Complexity Theory

Jan Křetínský

Chair for Foundations of Software Reliability
and Theoretical Computer Science
Technical University of Munich

Summer 2016

July 11, 2016

Lecture 25

Counting

Agenda

- examples of counting problems
- definition
- how hard are they?

Examples

Deciding is easy, counting is hard

Example (#CYCLE)

Number of simple cycles

- cycle detection in linear time
- if $\#CYCLE$ has a polynomial algorithm then **P** $=$ **NP**

Examples

Deciding is easy, counting is hard

Example (#CYCLE)

Number of simple cycles

- cycle detection in linear time
- if $\#\mathit{CYCLE}$ has a polynomial algorithm then $\mathbf{P} = \mathbf{NP}$

Example (GraphReliability)

$\frac{1}{2^n}\cdot$ number of subgraphs with a path from s to t

Example (Maximum likelyhood in Bayes nets)

Visible variables are $\vee$'s of ≤ 3 hidden variables.
What is the fraction of satisfying assignments with $x_1 = 1$?

- equivalent to $\#\mathit{SAT}$

Definition

Definition (#P)

A function $f : \{0,1\}^* \to \mathbb{N}$ is in **#P** if there is a polynomial-time TM M and a polynomial p such that $\forall x \in \{0,1\}^*$

$$f(x) = \left|\left\{y \in \{0,1\}^{p(|x|)} : M(x,y) = 1\right\}\right|$$

- counting certificates
- or accepting paths

Definition (FP)

A function $f : \{0,1\}^* \to \mathbb{N}$ is in **FP** if there is a deterministic polynomial-time TM computing f.

- efficeintly solvable counting

Decision analog

Theorem

$\mathbf{FP} = \#\mathbf{P}$

Decision analog

Theorem

FP $=$ **#P** $\iff$

Decision analog

Theorem

$\mathbf{FP} = \#\mathbf{P} \iff \mathbf{P} = \mathbf{PP}$

Completeness

Definition

A function f is $\#\mathbf{P}$-complete if $f \in \#\mathbf{P}$ and for every $g \in \#\mathbf{P}$ we have $g \in \mathbf{FP}^f$

- $\#SAT$ is $\#\mathbf{P}$-complete

Completeness

Definition

A function f is $\#\mathbf{P}$-complete if $f \in \#\mathbf{P}$ and for every $g \in \#\mathbf{P}$ we have $g \in \mathbf{FP}^f$

- $\#SAT$ is $\#\mathbf{P}$-complete

Example (Determinant)

$det(A) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i=1}^{n} A_{i,\sigma(i)}$

- computable in polynomial time

Example (Permanent)

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

- $\#\mathbf{P}$-complete (for 0,1 matrices) [Valiant'79]
- hence $perm \in \mathbf{FP} \implies \mathbf{P} = \mathbf{NP}$

Toda's theorem

Theorem (Toda'91)

$\mathbf{PH} \subseteq \mathbf{P}^{\#SAT}$

Proof idea

- randomized reduction from **PH** to $\oplus SAT$
  (odd number of satisfying assignments; $\oplus$**P**-complete problem)
- derandomization

What have we learnt?

- counting seems harder than deciding
- #**P**-complete problems arise from **NP**-complete problems as well as from those in **P**
- more powerful than alternating quantifiers
- classes **PP** and $\oplus$**P**: most and least significant bits of #**P** function