Advanced Complexity

TD n°1 : SPACE and NL

September 13, 2017

Exercise 1 : Graph representation and why it does not matter

Let $\Sigma = \{0, 1, ;, \bullet\}$, $n \in \mathbb{N}$ and $V = [0, n-1]$. We consider the following two representations of a directed graph $G = (V, E)$ by a word in Σ^* :

— By its adjency matrix : $m_{0,0}m_{0,1} \ \ldots m_{0,n-1} \bullet \cdots \bullet m_{n-1,0} \ldots m_{n-1,n-1}$, where for all $i, j \in [0, n-1]$, $m_{i,j}$ is equal to 1 if $(i, j) \in E$, 0 otherwise.

— By its adjency list : $k^0_0; \ldots ; k^0_{m_1} \bullet \cdots \bullet k^{n-1}_0; \ldots ; k^{n-1}_{m_{n-1}}$, where for all i, $[k^i_0, \ldots , k^i_{m_i}]$ is the list of neighbors of vertex i, written in binary, in increasing order.

1. Describe a logarithmic space bounded deterministic Turing machine which takes as input the graph G, represented by adjacency lists, and returns the adjacency matrix representation of G.
2. Conversely, describe a logarithmic space bounded deterministic Turing machine taking as input a graph G, represented by its adjacency matrix , and computing the adjacency list representation of G.

Therefore, the complexity of the problem `REACH` seen in class does not depend on the representation of the graph.

Exercise 2 : Inclusions of complexity classes

Definition 1 *A function $f : \mathbb{N} \to \mathbb{N}$ is said to be* space-constructible *if there exists a deterministic Turing machine that computes $f(|x|)$ in $O(f(|x|))$ space given x as input.*

Show that for a space-constructible function,

$$\mathsf{NSPACE}(f(n)) \subseteq \mathsf{DTIME}(2^{O(f(n))})$$

Exercise 3 : Restrictions in the definition of $\mathsf{SPACE}(f(n))$, and why they do not matter

In the course, we restricted our attention to Turing machines that always halt, and whose computations are space-bounded on every input. In particular, remember that $\mathsf{SPACE}(f(n))$ is defined as the class of languages L for which there exists some deterministic Turing machine M that always halts (i.e. on every input), whose computations are $f(n)$ space-bounded (on every input), such that M decides L.

Now, consider the following two classes of languages :

— $\mathsf{SPACE}'(f(n))$ is the class of languages L such that there exists a deterministic Turing machine M, running in space bounded by $f(n)$, such that M accepts x iff $x \in L$. Note that if $x \notin L$, M may not terminate.

— $\mathsf{SPACE}''(f(n))$ is the class of languages L such that there exists a deterministic Turing machine M such that M accepts x using space bounded by $f(n)$ iff $x \in L$ (M may use more space and not even halt when $x \notin L$).

1. Show that for a space-constructible function $f = \Omega(logn)$, $\mathsf{SPACE}'(f(n)) = \mathsf{SPACE}(f(n))$
2. Show that for a space-constructible function $f = \Omega(logn)$, $\mathsf{SPACE}''(f(n)) = \mathsf{SPACE}(f(n))$

Exercise 4 : Dyck's language

Let A be the language of balanced parentheses – that is the language generated by the grammar $S \rightarrow (S)|SS|\epsilon$. Show that $A \in L$.

— What about the language B of balanced parentheses of two types ? that is the language generated by the grammar $S \rightarrow (S)|[S]|SS|\epsilon$

Exercise 5 : NL alternative definition

A Turing machine with *certificate tape*, called a verifier, is a deterministic Turing machine with an extra read-only input tape called *the certificate tape*, which moreover is *read once* (*i.e.* the head on that tape can either remain on the same cell or move right, but never move left).

Define NL_{certif} to be the class of languages L such that there exists a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ and a Turing machine with certificate tape M that runs in logarithmic space such that :

$$x \in L \text{ iff } \exists u, |u| \leq p(|x|) \text{ and } M \text{ accepts on input } (x, u)$$

1. Show that $\mathsf{NL}_{certif} = \mathsf{NL}$
2. What complexity class do you obtain if you remove the read-once constraint in the definition of a machine with certification tape ?