Last lecture in complexity theory

The class #P

A function $f : 0, 1^* \to \mathbb{N}$ belongs to the class #P if

- exists polynomial p and a poly-time DTM M, such that
- $f(x) = |\{y \in \{0,1\}^{p(x)} \mid M(x,y) = 1\}|$.

In other words (a bit informally),

- Let $L \in$ NP. Let M be a poly-time DTM that checks certificates for L.
- If $f(x) =$ "number of certificates for x" then $f \in$ #P...
- and *vice versa*.

Generalizing $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ question

- FP — the class of all functions $f : 0, 1^* \to \mathbb{N}$ computable in poly-time.
- **Theorem.** If FP $=$ #P then P $=$ NP.

$\#$P **might be "richer" than** NP

- The problem CYCLE — given a directed graph. Does it contain a simple cycle?
  - CYCLE $\in$ P.
- The problem $\#$CYCLE — given a directed graph. How many simple cycles does it contain?
  - $\#$CYCLE $\in$ $\#$P.
- **Theorem.** If $\#$CYCLE $\in$ FP then P $=$ NP.
- **Proof.** In a graph G with n vertices, replace each edge with a gadget that turns a cycle of length m into $2^{m(n \log n+1)}$ cycles. Then Hamiltonian graphs have more cycles than others.

#P-completeness

- $f : 0, 1^* \to \mathbb{N}$ is #P-complete, if $f \in$ #P and #P $=$ FP^f.
- The problem #SAT — how many satisfying valuations does a propositional formula have?
- **Theorem.** #SAT is #P-complete.
- **Proof.** Our reduction from $\langle M, x, 1^n \rangle$ (where M is NTM) to SAT preserved the computation paths / certificates.
- #3-CNFSAT is also #P-complete.

Perfect matching in bipartite graphs

- A matching in a graph $G = (V, E)$ is a set of edges $M \subseteq E$, such that each $v \in V$ is incident to at most one edge in M.
- A matching is perfect if $|M| = |V|/2$. "Does G have a perfect matching?" is in P.
- Let $|X| = |Y| = n$. Consider a bipartite graph $G = (X \cup Y, E)$ with $E \subseteq X \times Y$.
- Consider a $n \times n$ matrix $A = (a_{ij})$, where $a_{ij} = 1$ iff $(x_i, y_j) \in E$, and $a_{ij} = 0$ otherwise.

Permanent of a matrix

The permanent of $n \times n$ matrix $A = (a_{ij})$ is

$$\mathrm{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

S_n — set of all permutations of $\{1, \ldots, n\}$.

- If A is obtained from a bipartite graph, then $\mathrm{perm}(A)$ counts perfect matchings.
- **Theorem.** perm is $\#$P-complete (even for 0,1-matrices).

Proof

- $\text{perm}(A)$ as the number of cycle covers in a n-node digraph.
  - Cycle cover — a set of cycles, such that each node is on exactly one cycle.
- First we consider general matrices (entries from $\mathbb{Z}$).
  - The entries of the matrix appear as weights of edges.
  - weight of a cycle cover — the product of edges in it.
- Let φ be a 3-CNF formula with clauses $C_1, \ldots, C_m$ and variables $x_1, \ldots, x_n$.
- See the gadgets in Arora&Barak's book (Fig 9.3)...

Going to 0,1-matrices

- Replace an edge $u \xrightarrow{(-)k} v$ with k new vertices $w_1, \ldots, w_k$, edges $u \xrightarrow{(-)1} w_i \xrightarrow{1} v$ and self-loops by w_i.

- Gives us graph with edge weights in $\{1, -1\}$. Its permanent is in $[-n!, n!]$ where n is the number of vertices.

- Compute the permanent *modulo* $2^m + 1$, where $2^m \geq 2n!$. Then $-1 \equiv 2^m$.

- Edge $u \xrightarrow{2^m} v$ may be replaced with $u \xrightarrow{2} w_1 \xrightarrow{2} \cdots \xrightarrow{2} w_{m-1} \xrightarrow{2} v$.

Optimization problems

- An optimization problem consists of a
  - Relation $\rho \subseteq \{0,1\}^* \times \{0,1\}^*$.
    - $x \; \rho \; y$ means that y is a feasible solution to the problem x.
  - cost function $c : \{0,1\}^* \to \mathbb{N}$.
  - (direction of optimization)
- If ρ, c are poly-time, then $\langle x, \rho, c, K \rangle \in$ NP
  - Does there exist a feasible solution to the problem x of cost at most/least K?

Examples

- Traveling salesman, independent set, vertex cover, knapsack,...

- Maximum satisfiability (MAXSAT) — given Boolean formulas $\varphi_1, \ldots, \varphi_m$ with the same variables $x_1, \ldots, x_n$. Find a valuation of these variables that satisfies as many of $\varphi_1, \ldots, \varphi_m$ as possible.

  - k-MAXSAT — each of the formulas φ_i depends on at most k variables.

- ...

Approximation algorithms

Let $\langle \rho, c \rangle$ be an optimization problem.

- For any $x \in \{0,1\}^*$, let $opt(x)$ be the cost of optimal feasible solutions to x.
- (Det.) Algorithm M is an ε-approximation algorithm for $\langle \rho, c \rangle$, if for all (sufficiently large) $x \in \{0,1\}^*$:
  - $x \ \rho \ M(x)$.
  - $|c(M(x)) - opt(x)| / \max\{c(M(x)), opt(x)\} \leq \varepsilon$
- A 0-approximation algorithm finds an optimal feasible solution.
- A 1-approximation algorithm finds any feasible solution.

An optimization problem can be ε-approximated if it has a poly-time ε-approximation algorithm.

Approximability

- Vertex cover can be $1/2$-approximated.
- k-MAXSAT can be $(1 - 2^{-k})$-approximated.
  - If all φ_i are disjunctions of variables then the problem can be $1/2$-approximated.
  - If all φ_i are disjunctions with exactly k different variables, then the problem can be $1/k$-approximated.
- If traveling salesman can be ε-approximated with $\varepsilon < 1$, then P $=$ NP.
- Traveling salesman with triangle inequality is still NP-complete. It can be $1/3$-approximated.
- Knapsack can be ε-approximated for any $\varepsilon > 0$.

Various complexity classes

- NPO — all optimization problems with poly-time ρ and c.
- APX — ε-approximable problems, where $\varepsilon < 1$.
- PTAS — ε-approximable problems, for any $\varepsilon > 0$.
  - The mapping $\varepsilon \to M_\varepsilon$ must be poly-time, where M_ε is the ε-approximation algorithm.
- FPTAS — there exists an algorithm $M(x, \varepsilon)$, such that
  - $M(\cdot, \varepsilon)$ is a ε-approximation algorithm.
  - Running time of $M(x, \varepsilon)$ is bounded by $q(|x|, 1/\varepsilon)$ for some polynomial q.

FPTAS $\subseteq$ PTAS $\subseteq$ APX $\subseteq$ NPO. All inclusions are strict unless P $=$ NP.

Separations (if $\mathsf{P} \neq \mathsf{NP}$)

- APX $\subsetneq$ NPO because travelling salesman is non-approximable.
- There is no fptas for strongly NP-complete problems.
  - In problem statement, numeric values are polynomial in problem size.
  - BIN PACKING separates PTAS and FPTAS.
- There are artificial problems that separate PTAS and APX.
  - Problem — propositional formula φ. Feasible solution — any valuation of variables. $c(x_1, \ldots, x_n) = 1$ iff $\varphi(x_1, \ldots, x_n) =$ true, otherwise $c(x_1, \ldots, x_n) = 2$.