Léonard Brice, Mathieu Sassolas
leonard.brice@ulb.be, mathieu.sassolas@ulb.be

Session 4:
Complexity

Reminders

The *complexity* of an algorithm corresponds to the rate at which the number of operations required to execute it, *in the worst case*, increases, as a function of the *size of the input* (that is, the number of letters, or bits, that are used to describe it). Thus, an algorithm that requires, for an input of size n, sometimes only 1 operation, and sometimes n^2 operations, is a quadratic algorithm. It is therefore more costly than a linear algorithm, but less than an exponential one.

To focus on the asymptotic behaviour of that number of operations, we use the notation O. Given two mappings $f, g : \mathbb{N} \to \mathbb{N}$, we write $f(n) = O(g(n))$ when there exist $n_0, c \in \mathbb{N}$ such that for every $n \geq n_0$, we have $f(n) \leq cg(n)$. Note that the notation O gives only an upper bound on how fast the mapping f grows: if $f(n) = O(n^2)$, then it is also true that $f(n) = O(n^3)$. In other words, an algorithm that runs in time $O(n^2)$ does also run in time $O(n^3)$.

The complexity class P gathers all the problems that can be solved by an algorithm that runs in polynomial time, i.e. an algorithm such that there exists $k \in \mathbb{N}$, such that inputs of size n are processed with $O(n^k)$ operations.

The complexity class EXP gathers all the problems that can be solved by an algorithm that runs in exponential time, i.e. an algorithm such that there exists $k \in \mathbb{N}$, such that inputs of size n are processed with $O(2^{n^k})$ operations.

1 Complexity

1. Prove that $3n^3 - 2n^2 + n - 10 \in O(n^3)$.
2. Sort by increasing asymptotic order the following functions over $\mathbb{N}$:
   (a) $n \mapsto n^2 + 10$
   (b) $n \mapsto \log(n)$
   (c) $n \mapsto 2^n$
   (d) $n \mapsto \frac{n^2}{4}$
   (e) $n \mapsto n - 1$
   (f) $n \mapsto n \log(n)$
   (g) $n \mapsto n^n$
   (h) $n \mapsto n^3 - \frac{n^2}{4}$
   (i) $n \mapsto n^{\log(n)}$

2 Algorithms

For each of the following problems, describe an algorithm that solves it and bound its execution time:

1. Given a list ℓ of integers and a number k, is every element of ℓ smaller than k?
2. Given an undirected graph (as an adjacency matrix) and an initial vertex v, can every vertex be reached from v?
3. Given a Boolean formula φ, and a valuation ν, does ν satisfy φ?

4. SAT: Given a Boolean formula φ in CNF, does there exists a valuation ν that satisfies it?
5. Given a number $p \in \{0, \ldots, 2^{64} - 1\}$, written in binary, is p a prime number?
6. Given $p \in \mathbb{N}$, written in binary, is p a prime number?

3 Complexity classes

1. Among the classes P and EXP, which one is included in the other?
2. Among the algorithms you give as answers to Exercise 2, which ones are polynomial? Which of the problems presented in that exercise belong to P?
3. Computer scientists usually consider, as a first approximation, that a problem is "easy" if it belongs to P, and "hard" if it does not. In what extent would you say it is true?