1 A BPP problem

1.1 Definitions

Definition 1. *We recall that the BPP class can be defined as the class of languages such that there exists a polynomial P and a language A recognized in polynomial time by a Turing Machine such that :*

$$x \in L \Rightarrow \mathop{\mathbb{P}}_{\varepsilon \in \{0,1\}^{p(|x|)}}\Big((x,\varepsilon) \in A\Big) \geq \frac{2}{3} \qquad x \notin L \Rightarrow \mathop{\mathbb{P}}_{\varepsilon \in \{0,1\}^{p(|x|)}}\Big((x,\varepsilon) \in A\Big) \leq \frac{1}{3}$$

Definition 2. *We recall that the* degree of a monomial $X_1^{d_1} \dots X_n^{d_n}$ *is* $\sum_j d_j$ *and that the* degree of a polynomial *is the maximum of the degree of the monomial composing the polynomial. For instance* $XXXXY + XYZ + Z + ZY$ *has degree 5 (since* $XXXXY$ *has degree 5).*

1.2 Polynomial in general form

Definition 3. *A polynomial can always be written as the sum of monomials. A polynomial is in* general form *when it is given as a simple variable (i.e. X_i) or as sum, difference or product of polynomials in general form. For instance* $((X_1 + X_2) \times (X_1 - X_2) + X_1) \times X_1$.

We voluntarily ignore the details of the encoding of polynomials in general form. We will simply suppose that it is non-ambiguous, that the encoding size for the n-th variable on the input takes size $O(ln(n))$ and that it takes $O(1) + n_1 + n_2$ in size in the input for the sum or product of two polynomials of size n_1 and n_2. It is as if we use the string notation composed of (,), +, -, *, X, and the decimal digits to encode integers for the variables.

Question 1. Show that the function transforming a polynomial given in general form into a polynomial as sum of monomials cannot be computed in polynomial time.

Question 2. What is the complexity of the function testing the equality of two polynomials given in general form by using the expansion into a sum of monomials ?

1.3 Schwartz-Zippel lemma

Lemme 1. *Let $P \in \mathbb{Z}[X_1, \dots, X_n]$ of degree d and let S be a finite subset of $\mathbb{Z}$. If $x_1, \dots, x_n$ are chosen uniformly from S then* $\mathbb{P}\Big(P(x_1, \dots, x_n) = 0\Big) \leq \frac{d}{|S|}$

Question 3. Demonstrate or admit this lemma.

Question 4. Deduce a BPP algorithm for testing the equality of two polynomials in general form.

2 Rice theorem

Definition 4. *A* useless *state of a Turing machine is a state that is never visited during a computation.*

Question 5. Show that deciding whether a Turing machine has useless states is undecidable.

Question 6. Why cannot you simply apply Rice's theorem ?

3 Busy beavers

The *busy beavers* problem was introduced by Radó with the goal of defining a simple uncomputable function. The model of Turing Machines is the following :

— the machine is deterministic ;

— the tape is infinite in both direction ;
— the tape alphabet is $\{0, 1\}$;
— the machine possesses a unique "end" state with no outgoing transitions ;
— the tape is initially filled with 0 (we do not distinguish between 0 and the initial tape symbol).

The busy beaver function $\Sigma(n)$ is defined as the maximum number of 1 written on the tape (not necessarily consecutively) by a machine of $n + 1$ states (n states plus the final state) before the machine enters into a final state (the machine has to halt)

Question 7. Justify the existence of $\Sigma(n)$ for all $n \in \mathbb{N}$.

Question 8. What are $\Sigma(0)$? $\Sigma(1)$? $\Sigma(2)$? (or just show that $\Sigma(2) \geq 4$)

Question 9. Show that Σ is strictly increasing.

We say that a function f is computable when there exists a Turing Machine M such that M prints $f(n)$ consecutive 1 after reading n consecutive 1 on the input.

Question 10. Show that the function Σ increase is strictly faster that any computable function f (i.e. $f(n) = o(\Sigma(n))$).