Lecture 2

- Counting Classe and $\sharp\mathbf{P}$
- Algebraic complexity:
  the classes $\mathbf{VP}_{\mathfrak{k}}$ and $\mathbf{VNP}_{\mathfrak{k}}$.
- Blum-Shub-Smale: The BSS Model

Counting Problems

Let $M(\bar{x}, \bar{y})$ be a non-deterministic polynomial time Turing machine which takes $\bar{y}$ as input and $\bar{x} \in \{0, 1\}^{\ell(\bar{y})}$ as guess.

We define a function f_M by

$$f_M(\bar{y}) = \mid \{\bar{x} : M(\bar{x}, \bar{y}) = 1\} \mid$$

f_M has values in $\mathbb{N}$.

The class of such functions f_M for non-deterministic polynomial time Turing machines M is denoted by $\sharp\mathbf{P}$.

With f_M we associate the decision problem

$$\bar{y} \in B_M \text{ iff } f_M(\bar{y}) \neq 0$$

This shows that $\mathbf{NP} \subseteq \mathbf{P}^{\sharp\mathbf{P}}$, and hence, if $\mathbf{P} = \mathbf{P}^{\sharp\mathbf{P}}$, then $\mathbf{NP} = \mathbf{P}$.

The Hamiltonian

Given an $(n \times n)$- matrix $M = (m_{i,j})$ over a field $\mathfrak{k}$, the *hamiltonian* of M is defined by

$$ham(M) = \sum_{\pi \in C_n} \prod_{i=1}^{n} m_{i,\pi(i)}$$

Here C_n is the set of cyclic permutations of n elements.

Note the similarity to the determinant and permanent !

On $(0,1)$ matrices (graphs) $ham(M)$ counts the number of hamiltonian paths.

$***$

What is the complexity of $ham(M)$

$\sharp\mathbf{P}$-hard problems

We denote the class of discrete functions computable in polynomial time by a deterministic Turing machine by $\mathbf{FP}$.

For a function f (class of functions C) we denote by

$$\mathbf{FP}^f, \text{ resp. } \mathbf{FP}^C$$

the class of functions computable by a deterministic polynomial time Oracle Turing Machine (OTM) which uses f (C) as oracle.

g is **polynomial time Turing reducible to** f if $g \in \mathbf{FP}^f$. We denote this by $g \leq_{Turing} f$,

f is **$\sharp\mathbf{P}$-hard** if $\sharp\mathbf{P} \subseteq \mathbf{FP}^f$.

Examples in $\sharp\mathbf{P}$

$\sharp$-SAT: For a set of clauses Σ, the number of satisfying assignments.

$\sharp$-p.matchings: For undirected graphs G, the number of perfect matchings.

$\sharp$-hamiltonian: For undirected graphs G, the number of hamiltonian cycles (paths).

$\sharp$-reliability: For a graph G and a subgraph H and a node $v \notin H$ the number of paths from v to some node in H.

Parsimonious reductions

SAT is the set of satisfiable formulas Σ in CNF (sets of clauses). An instance of SAT is given by Σ together with a satisfying assignment z.

HAM is the set of graphs with a hamiltonian cycle. An instance of HAM is given by $G = \langle V, E \rangle$ together with a hamiltonian cycle $E_0 \subseteq E$.

A **parsimonious reductions** is a polynomial time computable one-one map α between instances of a problem, say SAT, into instances of another problem, say HAM, such that for two instances Σ, z_1 and Σ, z_2 we have
$\alpha(\Sigma, z_1) = (G, E_{0,1})$ and $\alpha(\Sigma, z_2) = (G, E_{0,2})$.

Theorem:(Cook) SAT is $\mathbf{NP}$-complete for parsimonious reductions.

$\sharp\mathbf{P}$-complete problems

Theorem: If $Prob$ is an $\mathbf{NP}$-complete problem with respect to parsimonious reductions, then $\sharp Prob$ is $\sharp\mathbf{P}$-complete.

Corollary: $\sharp SAT$ and $\sharp HAM$ are $\sharp\mathbf{P}$-complete.

$\sharp$-p.matchings and $\sharp$-reliability are not settled by the above theorem, because perfect matchings and connecting paths can be found in polynomial time resp, even in logarithmic space.

Nevertheless, we have

Theorem:(Valiant)
$\sharp$-p.matchings and $\sharp$-reliability are $\sharp\mathbf{P}$-complete.

Complexity of the permanent

Recall:
Let $G = \langle V, E \rangle$ be a graph and M_G be its (symmetric) adjacency matrix.

Theorem: The number of perfect matchings of G is given by

$$\sum_{E' \subseteq E} \prod_{(i,j) \in E'} 1 = per(M_G)$$

where E' ranges over all perfect matchings of G.

Corollary: Computing the permanent of $(0,1)$-matrices is $\sharp\mathbf{P}$-complete.

Complexity of the hamiltonian

Proposition: For $(0,1)$-matrices M the hamiltonian $ham(M)$ is in $\sharp\mathbf{P}$.

Proof: It counts the number of hamiltonian cycles,

Proposition: For $(0,1)$-matrices M the hamiltonian $ham(M)$ is $\sharp\mathbf{P}$-complete.

Proof: HAM is $\mathbf{NP}$-complete for parsimonious reductions.

$***$

What about integer-valued matrices ?

What about matrices over $\{0,1,-1\}$?

Algebraic Circuits

Let $\mathfrak{k}$ be a field and $X_1, \ldots, X_n, \ldots$
be indeterminates over $\mathfrak{k}$.

We look at straight-line programs (algebraic circuits) Γ using $+, -, *$, indeterminates and constants from $\mathfrak{k}$.

The **depth of** Γ is its longest path.
The **size of** Γ is the number of instructions.

Γ computes a finite set F of polynomials
(one for each intermediate results).

The **complexity** $L(F)$ **of** F is the minimal size of all Γ which compute F.

$\mathbf{VP}_{\mathfrak{k}}$

A function $t : \mathbb{N} \to \mathbb{N}$ is called **p-bounded** iff there exists $c > 0$ such that $t(n) \leq n^c + c$ for all $n \in \mathbb{N}$.

A sequence f_n of multivariate polynomials over $\mathfrak{k}$ is called a **p-family** (over $\mathfrak{k}$) iff the number of variables and the degree of f_n are p-bounded functions of n.

A p-family f_n is said to be **p-computable** iff the complexity $L(f_n)$ is a p-bounded function of n.

We denote by $\mathbf{VP}_{\mathfrak{k}}$ the class of all p-computable families over $\mathfrak{k}$.

Examples in $\mathbf{VP}_{\mathfrak{k}}$

- For two vectors $\bar{X}, \bar{Y}$ of length m put
$$SC_m(\bar{X}, \bar{Y}) = \sum_i X_i Y_i$$

- For an $(n \times n)$-matrix $M = (X_{i,j})$ put
$$DET_n(M) = \sum_\pi (-1)^{sign(\pi)} \prod_i X_{i,\pi(i)}$$

SC_m is a p-family which is p-computable.

DET_n is a p-family. Using division, and Gaussian elimination, it is p-computable. It is also p-computable without division, since

Theorem: (Strassen, 1973) Division can be avoided and the division-free complexity is p-bounded in the complexity with divisions, the degree and the number of variables.

$\mathbf{VNP}_{\mathfrak{k}}$

A family f_n is called **p-definable** iff there exists a p-computable family $g_n \in \mathfrak{k}[X_1, \ldots, X_{u(n)}]$ such that for all n

$$f_n(X_1, \ldots, X_{v(n)}) = \sum_{e \in \{0,1\}^{u(n)-v(n)}} g_n(X_1, \ldots, X_{v(n)}, e_{v(n)+1}, \ldots e_{u(n)})$$

The class of p-definable families form the complexity class $\mathbf{VNP}_{\mathfrak{k}}$.

Example: The characteristic polynomial $\chi(G)$ of graph G is defined by $\chi(G) = det(\lambda \cdot I - M_G)$. For a family of graphs G_n of size n the family $\chi(G_n)$ is p-definable without summation.

The Permanent, VI

Lemma: For a matrix $E \in \{0,1\}^{n\times n}$

$$\gamma_n(E) = \left(\prod_{i,j,l,m}(1 - e_{i,j}e_{l,m})\right) \cdot \left(\prod_i^n \sum_j^n e_{i,j}\right) \neq 0$$

(where $1 \leq i,j,l,m \leq n$ and $i = l$ iff $j \neq m$)
iff E is a permutation matrix.

Proposition: The permanent $PER_n = per(X_{i,j})$ is p-definable.

Proof: Put

$$g_n(\bar{X}, E) = \gamma_n(E) \cdot \left(\prod_i \sum_j X_{i,j}e_{i,j}\right)$$

and

$$PER_n = \sum_E g_n(\bar{X}, E)$$

Algebraic Reductions

A polynomial f is called a **projection** of a polynomial g, $f \leq g$, iff

$$f(X_1, \ldots X_n) = g(a_1, \ldots a_n)$$

for some $a_i \in k \cup \{X_1, \ldots, X_n\}$, f can be obtained from g by substitution of indeterminates and constants.

A **p-family** $f = (f_n)$ **is a p-projection of** $g = (g_n)$, $f \leq_p g$, iff there exists a function $t : \mathbb{N} \to \mathbb{N}$ which is p-bounded from above and below such that

$$\exists n_0 \forall n \geq n_0 : f_n \leq g_{t(n)}$$

A p-definable family g is $\mathbf{VNP}_{\mathfrak{k}}$**-complete** iff $f \leq_p g$ for all $f \in \mathbf{VNP}_{\mathfrak{k}}$.

The Permanent, VII

Theorem: (Valiant, 1979)
The family PER_n is $\mathbf{VNP}_{\mathfrak{k}}$-complete for fields $\mathfrak{k}$ of characteristic $\neq 2$.

The proof is rather involved.

Problems:

- What is the relationship between $\sharp\mathbf{P}$ and $\mathbf{VNP}_{\mathfrak{k}}$?

- What corresponds to $\sharp\mathbf{P}$ and $\mathbf{VNP}_{\mathfrak{k}}$ in the BSS-model of computation ?

- What rôle does the permanent $per(M)$ play in the BSS-model ?

Valiant's criterion for p-definability

Suppose $f : \{0,1\}^\star \to \mathbb{N}$ is a function in $\mathbf{FP}$ or even $\mathbf{P}/poly$.

It is open whether $\mathbf{NP} \subseteq \mathbf{P}/poly$, but if yes then $\mathbf{PH} = \Sigma_2^P$.

Proposition: (Valiant's criterion)
If $f \in \mathbf{P}/poly$, then

$$f_n = \sum_{e \in \{0,1\}^n} f(e) \cdot X_1^{e_1} \cdot \ldots \cdot X_n^{e_n}$$

is p-definable, and hence in $\mathbf{VNP}_{\mathfrak{k}}$ for every field $\mathfrak{k}$.

Corollary:

$$ham_n = ham(X) = \sum_{\pi \in C_n} \prod_{i=1}^{n} X_{i,\pi(i)} \in \mathbf{VNP}_{\mathfrak{k}}$$

Complexity of the hamiltonian (revisited)

Theorem:(Valiant) ham_n is $\mathbf{VNP}_{\mathfrak{k}}$-complete for every field $\mathfrak{k}$.

The proof is rather involved but follows the same strategy as the proof for $\sharp\mathbf{P}$.

We find the proof in Bürgisser's book, Theorem 3.18 and corollary 3.19.

There are many more such polynomials, including

- Cycle format polynomials
- Graph factors
- Connectivity polynomials

For many polynomials the problem is also open.

Non-uniform complexity

We are given a polynomial, say per_n with n_2 variables, and we want to compute $per_n(M)$ for matrices $M \in \mathbb{R}^{n \times n}$.

In the **non-uniform** approach we look, for each n, for a device $Machine_n$ which solves our problem, and ask for the device which uses the least resources.

For each n there is a device

$$Machine_n$$

which computes for every $M \in \mathbb{R}^{n \times n}$

$$per_n(M).$$

No relationship between the optimal device $Machine_n$ and $Machine_{n'}$ is required.

Uniform complexity

We are given a polynomial, say per_n with n_2 variables, and we want to compute $per_n(M)$ for matrices $M \in \mathbb{R}^{n \times n}$.

In the **uniform** approach we look for ONE device $Machine(n, M)$ which solves our problem, for each n and $M \in \mathbb{R}^{n \times n}$.

We look for the device which uses, **asymptotically** the least resources.

There is a device

$$Machine(n, M)$$

which computes for each n and

for every $M \in \mathbb{R}^{n \times n}$

$per_n(M)$.

$\sharp\mathbf{P}$ vs $\mathbf{VNP}_{\mathfrak{k}}$

$\sharp\mathbf{P}$:

- Uniform Turing Machine
- Input: strings in $\Sigma^\star$
- Output: $n \in \mathbb{N}$

$\mathbf{VNP}_{\mathfrak{k}}$:

- Non-uniform Algebraic Circuits
- Well defined over arbitrary rings $\mathfrak{R}$.

$BSS_{\mathfrak{R}}$

A $BSS_{\mathfrak{R}}$ program (machine) M over an arbitrary *ordered* ring $\mathfrak{R}$

- Consists of a set of *labeled (numbered) instructions*.
- Each instruction has (at most two) *input registers* and (at most) one *output register*.
- *Compute* instructions either *add* (+) or *subtract* (−) or *multiply* (∗) or *divide* (:).
- *Branch* instructions *test for equality* or *compare* and jump to *fixed* addresses.
- *Shift* instructions *increase* or *decrease* all the register numbers by 1.
- A finite set of *constants* is stored in specially chosen registers.

Polynomials in $\Re[X_1, X_2, \ldots, X_n, \ldots]$

We want to compute polynomials

$$p(X_1, X_2, \ldots, X_{v(n)})$$

with $v(n)$ variables, $m(n)$ monomials of degree at most $d(n)$.
$m(n) = m(d(n), v(n)) = m(d, v)$

For det_n, per_n and ham_n we have $v(n) \leq n^2$, $m(n) \leq n!$ and $d(n) \leq n$.
Evaluation may be exponential in n.

If $d(n) \leq d$ independently of n, evaluation is polynomial in n.

In BSS: $\mathbf{P}_{\mathfrak{R}}$, I

We want to compute polynomials

$$p(X_1, X_2, \ldots, X_{v(n)})$$

with $v(n)$ variables, $m(n)$ monomials of degree at most $d(n)$.

$det_n(M)$ is in $\mathbf{P}_{\mathfrak{R}}$.

$per_n(M)$ and $ham_n(M)$ are not known to be $\mathbf{P}_{\mathfrak{R}}$.

What corresponds to $\sharp\mathbf{P}$ in the BSS model ?

In BSS: $\mathbf{P}_{\mathfrak{R}}$, II

In general, we are interested in are functions

$$f : \mathfrak{R}^{\infty} = \bigcup_{n} \mathfrak{R}^{n} \to \mathfrak{R}$$

where the input size is given by the length of the vector $\bar{x} \in \mathfrak{R}^{\infty}$.

Problems are functions

$$f : \mathfrak{R}^{\infty} = \bigcup_{n} \mathfrak{R}^{n} \to \{0, 1\} \subseteq \mathfrak{R}$$

$\mathbf{P}_{\mathfrak{R}}$ is the set of problems computable in polynomially many computation steps (=executed instructions) measured in the input size.

$\mathbf{FP}_{\mathfrak{R}}$ is the set of functions computable in polynomially many computation steps.

In BSS: $\mathbf{NP}_{\mathfrak{R}}$, I

Guessing values:

Given a function

$$f(\bar{X}, \bar{Y}) : \mathfrak{R}^{\infty} \times \mathfrak{R}^{\infty} \to \{0, 1\}$$

where $\ell(\bar{X}) = \ell(\bar{Y})$,
we associate with it the problem

$$P_f = \{y \in \mathfrak{R}^{\infty} : \exists \bar{x} f(\bar{x}, \bar{y}) = 1\}$$

A problem P is in $\mathbf{NP}_{\mathfrak{R}}$ if it is of the form P_f for some $f \in \mathbf{FP}_{\mathfrak{R}}$.

In BSS: $\mathbf{NP}_{\mathfrak{R}}$, II

Examples:

- Given $M \in \mathfrak{R}^{n\times n}$ and $\bar{a} \in \mathfrak{R}^n$ is there $\bar{x} \in \mathfrak{R}^n$ such that $M \cdot \bar{x} = \bar{a}$?

  This is actually in $\mathbf{P}_{\mathfrak{R}}$ using the $det(M) = 0$ for the negative answer.

- Given a polynomial $p(\bar{x})$ of degree at most 3, does p have a zero in $\mathfrak{R}$?

  Exercise: Show that this can be done in polynomial time in the case of $\mathfrak{R} = \mathbb{R}$.

- Given a polynomial $p(\bar{x})$ of degree at most 4, does p have a zero in $\mathfrak{R}$?

  This is not known to be in $\mathbf{P}_{\mathfrak{R}}$.

  Problem: What is the complexity of this in Valiant's model? State this precisely.