficult than" relation that we want to use.
  - We will write this as $P \lhd Q$

The "Hard Problems"—Class $\mathcal{NPC}$

- We want to ask the question—What are the hardest problems in $\mathcal{NP}$?
  - We'll call this class of problems $\mathcal{NPC}$, "$\mathcal{NP}$-Complete".
- Using the definitions we have made, we would like to say that if $P \in \mathcal{NPC}$, then $Q \in \mathcal{NP} \Rightarrow Q \lhd P$
  - If $P \in \mathcal{NP}$ and we can convert in polynomial time *every* other problem $Q \in \mathcal{NP}$ to P, then P is in this sense the "hardest" problem in $\mathcal{NP}$. $P \in \mathcal{NPC}$
- Is it obvious that such problems exist?
  - **No!** – We'll come to this later...
- Thm: $Q \in \mathcal{P}, P \lhd Q \Rightarrow P \in \mathcal{P}$
- Thm: $P \in \mathcal{NPC}, P \lhd Q \Rightarrow Q \in \mathcal{NPC}$

$\mathcal{P} = \mathcal{NP}$?

- We've seen lots of problems in $\mathcal{P}$, and we've seen some problems (today) in $\mathcal{NP}$.
- We know that $\mathcal{P} \subseteq \mathcal{NP}$.
- Have we seen any problems in $\mathcal{NP} \setminus \mathcal{P}$?
  - Do such problems exist?
  - No one knows for sure!
- If you can answer this, you will one million dollars!
- `www.claymath.org/Millennium_Prize_Problems/P_vs_NP/`
- I will also give you an A++++++++++++ in the class if you write my name on the paper. :-)

The Satisfiability Problem

- This is the first problem to be shown to be NP-complete.
- The problem is described by
  - a finite set $N = \{1, \ldots, n\}$ (the *literals*), and
  - m pairs of subsets of N, $C_i = (C_i^+, C_i^-)$ (the *clauses*).
- An instance is feasible if the set

$$\left\{ x \in \mathbb{B}^n \mid \sum_{j \in C_i^+} x_j + \sum_{j \in C_i^-} (1 - x_j) \geq 1 \; \forall i = 1, \ldots, m \right\}$$

  is nonempty.
- This problem is in $\mathcal{NP}$. **Why?**
- In 1971, Cook defined the class $\mathcal{NP}$ and showed that satisfiability was NP-complete.

Proving $\mathcal{NP}$-completeness

- Once we know that satisfiability is $\mathcal{NP}$-complete, we can use this to prove other problems are $\mathcal{NP}$-complete using the "reduction theorem":
  - $P \in \mathcal{NPC}, P \lhd Q \Rightarrow Q \in \mathcal{NPC}$

How to Win \$1M

- Here's a hint
- Thm: If $P \cap \mathcal{NPC} \neq \emptyset \Rightarrow \mathcal{P} = \mathcal{NP}$
  - Proof: Let $Q \in \mathcal{P} \cap \mathcal{NPC}$ and take $R \in \mathcal{NP}$.
  - $R \lhd Q$
  - $Q \in \mathcal{P}, R \lhd Q \Rightarrow R \in \mathcal{P}$
  - $\mathcal{NP} \subseteq \mathcal{P} \Rightarrow \mathcal{P} = \mathcal{NP}$

QUITE ENOUGH DONE

- To prove $\mathcal{P} = \mathcal{NP}$, you only need to find a polynomial algorithm for any problem that has shown to be $\mathcal{NP}$-complete
  - How good are you at Minesweeper? :-)
    - `http://web.mat.bham.ac.uk/R.W.Kaye/minesw/ordmsw.htm`

Theory versus Practice

- In practice, it is true that most problem known to be in $\mathcal{P}$ are "easy" to solve.
- This is because most known polynomial time algorithms are of relatively low order.
- It seems very unlikely that $\mathcal{P} = \mathcal{NP}$
- Although all NP-complete problems are "equivalent" in theory, they are not in practice.
- TSP vs. QAP
  - TSP—Solved instances of size ≈ 25000
  - QAP—Solved instances of size ≈ 30

Some "Easy" problems—Class $\mathcal{P}$

- $\mathcal{P}$ is the class of problems for which there exists a polynomial algorithm.
- $\mathcal{P} \in \mathcal{NP} \cap$ co-$\mathcal{NP}$: Why?
- Some problems in $\mathcal{P}$

Matching

- Given: Graph $G = (V, E), k \in \mathbb{Z}$
- Question: Does $\exists$ a matching M in G with $|M| \geq k$. A matching is a subset of edges such that no two edges share a common endpoint). More mathy: $(i, j) \in M \Rightarrow (i, k) \notin M \ \forall k \neq j$.

More Problems in $\mathcal{P}$

LP

- Given: $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$.
- Question: Does $\exists x \in \mathbb{R}^n_+$ such that $Ax \leq b, c^T x \geq K$?

Assignment Problem

- Given: set $N = \{1, 2, \ldots, n\}$, costs $c_{ij} \in \mathbb{Z}_+ \forall (i, j) \in (N \times N)$
- Question: Does $\exists$ a permutation Π of N such that $\sum_{i \in N} c_{i\pi(i)} \geq Q$

More Problems in $\mathcal{P}$

Longest Path in a DAG

- Given: A directed acyclic graph $G = (N, A)$, lengths $\ell_a \in \mathbb{Z} \ \forall a \in A$
- Question: Does $\exists$ a path P in G such that $\sum_{a \in P} \ell_a \geq K$?

The Line Between $\mathcal{P}$ and $\mathcal{NPC}$

The line between these two classes is very thin!

- Shortest Path (with non-negative edge weights) is in $\mathcal{P}$.
- Longest Path (with non-negative edge weights) is in $\mathcal{NPC}$

- A graph $G = (V, E)$ is Hamiltonian if and only if there is a walk in G that traverses each vertex $v \in V$ exactly once
- A graph $G = (V, E)$ is Eulerian if and only if there is is a walk in G that traverses each edge $e \in E$ exactly once

Thin Line

Example: Hamiltonian Circuit

Instance: Graph $G = (V, E)$
Question: Does G contain a Hamiltonian Circuit?

Example: Eulerian Circuit

Instance: Graph $G = (V, E)$
Question: Does G contain a Eulerian Circuit?

- Hamiltonian Circuit $\in \mathcal{NPC}$
- Eulerian Circuit $\in \mathcal{P}$

Weird Stuff

Chinese Postman

- Given: Undirected graph $G = (V, E), w_e \in \mathbb{Z}_+ \forall e \in E, B \in \mathbb{Z}$
- Question: Does $\exists$ a cycle in G traversing each edge at least once whose total weight is $\leq B$?

Chinese Postman $\in \mathcal{P}$

Weird Stuff

Directed Chinese Postman

- Given: Directed Graph $G = (N, A), w_a \in \mathbb{Z}_+ \forall a \in A, B \in \mathbb{Z}$
- Question: Does $\exists$ a cycle in G traversing each arc at least once whose total weight is $\leq B$?

Directed Chinese Postman $\in \mathcal{P}$

Mixed Chinese Postman

- Given: Mixed Graph $G = (V, A \cup E), w_e \in \mathbb{Z}_+ \forall e \in (A \cup E), B \in \mathbb{Z}$
- Question: Does $\exists$ a cycle in G traversing each edge and each arc at least once whose total weight is $\leq B$?

Mixed Chinese Postman $\in \mathcal{NPC}$

That Thin, Thin Line

- Consider a 0-1 matrix A an integer k defining the decision problem

$$\exists \left\{x \in \mathbb{B}^n \mid Ax \leq e, e^T x \geq k\right\}?$$

- If we limit the number of nonzero entries in each column to 2, then this problem is known to be in $\mathcal{P}$.
  - What is this problem?
- If we allow the number of nonzero entries in each column to be 3, then this problem is $\mathcal{NP}$-complete!
- If we allow at most one '1' per row, the problem is in $\mathcal{P}$
- If we allow two '1's per row, it is in $\mathcal{NPC}$

Next Time!

- Begin Review Sessions