README Ce fichier n'est qu'un brouillon de réécriture de l'article de Grandjean et Schwentick *Machine-independent characterizations and complete problems for deterministic linear time* pour voir si ces notions peuvent s'étendre au temps $\mathcal{O}\left(n^k\right)$.

Notations Par abus de notation découlant de la théorie des ensembles, j'écrirai n pour $[\![0, n-1]\!]$ voire pour $[\![1, n]\!]$ quand il n'y aura pas d'ambiguïté. De manière générale, si un terme ressemble à un entier naturel mais qu'il est mis à la place d'un ensemble, il faut le lire comme l'ensemble $[\![0, n-1]\!]$.

Preliminaries (p.198)

RAM data structures (p.198)

Définition 1 (RAM data structures)**.** Soit t un type, c'est-à-dire une signature fonctionnelle ne contenant que des symboles de constantes ou de fonctions unaires.

Une RAM-structure s de type t est un uplet constitué de :

- $n \in \mathbb{N}$ est la taille de la structure ;
- $C \in \mathbb{N}$ pour chaque symbole $C \in t$;
- $f : n \to \mathbb{N}$ pour chaque symbole $f \in t$.

On notera $s.n, s.C, s.f$ les composantes n, C, f de s (cette notation est à rapprocher de l'accès à un attribut ou à une fonction membre en programmation objet).

On dira que s est c-bornée pour $c \in \mathbb{N}$ lorsque $s.C, s.f(i) < cs.n$ pour tous $C, f \in t$ et $i \in n$.

Définition 2 (Fonction de RAM)**.** Soient t_1, t_2 des types.

Une (t_1, t_2)-fonction de RAM Γ est une fonction telle qu'il existe $c_1, c_2 \in \mathbb{N}$, tels que Γ envoie les structures c_1-bornées de type t_1 sur des structures c_2-bornées de type t_2.

On dit que Γ est polynomiale lorsque $\Gamma(s).n = \mathcal{O}\left((s.n)^k\right)$.

Machine RAM (p.200)

La machine RAM reste la même (heureusement). On va utiliser la $\{+\}$-RAM ou des versions un brin plus puissantes comme la $\{+, -, \times, \div k\}$-RAM pour un $k \in \mathbb{N}$ fixé.

Définition 3 (Temps polynomial)**.** On définit $\mathrm{DTIME}_{\mathrm{RAM}}\left(\mathcal{O}\left(n^k\right)\right)$ comme étant l'ensemble des fonctions calculables sur $\{+\}$-RAM en temps $\mathcal{O}\left(n^k\right)$, telles que le nombre de registres utilisés, la longueur des nombres manipulés (y compris les adresses de registres) soient bornés par $\mathcal{O}\left(n^k\right)$.

Réductions affines (p.202)

On laisse inchangées les notions de *transformations affines* (définition 2.3), de *réductions affines* (définition 2.5),de projections affines (définition 2.7). Les théorèmes et lemmes suivants ou intermédiaires sont aussi inchangés. Ils permettront de définir des réductions qui *restent* dans $\mathrm{DTIME}_{\mathrm{RAM}}\left(\mathcal{O}\left(n^k\right)\right)$ pour un k fixé.

Le framework algébrique (p.208)

Le LSRS en tant que tel n'a pas l'air collé à la définition du temps linéaire. On garde la définition pour le moment.

On doit refaire la définition 3.3.

Pour t un type, on note F_t l'ensemble de symboles de fonctions suivants : $\{1(-), n(-), id(-)\} \cup \{f_C | C \in t\} \cup \{f | f \in t\}$.

Remarque 1. *Soient t un type et S un LSRS pour $f_1, \ldots, f_k$ sur F_t. L'entrée d'un LSRS peut être vue comme étant une RAM-structure s de type t, qu'il* lit *en interprétant les symboles de F_t de la façon suivante :*

- $\forall f \in t_1 : f(i) = \begin{cases} s.f(i) & \text{si } i < s.n \\ 0 & \text{sinon} \end{cases}$
- $\forall C \in t_1 : f_C(i) = s.C$
- $1(i) = 1, n(i) = s.n, id(i) = i$

La sortie du LSRS peut aussi être vue comme une nouvelle structure, de type $\{f_1, \ldots, f_k\}$.

Définition 4 (RAM n^k-représentée)**.** Soient t_1, t_2 des types. Soit Γ une (t_1, t_2)-fonction de RAM.

Soit S un LSRS pour $f_1, \ldots, f_k$ sur F_{t_1}. On dit que Γ est n^k-représentée par S lorsqu'il existe un entier c et une projection affine P tels que, pour chaque structure s c-bornée, S définit des fonctions $f_1, \ldots, f_k : c(s.n)^k \to c(s.n)^k$ à partir des interprétations des fonctions de F_{t_1} suivantes :

Ce LSRS définit une structure s' de type $\{f_1, \ldots, f_k\}$.