Modèles de calcul sur les réels, résultats de comparaison

Emmanuel Hainry

Institut National
Polytechnique de Lorraine

Département de formation doctorale en informatique

École doctorale IAEM Lorraine

Modèles de calcul sur les réels résultats de comparaisons

THÈSE

présentée et soutenue publiquement le 7 décembre 2006

pour l'obtention du

Doctorat de l'Institut National Polytechnique de Lorraine

(spécialité informatique)

par

Emmanuel Hainry

Composition du jury

Rapporteurs :	Serge Grigorieff	Professeur, Université Paris 7
	Giuseppe Longo	Directeur de recherche, CNRS & École Normale Supérieure, Paris
Examinateurs :	Vincent Blondel	Professeur, Université catholique de Louvain, Belgique
	Olivier Bournez	Chargé de recherche INRIA, Nancy
	José Félix Costa	Professeur, Instituto Superior Técnico, Lisbonne, Portugal
	Jean-Paul Haton	Professeur, Université Henri Poincaré, Nancy
	Jean-Yves Marion	Professeur, École des Mines de Nancy

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Remerciements

De nombreuses personnes méritent des remerciements à divers titres parce qu'elles ont contribué à ce que je puisse réaliser ce travail. J'adresse donc des remerciements aux personnes suivantes.

Je voudrais tout d'abord remercier mon encadrant, Olivier Bournez. Ses conseils éclairés et éclairants m'ont guidé depuis mon stage de DÉA et tout au long de ma thèse. Son optimisme a aussi contribué à faire avancer le travail dans le bon sens.

Je remercie aussi Jean-Yves Marion, mon directeur de thèse qui a su être présent aux bons moments.

Je remercie Serge Grigorieff et Giuseppe Longo qui ont accepté d'être les rapporteurs de cette thèse.

J'adresse également mes remerciements à Vincent Blondel qui m'a fait l'honneur de présider mon jury de thèse.

Mes profonds remerciements vont aussi à José Félix Costa qui a accepté de faire partie du jury, et qui m'a fait découvrir de beaux paysages portugais et des points de vue intéressants sur le sujet. Mes remerciements vont aussi à Jean-Paul Haton, référent interne au LORIA de mon travail de thèse.

Je dois aussi remercier Daniel Graça dont le séjour à Nancy m'a permis de découvrir des expressions françaises étranges. Le travail avec lui a été agréable et fructueux. Je remercie également Manuel Campagnolo qui a aussi participé à ce travail.

Merci à Antoine, Florent, Germain, Loubna, Romain avec qui j'ai partagé mon bureau mais aussi des idées et des doutes.

Plus généralement, merci à tous les membres des équipes PROTHEO et CARTE pour m'avoir accueilli au sein de ces équipes. Je suis également reconnaissant au LORIA dans le cadre duquel j'ai pu effectuer ma thèse confortablement et sans soucis.

Merci aussi à l'équipe enseignante du département d'informatique de l'école des mines de Nancy.

Je voudrais enfin remercier ma famille et en particulier mes parents.

Table des matières

Introduction

> Hobbes: That's a tricky one. You have to use calculus and imaginary numbers for this.You know. Eleventeen, thirty-twelve and all those. It's a little confusing at first.
>
> *(*Calvin & Hobbes, *Bill Watterson)*

Les machines à calculer ont été inventées il y a bien longtemps : il existe des traces de l'existence de bouliers chinois dès le douzième siècle. Les premières calculatrices remontent quant à elles au dix-septième siècle : la Pascaline a été créée en 1642 par Blaise Pascal. Wilhelm Schickard avait lui conçu ce qu'il appelait une horloge calculante en 1623. L'histoire des machines à calculer passe ensuite au dix-neuvième siècle par les métiers Jacquard puis Charles Babbage et Ada Lovelace et leur machine analytique (qui n'a pas existé). Les ordinateurs (que les anglophones appellent *computers*, soit calculateurs) d'aujourd'hui descendent des travaux d'Alan Turing. Ces ordinateurs, et la théorie des machines de Turing donnent la sensation que le calcul se résume au calcul sur les entiers. En effet, ces machines prennent une entrée entière, lui font subir des modifications discrètes et donnent donc des résultats entiers. Les seules façons de modéliser le calcul infinitésimal semblent être en itérant des processus numériques. Pourtant, la Pascaline, comme la machine de Babbage, si elles manipulent effectivement des données numériques et renvoient des résultats numériques travaillent de façon continue. Les rouages de ces machines sont en effet analogiques : les valeurs sont transmises sous forme de rotation d'une roue ou d'un engrenage, ce qui est par essence un phénomène continu. Parmi les éléments qui les composent, on peut trouver des intégrateurs qui sont des dispositifs physiques capables de réaliser la primitive d'une fonction, ce qui implique nécessairement des éléments travaillant sur les réels et non sur un ensemble discret, ainsi que l'illustre la citation suivante d'Ada Lovelace :

> « Many persons who are not conversant with mathematical studies imagine that because the business of the engine is to give its results in numerical notation, the nature of its process must consequently be arithmetical rather than algebraic and analytical. This is an error. The engine can arrange and combine its numerical quantities exactly as if they were letters or any other general symbols; and, in fact, it might bring out its results in algebraic notation were provisions made accordingly. It might develop three sets of results simultaneously, viz. symbolic results... ; numerical results (its chief and primary object); and algebraic results in literal notation. »

La prédominance des ordinateurs numériques aujourd'hui aurait donc pu nous faire oublier que les calculateurs peuvent être analogiques, et qu'ils l'ont été : Lord Kelvin a imaginé en 1876 un analyseur différentiel, machine capable de résoudre des équations différentielles. En 1931, Vannevar Bush et son équipe réalisent au *Massachusetts Institute of Technology* un prototype de cette machine qui sera utilisé pour résoudre des problèmes de balistique ou des questions d'architecture aéronautique (qui font intervenir des équations différentielles d'écoulement des fluides entre autres pour les calculs de portance). Cette machine a ensuite été modélisée par Claude Shannon dans [Sha41] sous le nom de General Purpose Analog Computer. Ce modèle a été étendu et modifié par Lee Rubel [Rub93] qui a définie l'Extended Analog Computer et récemment des prototypes d'Extended Analog Computer ont été conçus par Jonathan Mills [Mil95]. Cette machine (qui peut se présenter sous la forme d'un périphérique pour micro-ordinateur) permet elle aussi de résoudre des problèmes d'équations différentielles. D'autres modèles de calcul et machine à calculer ont vu le jour, citons par exemple les fonctions $\mathbb{R}$-récursives de Cristopher Moore [Moo96], les systèmes dynamiques continus tels les systèmes à dérivée constante par morceaux [AM98] ou le calcul à l'aide du problème des n corps [Smi06], les réseaux de neurones [Sie99, SS94], le modèle BSS [BSS89, BCSS98] inspiré des machines RAM sur les réels, les modèles de calcul optiques et l'analyse récursive [Wei00]. Le modèle de l'analyse récursive est l'un des modèles de calcul sur les réels les plus acceptés. Il est doté d'une théorie de la complexité (étudiée dans [Ko91]), a été étendu à des espaces quelconques dans [Bra03]. [Orp97] présente un recueil des modèles de calcul réel et résultats sur ces modèles.

La prédominance des machines discrètes par rapport aux machines à fonctionnement continu est probablement responsable du manque d'unification qui existe entre les différents modèles de calcul sur les réels. En effet, ces différents modèles manipulent des fonctions ayant des caractéristiques communes, mais il existe peu de résultats liant différents modèles entre eux excepté des résultats négatifs exhibant des fonctions qui peuvent être calculées par un modèle mais pas par un autre. On sait par exemple qu'il existe des fonctions calculables dans le cadre de l'analyse récursive qui ne le sont pas par le modèle BSS et inversement des fonctions calculables pour le modèle BSS mais pas pour l'analyse récursive. Peut-être est-ce cependant l'apparente incomparabilité de ces modèles qui les a faits oublier au profit de la théorie bien unifiée des fonctions discrètes calculables : on sait grâce à la thèse de Church-Turing que tous les modèles « raisonnables » calculent les mêmes fonctions, à savoir les fonctions récursives. Certains des modèles de calcul sur les réels ont en outre la capacité de calculer plus que les machines de Turing [Sta01].

L'absence d'équivalences entre les différents modèles de calcul sur les réels pourrait être remise en cause : des travaux récents sur lesquels nous nous appuyons ont montré des liens entre certaines classes de fonctions $\mathbb{R}$-récursives et des fonctions issues de l'analyse récursive ([Cam01, CM01, CMC00b, CMC02]) ; il a été également montré [Gra04] que la fonction Γ d'Euler, que l'on savait calculable au sens de l'analyse récursive mais pas générable par un GPAC, pouvait être calculée par un GPAC si l'on permettait à celui-ci de ne pas calculer uniquement en temps réel. D'autres travaux, comme [Kaw05], exhibent des inclusions entre les classes de fonctions de l'analyse récursive et les fonctions $\mathbb{R}$-récursives. Il est également possible de comparer analyse récursive et modèle BSS :

[BH98] montre un modèle particulier de machines BSS, nommés machines RAM sur les réels réalistes qui capture les mêmes fonctions que l'analyse récursive. Dans cette thèse, nous montrerons des équivalences entre des sous-classes de fonctions $\mathbb{R}$-récursives et les fonctions élémentairement calculables et les fonctions récursivement calculables, qui sont des ensembles issus de l'analyse récursive. Ces résultats ont été publiés dans les articles [BH04, BH05a, BH05b, BH06]. Nous montrerons également une équivalence entre les fonctions GPAC-calculables au sens de [Gra04] et les fonctions récursivement calculables, résultat qui a été publié dans [BCGH06].

Dans le chapitre 1, nous présenterons rapidement des modèles de calcul discrets, hybrides ou continus. Nous rappellerons des résultats intéressants sur ces modèles, en particulier la thèse de Church-Turing et les propriétés des fonctions récursives. Nous discuterons également de la réalisabilité physique de certains des modèles continus, ce qui permet d'introduire une question fondamentale sur ces modèles : qu'est-ce qu'un modèle de calcul sur les réels raisonnable ?

Le chapitre 2 présentera plus en détail les trois modèles sur lesquels nous présenterons ultérieurement des résultats de comparaison : l'analyse récursive, les fonctions $\mathbb{R}$-récursives et le General Purpose Analog Computer. Nous présenterons en particulier des définitions simplifiées ou améliorées qui si elles peuvent faire perdre en généralité suffisent à notre discours et permettent dans certains cas de s'affranchir des problèmes physiques que présentent les définitions originelles.

Pour parvenir à présenter une algèbre de fonctions continues équivalente aux fonctions récursivement calculables, nous devons caractériser les extensions aux réels des fonctions discrètes calculables. C'est ce que nous faisons dans le chapitre 3. Nous y présentons en effet un opérateur de recherche de zéro « raisonnable », et une algèbre de fonctions définie à l'aide de cet opérateur dont les restrictions des fonctions aux entiers naturels sont exactement les fonctions discrètes récursives.

Dans le chapitre 4 : nous présenterons un opérateur de limite qui permettra de passer de notre caractérisation des fonctions discrètes récursives à une caractérisation des fonctions récursivement calculables au sens de l'analyse récursive. Nous montrerons ce résultat d'équivalence. Nous allons également étendre ce résultat pour obtenir des caractérisations des fonctions élémentairement calculables et des fonctions $\mathscr{E}_n$-calculables où les $\mathscr{E}_n$ sont les niveaux de la hiérarchie de Grzegorczyk en nous basant sur des caractérisations des fonctions discrètes élémentaires et des fonctions de la hiérarchie de Grzegorczyk dues à Manuel Campagnolo, José Félix Costa et Cristopher Moore [CMC00b, Cam01, CMC02].

Le chapitre 5 montre que les fonctions GPAC-calculables sont exactement les fonctions récursivement calculables. Comme on sait que les fonctions générées par GPAC ne contiennent pas toutes les fonctions récursivement calculables (par exemple la fonction Γ), nous adaptons la définition de ce qui est calculable par GPAC en permettant le calcul convergent, ce qui signifie que le GPAC peut ne pas calculer en temps réel mais renvoyer un résultat qui s'affine quand le temps croît.

Première partie

Prolégomènes

1 Des modèles de calcul discrets aux modèles continus

Die ganzen Zahlen hat der liebe Gott gemacht, alles andere ist Menschenwerk.

(Leopold Kronecker)

De nombreux modèles ont été créés pour donner des bases théoriques au fonctionnement des machines à calculer et ordinateurs. Nous allons dans ce chapitre présenter quelques uns de ces modèles de calcul. Tout d'abord, dans la section 1.1, des modèles de calcul sur les entiers simulant des calculs se déroulant en temps discret : la machine de Turing, les automates à deux compteurs, les automates cellulaires et un modèle indépendant des machines : les fonctions récursives. La section 1.2 présente des modèles dont le fonctionnement est encore discret, mais qui travaillent sur des structures continues. La section 1.3 présente des modèles dont le fonctionnement est totalement continu.

1.1 Modèles discrets

1.1.1 Machine de Turing

Présentons tout d'abord la machine de Turing. Le lecteur pourra se référer à [Sip97] pour une présentation plus complète. De façon imagée, une machine de Turing est constituée d'un ou plusieurs rubans infinis (sa mémoire), de têtes de lecture et écriture et d'un dispositif de contrôle. Une représentation schématique est donnée en figure 1.1. Au début d'un calcul la chaîne d'entrée est écrite sur un ruban, toutes les autres cases des rubans sont blanches. La tête de lecture lit un symbole sur le ruban, le transmet au dispositif de contrôle et celui ci commande une écriture et un déplacement de la tête. Le déplacement peut se faire d'une case vers la gauche ou vers la droite ; la tête peut aussi ne pas bouger. Le dispositif de contrôle peut également entrer dans un état particulier signalant la fin du calcul. À ce moment, le résultat du calcul peut être lu sur le ruban de sortie. Une machine de Turing peut ne pas s'arrêter sur certaines entrées, dans ce cas, elle est dite non-terminante.

On peut définir formellement une machine de Turing comme suit :

Définition 1.1 (Machine de Turing) *Une machine de Turing est un sextuplet*

$$(Q, \Sigma, \Gamma, \delta, q_0, q_{final}).$$

Q, Σ *et* Γ *sont des ensembles finis et*

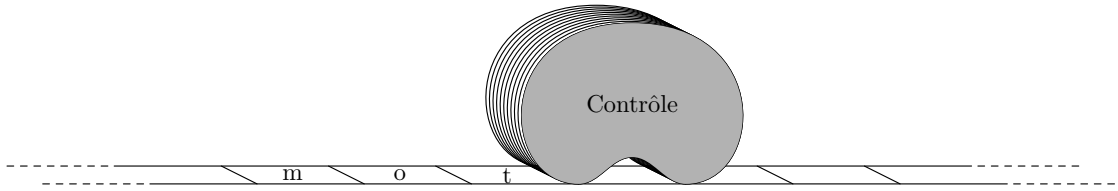

FIG. 1.1: Représentation d'une machine de Turing

- *Q est l'ensemble des états du dispositif de contrôle ;*
- *Σ est l'alphabet d'entrée. Il ne contient pas le symbole blanc $\_$;*
- *Γ est l'alphabet de travail. $\Sigma \cup \{\_\} \subseteq \Gamma$;*
- *δ est la fonction de transition, c'est à dire le mode de fonctionnement du dispositif de contrôle : $\delta : Q \times \Gamma \to Q \times \Gamma \times \{G, D, 0\}$. À l'état courant et au symbole lu, on associe le nouvel état, le symbole à écrire et le déplacement (gauche, droite ou néant) ;*
- *$q_0 \in Q$ est l'état initial ;*
- *$q_{final} \in Q$ est l'état final.*

Il est possible de définir une configuration d'une machine de Turing par un triplet contenant deux mots sur l'alphabet Γ et l'état de la machine. Rappelons que l'on note Γ^* l'ensemble des mots, c'est-à-dire des suites finies, sur l'alphabet Γ.

Définition 1.2 (Configuration) *La configuration d'une machine de Turing à un ruban est donnée de façon non ambiguë par le triplet de $\Gamma^* \times \Gamma^* \times Q$ formé du contenu du ruban à gauche de la tête de lecture, du contenu du ruban à droite et sous la tête de lecture et de l'état.*

Étant donnée une machine de Turing et sa configuration à un instant donné, il est possible de connaître la configuration suivante de la machine. La fonction qui associe à une configuration la configuration suivante est appelée fonction d'évolution de la machine.

Définition 1.3 (Fonction d'évolution) *Pour une machine de Turing M, on définit $f_M : \Gamma^* \times \Gamma^* \times Q \to \Gamma^* \times \Gamma^* \times Q$ qui à une configuration associe la configuration suivante de M.*

On peut donc maintenant définir les fonctions calculables par machine de Turing.

Définition 1.4 (Fonctions Turing-calculables) *Une fonction $f : \Sigma^* \to \Gamma^*$ est dite Turing calculable s'il existe une machine de Turing $(Q, \Sigma, \Gamma, \delta, q_0, q_{final})$ qui s'arrête sur l'entrée $s \in \Sigma^*$ si et seulement si $f(s)$ est définie, et dans ce cas, la sortie vaut $f(s)$.*

On appelle langage sur l'alphabet Γ un ensemble quelconque de mots de Γ^*. On peut se servir des machines de Turing comme reconnaisseurs de langages.

Définition 1.5 (Langage reconnu par machine de Turing) *Un langage est qualifié de décidable si une machine de Turing terminante répond "accept" pour tout mot du langage et "reject" pour tout mot hors langage.*

Un langage est qualifié de semi-décidable s'il existe une machine de Turing répondant "accept" si et seulement si l'entrée appartient au langage.

On dit parfois d'un langage décidable qu'il est récursif, et d'un langage semi-décidable qu'il est récursivement énumérable.

Exemple 1.6 *L'ensemble des nombres premiers est décidable.*

1.1.2 Automates à deux compteurs

Un automate à deux compteurs est constitué d'un système de contrôle et de deux piles unaires qui font office de mémoire [Min67].

Définition 1.7 (Automate à deux compteurs) *Un automate à deux compteurs est un quintuplet*

$$(Q, \Sigma, \delta, q_0, q_{final}).$$

- Q *est l'ensemble des états de l'automate ;*
- Σ *est l'alphabet du ruban ;*
- $\delta : Q \times \Sigma \times \{0,1\} \times \{0,1\} \to Q \times \{-1,0,+1\} \times \{-1,0,+1\}$ *est la fonction de transition : en fonction de la nullité ou non des deux compteurs, de l'état de l'automate et de la lettre lue, elle décide du nouvel état de l'automate, et du fait d'incrémenter ou décrémenter les compteurs ;*
- $q_0 \in Q$ *est l'état initial ;*
- $q_{final} \in Q$ *est l'état final de l'automate.*

L'entrée d'un calcul est le contenu originel d'un ou des compteurs. Le résultat d'un calcul par un automate à deux compteurs est le contenu des compteurs quand l'automate atteint l'état final.

Les automates à deux compteurs sont trivialement équivalents à un petit langage de programmation à étiquettes (les états) et agissant sur deux registres **A** et **B**. Les instructions possibles étant de la forme

- « **A++** »
- « **B++** »
- « **A−−** »
- « **B−−** »
- « **si A** $\neq$ **0 aller à** étiquette »
- « **si B** $\neq$ **0 aller à** étiquette »

Exemple 1.8 *La fonction qui à un nombre associe son double peut être calculée par un automate à deux compteurs :*

```
begin :
        si A ≠ 0 aller à calc
calc :
        B++
        B++
        A−−
        si A ≠ 0 aller à calc
```

1.1.3 Automates cellulaires

Un automate cellulaire est un modèle de calcul que l'on peut qualifier de parallèle. L'espace de travail est une grille (en général $\mathbb{Z}^n$), le processus de calcul suit un certain nombre de règles concernant une cellule et ses voisines : en chaque cellule est calculée simultanément la valeur de la cellule à l'instant suivant. Le calcul est considéré fini quand une cellule particulière, par exemple l'origine, prend une valeur prédéfinie.

On peut également définir une notion de fonction calculable par automate cellulaire en considérant que l'on place l'entrée dans des cellules contiguës en commençant à l'origine, que l'on laisse la machine travailler et que si l'état final est atteint, le résultat peut être lu sur la grille.

Exemple 1.9 *La fonction successeur est calculable par automate cellulaire.*

Plaçons-nous sur une grille linéaire, en binaire. On se donne l'alphabet

$$\Sigma = \{0, 1, \_, 0_f^+, 0_f, 1_f\},$$

où _ est le symbole blanc. On suppose que le nombre dont on veut calculer le successeur est écrit de gauche à droite : par exemple, pour le nombre 11,

_	1	0	1	1	_	_

.

Le symbole 0_f^+ indique qu'il y a une retenue à propager vers la gauche. Les indices f indiquent que le calcul pour ce chiffre a déjà été effectué. On se donne les règles suivantes (on représente par ? n'importe quel symbole) :

$$\begin{array}{llll}
? & 1 & _ & \to 0_f^+ \\
? & 0 & 1_f & \to 0_f \\
? & 1 & 0_f & \to 1_f \\
? & X & ? & \to X
\end{array}
\qquad
\begin{array}{llll}
? & 0 & 0_f^+ & \to 1_f \\
? & 1 & 0_f^+ & \to 0_f^+ \\
? & _ & 0_f^+ & \to 1_f
\end{array}
\qquad
\begin{array}{llll}
? & 0 & _ & \to 1_f \\
? & 1 & 1_f & \to 1_f \\
? & 0_f^+ & ? & \to 0_f
\end{array}$$

La dernière règle s'interprète comme « si aucune des autres règles ne s'applique, la cellule ne change pas.»

Appliquons ces règles au nombre 11 (le temps croît vers le bas) :

_	1	0	1	1	_	_
_	1	0	1	0_f^+	_	_
_	1	0	0_f^+	0_f	_	_
_	1	1_f	0_f	0_f	_	_
_	1_f	1_f	0_f	0_f	_	_

1.1.4 Fonctions récursives

Les fonctions récursives sont un ensemble de fonctions défini par clôture : il s'agit du plus petit ensemble de fonctions contenant un certain nombre de fonctions de base et stable pour des opérateurs sur ces fonctions. Une algèbre de fonctions est un tel ensemble de fonctions défini par clôture. La notation suivante due à Peter Clote ([Clo98]) permet de définir de façon concise les algèbres de fonctions.

Définition 1.10 (Algèbre de fonctions) *Soient $\mathcal{F}$ un ensemble de fonctions et $\mathcal{O}$ un ensemble d'opérateurs sur les fonctions, c'est-à-dire d'éléments o_i prenant en argument une ou plusieurs fonctions et générant une fonction.*

$[\mathcal{F};\mathcal{O}]$ est le plus petit ensemble de fonctions contenant $\mathcal{F}$ et stable par les opérateurs de $\mathcal{O}$.

Le lecteur pourra se référer à [Odi92] pour plus de détails sur les fonctions récursives et [Ros84] pour plus de détails sur les fonctions sous-récursives Les fonctions de base utiles pour définir l'ensemble des fonctions récursives sont 0 : fonction nulle (on peut la définir de n'importe quelle arité), la fonction successeur

$$S : \begin{cases} \mathbb{N} & \to \mathbb{N} \\ n & \mapsto n+1 \end{cases}$$

et les fonctions projections

$$U_i^n : \begin{cases} \mathbb{N}^n & \to \mathbb{N} \\ \vec{x} & \mapsto x_i \end{cases}$$

qui seront par la suite globalement désignées par U.

Les opérateurs sont la composition, définie de façon naturelle : $\mathrm{COMP}(f,g) = f \circ g$, la récursion primitive et le schéma de minimisation μ.

Définition 1.11 (Récursion primitive) *Étant données deux fonctions $g : \mathbb{N}^n \to \mathbb{N}^p$ et $h : \mathbb{N}^{n+p+1} \to \mathbb{N}^p$, on définit $f = \mathrm{REC}(g,h)$ comme l'unique fonction de $\mathbb{N}^{n+1} \to \mathbb{N}^p$ définie par*

$$f(\vec{x},0) = g(\vec{x})$$
$$f(\vec{x},y+1) = h(\vec{x},y,f(\vec{x},y)).$$

Définition 1.12 (Schéma de minimisation) *Étant donnée une fonction $g : \mathbb{N}^{n+1} \to \mathbb{N}$, $f = \mu(g)$ est définie par*

$$f : \begin{cases} \mathbb{N}^n & \to \mathbb{N} \\ x & \mapsto \begin{cases} \min\{y \in \mathbb{N}; g(x,y) = 0 \text{ et } \forall z \leq y, g(x,z) \text{ est défini.}\} \\ \text{non défini si } \{y \in \mathbb{N}; g(x,y) = 0 \text{ et } \forall z \leq y, g(x,z) \text{ est défini.}\} = \emptyset. \end{cases} \end{cases}$$

$\mu(f)$ peut ne pas être définie sur tout $\mathbb{N}^n$. Les fonctions de base sont définies sur tout $\mathbb{N}$ (ou $\mathbb{N}^p$), les opérateurs de composition et de récursion primitive ne font pas apparaître de fonctions partielles. Seul l'opérateur de minimisation fait apparaître des fonctions partielles. On peut définir trois classes de fonctions récursives : les fonctions récursives totales et les fonctions récursives partielles (que nous appellerons fonctions récursives) et les fonctions récursives préfixe-partielles.

Définition 1.13 (Fonctions récursives partielles)

$$\mathcal{R}ec(\mathbb{N}) = [0, S, U; \mathrm{COMP}, \mathrm{REC}, \mu]$$

Définition 1.14 (Fonctions totales récursives)

$$\mathcal{TR}ec(\mathbb{N}) = \{f \in \mathcal{R}ec(\mathbb{N}); f \ \textit{totale}\}.$$

Définition 1.15 (Fonctions récursives préfixe-partielles)

$$\mathcal{PPR}ec(\mathbb{N}) = \{f \in \mathcal{R}ec(\mathbb{N}); \exists n \in \mathbb{N}, \mathcal{D}om(f) = [0, n]\}$$

Nous allons également définir quelques classes de fonctions sous-récursives :

Définition 1.16 (Fonctions primitives récursives)

$$\mathcal{PR}(\mathbb{N}) = [0, S, U; \mathrm{COMP}, \mathrm{REC}]$$

On peut noter que toutes les fonctions primitives récursives sont totales.

Définition 1.17 (Fonctions élémentaires)

$$\mathscr{E}(\mathbb{N}) = [0, S, U, +, \times, \ominus; \mathrm{COMP}, \mathrm{BSUM}, \mathrm{BPROD}]$$

avec

- $\ominus : \begin{cases} \mathbb{N}^2 & \to \mathbb{N} \\ n_1, n_2 & \mapsto \begin{cases} n_1 - n_2 & si\ n_1 - n_2 \geq 0 \\ 0 & sinon. \end{cases} \end{cases}$
- $\mathrm{BSUM}(f) : \begin{cases} \mathbb{N}^{k+1} & \to \mathbb{N}^{k+1} \\ \vec{x}, y & \mapsto \sum_{z<y} f(\vec{x}, z) \end{cases}$
- $\mathrm{BPROD}(f) : \begin{cases} \mathbb{N}^{k+1} & \to \mathbb{N}^{k+1} \\ \vec{x}, y & \mapsto \prod_{z<y} f(\vec{x}, z) \end{cases}$

Définition 1.18 (Hiérarchie de Grzegroczyk) *Pour $n \geq 3$, on définit le n-ième niveau de la hiérarchie de Grzegorczyk par*

$$\mathscr{E}_n(\mathbb{N}) = [0, S, U, +, \ominus, E_{n-1}; \mathrm{COMP}, \mathrm{BSUM}, \mathrm{BPROD}]$$

avec

- $E_2(x) = 2^x$
- *Pour* $n \geq 2$, $E_{n+1}(x) = E_n^{[x]}(1)$ *avec* $f^{[0]}(x) = x$ *et* $f^{[n+1]}(x) = f\left(f^{[n]}(x)\right)$.

Proposition 1.19 ([Ros84, Odi92]) *Pour $n > 3$, on a*

$$\mathscr{E}(\mathbb{N}) = \mathscr{E}_3(\mathbb{N}) \subsetneq \mathscr{E}_n(\mathbb{N}) \subsetneq \mathscr{E}_{n+1}(\mathbb{N}) \subsetneq \mathcal{PR}(\mathbb{N}) \subsetneq \mathcal{R}ec(\mathbb{N}).$$

Et

$$\bigcup_{n\in\mathbb{N}} \mathscr{E}_n(\mathbb{N}) = \mathcal{PR}(\mathbb{N}).$$

Si l'on note $e_2^{[d]}$ l'itérée d-ième de la fonction $n \mapsto 2^n$, que l'on peut définir par $e_2^{[0]} = x \mapsto x$ et $e_2^{[d+1]} = x \mapsto 2^{e_2^{[d]}(x)}$.

Proposition 1.20 ([Cut80]) *Soit $f \in \mathcal{R}ec(\mathbb{N})$, s'il existe $a, d \in \mathbb{N}$ tels que $\forall n, f(n)$ se calcule en temps borné par $ae_2^{[d]}(n)$, alors $f \in \mathscr{E}(\mathbb{N})$*

Proposition 1.21 ([Ros84, Odi92]) *Les fonctions récursives sont dénombrables.*

Démonstration : Nous aurions en fait pu écrire une proposition plus générale : toute algèbre de fonctions définie à l'aide de la notation 1.10 est dénombrable.

En effet, on peut numéroter ses éléments par récurrence structurelle. Nous allons montrer sur l'exemple de l'algèbre $\mathcal{A} = [f_1, f_2; o_1, o_2]$ comment construire une fonction injective ψ qui à une fonction appartenant à l'algèbre $\mathcal{A}$ associe un entier. Posons $\psi(f_1) = 1$ et $\psi(f_2) = 2$. Supposons que o_1 est d'arité 1 et que o_2 est d'arité 2. On choisit 3 (la somme des arités des o_i) nombres premiers supérieurs à tous les $\psi(f_i)$. Ici 3, 5 et 7 font parfaitement l'affaire. Et on pose $\psi(o_1(f)) = 3^{\psi(f)}$, $\psi(o_2(f, g)) = 5^{\psi(f)}7^{\psi(g)}$.

Cette fonction ψ est bien définie et est injective. Remarquons tout d'abord que

$$\{\psi(f); f \in [f_1, f_2; o_1, o_2]\}$$

est inclus dans $\{1, 2\} \cup 3^{\mathbb{N}} \cup 5^{\mathbb{N}}7^{\mathbb{N}}$. Si $\psi(f) = \psi(g)$, soit ils valent 1 ou 2, et comme l'application de o_1 ou o_2 fait croître ψ, cela signifie que $f = g = f_i$. Sinon, $\psi(f)$ est une puissance de 3 auquel cas, $f = o_1(h)$ et $g = o_1(l)$ avec $\psi(h) = \psi(l)$ ou bien $\psi(f)$ est de la forme 5^n7^m et donc f et g sont des applications de o_2 à deux fonctions qui ont respectivement les mêmes ψ. Par récurrence, on montre que finalement, $\psi(f) = \psi(g) \Rightarrow f = g$. □

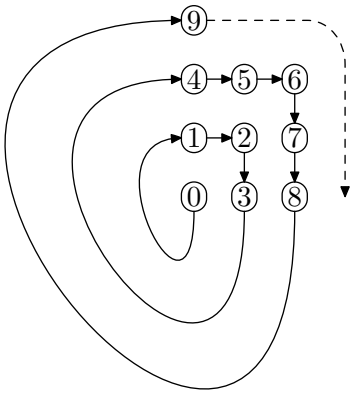

FIG. 1.2: Un indexage des couples d'entiers naturels par les entiers naturels

Il est donc possible de numéroter les fonctions récursives. De plus il est possible d'encoder 2 entiers en un seul par exemple en posant $\phi(n,p) = 2^n \times 3^p$, chaque paire d'entiers est associée à un entier à partir duquel il est possible de retrouver de façon calculable l'entier de départ. On peut construire une fonction plus évoluée faisant véritablement une bijection de $\mathbb{N}^2$ sur $\mathbb{N}$ comme indiqué dans la figure 1.2.

On peut écrire cette bijection

$$\phi(n,p) = (1 \ominus (n \ominus p)) \left(p^2 + n\right) + (1 \ominus (p \ominus (n+1))) \left((n+1)^2 \ominus (p+1)\right).$$

Cette fonction est élémentaire.

La réciproque de cette fonction est également élémentaire. Considérons c, on veut trouver n et p tels que $\phi(n,p) = c$. Pour cela, nous allons utiliser les fonctions racine, min et max. Nous notons $\lfloor x \rfloor$ la partie entière de x.

Lemme 1.22 *Calculer $\lfloor \sqrt{n} \rfloor$ se fait de façon élémentaire.*

Démonstration : $\lfloor \sqrt{n} \rfloor$ est le plus grand entier k tel que $k^2 \leq n$.

$1 \ominus (k^2 \ominus n) = \begin{cases} 1 & \text{si } k^2 \leq n \\ 0 & \text{sinon} \end{cases}$

Donc, $\lfloor \sqrt{n} \rfloor = \sum_{k \leq n} 1 \ominus (k^2 \ominus n) \ominus 1$. □

Lemme 1.23 max *et* min *sont élémentaires.*

Démonstration :

$$\max(a,b) = a + (b \ominus a)$$
$$\min(a,b) = a \ominus (a \ominus b)$$

□

$\lfloor\sqrt{c}\rfloor$ indique dans quel circuit droit se trouve le couple (n,p). Il suffit ensuite de décider si l'on se trouve dans la branche horizontale ou dans la branche verticale.

$$\begin{aligned} n &= \min\left(c \ominus \lfloor\sqrt{c}\rfloor^2, \lfloor\sqrt{c}\rfloor\right) \\ p &= \min\left(\lfloor\sqrt{c}\rfloor, (\lfloor\sqrt{c}+1\rfloor^2 \ominus (c+1)\right). \end{aligned}$$

Notons $\langle n,p\rangle = \phi(n,p)$. On peut donc encoder deux entiers en un seul. Mais on peut aller plus loin et encoder 3 entiers en un avec $\langle m,\langle n,p\rangle\rangle$. Les fonctions discrètes de n'importe quelle arité peuvent donc être ramenées à des fonctions d'arité 1.

Définition 1.24 (encodage) *Il existe une fonction élémentaire bijective dont la réciproque est élémentaire qui encode deux entiers en un seul. On note cette fonction $\langle .,. \rangle$.*

Il est donc possible d'indexer les fonctions récursives par les entiers. Il existe donc une fonction $u : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ qui à un entier n associe la fonction récursive indexée par n. Qui plus est, il existe une telle fonction u récursive. Grâce à l'encodage présenté par la définition 1.24, on peut même définir une fonction récursive ϕ_u qui à $\langle x,y\rangle$ associe $\phi_x(y)$ où ϕ_x est la x-ième fonction récursive pour l'indexage défini précédemment. On peut remarquer que $\phi_u(\langle u,x\rangle) = \phi_u(x)$.

Proposition 1.25 (Fonction universelle, [Odi92]) *Il existe $u \in \mathbb{N}$ tel que*

$$\forall x,y \in \mathbb{N}, \phi_u(\langle x,y\rangle) = \phi_x(y).$$

ϕ_u est une fonction universelle.

1.1.5 Thèse de Church-Turing

Ces différents modèles permettent de caractériser des ensembles de fonctions « calculables ». Des résultats reliant ces classes de fonctions existent. Par exemple, les fonctions récursives se relient aux fonctions Turing-calculables :

Proposition 1.26 (Turing) *Les fonctions récursives sont exactement les fonctions Turing-calculables.*

Proposition 1.27 (Minsky) *Les fonctions calculées par automate à deux compteurs sont exactement les fonctions Turing-calculables.*

Tous les modèles présentés précédemment sont en fait équivalents. Les fonctions calculées par automates à deux compteurs, par automates cellulaires ou par machine de Turing sont exactement les fonctions récursives.

Nous avons présenté des classes de fonctions sous-récursives qui sont strictement incluses dans l'ensemble des fonctions récursives. Les automates à un seul compteur sont également moins puissants que les automates à deux compteurs. Par contre, ajouter un compteur à un automate à deux compteurs n'augmente pas sa puissance. Il semble que

cette classe de fonctions, avec ses multiples définitions équivalentes, contienne toutes les fonctions que l'on est en droit d'attendre comme pouvant être produites par un procédé automatique.

La thèse de Church-Turing affirme que n'importe quelle machine à calculer, n'importe quel modèle suffisamment puissant d'une telle machine calculera exactement les fonctions récursives. On peut l'énoncer de la façon suivante :

Proposition 1.28 (Thèse de Church-Turing) *Un modèle de calcul discret raisonnable ne calcule que des fonctions récursives.*

Bien sûr, dans cet exposé, le terme « raisonnable » est attaquable, car on peut décider de ne définir un modèle de calcul comme raisonnable que s'il calcule les fonctions récursives mais pas plus. D'autre part, des modèles super-Turing (c'est-à-dire capables de calculer des fonctions que ne peut calculer une machine de Turing) ont été exhibés, comme le soutient par exemple [Cop98].

Pour un exposé plus détaillé sur la thèse de Church-Turing, et sur les différentes interprétations de cette thèse, le lecteur pourra se référer à [Cop02].

1.2 Modèles hybrides

Ce que nous appelons ici modèles hybrides sont des modèles qui travaillent sur une structure continue (typiquement, les réels, $\mathbb{R}^2$ ou $\mathbb{R}^3$) mais en temps discret, c'est à dire pas à pas. Parmi ces modèles, nous allons présenter succinctement les systèmes dynamiques, l'analyse récursive, qui sera plus amplement et précisément définie en section 2.1 et le modèle de Blum, Shub et Smale.

1.2.1 Système dynamique à temps discret

Un système dynamique à temps discret suit le principe des suites récurrentes : on se donne un espace X qui ici pourra être $\mathbb{R}^d$ ou un sous-ensemble de $\mathbb{R}^d$, et une fonction $f : X \to X$. Étant donné un point initial $x_0 \in X$, on peut alors créer une suite $(u_n)_{n\in\mathbb{N}}$ définie par $u_{n+1} = f(u_n)$. Il est donc possible de définir la trajectoire du système partant de x_0.

Définition 1.29 (Systèmes dynamiques à temps discret) *Un système dynamique à temps discret est un couple (X, f) où X est l'espace du système et $f : X \to X$ est la fonction d'évolution du système.*

La trajectoire du système partant d'un point $x_0 \in X$ est la suite (x_n) définie par $x_{n+1} = f(x_n)$.

La question que l'on peut se poser pour un tel système est l'appartenance d'un point x à la trajectoire d'un point x_0. Il s'agit d'un problème d'atteignabilité.

Définition 1.30 (Atteignabilité) *Étant donné un système dynamique (X, f) et un point x. On dit que x est atteignable depuis x_0 s'il existe $(x_1, x_2, \ldots, x_m)$ tel que*

$$x_m = x$$
$$\forall n, x_{n+1} = f(x_n)$$

Étant donné un ensemble $T \subseteq X$, on dit que T est atteignable depuis x_0 s'il existe $x \in T$ atteignable depuis x_0.

On peut ainsi définir une notion de reconnaissance de langage par ces systèmes en posant un ensemble $A \subseteq X$ acceptant, et en considérant qu'un point x_0 appartient au langage considéré si A est atteignable depuis x_0. Cela représente une notion de semi-décision de l'appartenance à un langage. En posant un ensemble $R \subseteq X$ refusant, on peut obtenir une notion de décision de l'appartenance à un langage. Bien sur, il ne doit pas exister de trajectoire menant de R à T et vice-versa. Pour que cette deuxième notion puisse être vérifiée de façon effective, on doit supposer que pour tout point, A ou R est atteignable et que ces deux ensembles sont séparés :

$$\forall a \in A, r \in R, \exists x \in [a, r] \text{ tel que } x \notin R \cup A$$

où $[a, r]$ représente le segment liant les points a et r.

Définition 1.31 (Langage semi-reconnu par système dynamique) *Soit $\mathcal{L} \subseteq X$. On dit que le langage $\mathcal{L}$ est semi-reconnu par le système dynamique à temps discret (X, f) avec états acceptants $A \subseteq X$ si*

$$x \in \mathcal{L} \Leftrightarrow A \text{ est atteignable depuis } x.$$

1.2.2 Analyse récursive

Le modèle de l'analyse récursive peut être interprété comme une adaptation des machines de Turing à un espace continu. On a vu que les machines de Turing prenaient leur entrée discrète sur un ruban infini, et de même écrivaient leur résultat discret sur un ruban infini. On obtient donc des fonction $\mathbb{N} \to \mathbb{N}$. Mais il est possible de mettre sur le ruban d'entrée une infinité d'entiers, c'est à dire de coder une fonction $\mathbb{N} \to \mathbb{N}$ (en d'autres termes, une suite à valeurs dans $\mathbb{N}$ indexée par $\mathbb{N}$). Et de même, le ruban de sortie peut représenter une fonction $\mathbb{N} \to \mathbb{N}$. Il s'agit alors de fonction d'ordre supérieur.

On appelle machine de type 1 la machine de Turing qui représente des fonctions $\mathbb{N} \to \mathbb{N}$.

Les machines de type 2 représentent des fonctions $(\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to \mathbb{N})$.

On a vu dans la sous-section 1.1.4 que deux entiers naturels pouvaient être codés en un seul, il est facile de voir qu'un couple contenant un entier relatif et un entier naturel peut être codé par un unique entier naturel, donc un rationnel peut être représenté par un entier naturel, et l'on sait que $\mathbb{Q}$ est dense dans $\mathbb{R}$. Donc une suite de valeurs entières peut

représenter un réel. Cela signifie que ces machines de type 2 peuvent être considérées comme des machines calculant des fonctions de $\mathbb{R}$ dans $\mathbb{R}$ à condition qu'elles soient bien définies : un même réel peut être représenté par différentes suites de rationnels, donc toutes les suites représentant un même réel doivent avoir pour image une suite représentant le même réel.

Un résultat crucial pour ce modèle montre que toute fonction calculable au sens de l'analyse récursive est continue.

Les machines de type 2 offrent un cadre pour l'analyse récursive. La section 2.1 présente de façon plus détaillée et précise l'analyse récursive.

1.2.3 Modèle de Blum, Shub et Smale

Un autre moyen d'adapter la machine de Turing pour calculer sur les réels serait au lieu de mettre sur le ruban des éléments d'un alphabet fini d'utiliser directement des réels sur ce ruban. De façon très simplifiée, c'est le principe du modèle BSS dû à Lenore Blum, Mike Shub et Steve Smale [BSS89, BCSS98].

Une machine BSS est constituée d'un ruban infini et d'un système de contrôle. Le ruban infini peut être représenté par une suite de réels $(x_n)_{n\in\mathbb{Z}} \in \mathbb{R}^{\mathbb{Z}}$ (on peut choisir de travailler sur d'autres espaces que les réels, mais nous nous contenterons ici de cette définition). Le système de contrôle est doté d'un nombre fini d'états $Q = \{0, 1, \ldots, m\}$ et d'instructions. Les différentes instructions possibles sont

– déplacer le ruban vers la droite. C'est-à-dire changer la suite en

$$(x_{n+1})_{n\in\mathbb{Z}},$$

– déplacer le ruban vers la gauche. C'est-à-dire changer la suite en

$$(x_{n-1})_{n\in\mathbb{Z}},$$

– changer d'état conditionnellement :

$$\text{Si } \left\{ \begin{array}{l} x_0 > 0 \\ x_0 = 0 \end{array} \right\}, q \leftarrow q',$$

– changer la valeur de la suite sous la tête de lecture (x_0) en fonction éventuellement d'une constante k par

$$\begin{aligned} x_0 &\leftarrow -x_0 \\ x_0 &\leftarrow k \\ x_0 &\leftarrow kx_0 \\ x_0 &\leftarrow x_0 + x_1 \\ x_0 &\leftarrow x_0 x_1. \end{aligned}$$

Un programme de machine BSS est constitué d'une séquence de telles instructions pour chaque état.

Parmi les fonctions que peut calculer ce modèle, on trouve la fonction signe. Rappelons que la fonction signe n'étant pas continue, elle n'est pas calculable au sens de l'analyse récursive. Par contre, des fonctions calculables en analyse récursive ne sont pas calculables par machine BSS. Par exemple, exp ne peut être calculée dans ce modèle.

1.3 Modèles continus

Les modèles présentés dans la précédente section travaillent sur des ensembles continus mais en temps discret. Les systèmes dynamiques modélisent un mécanisme d'itération, l'analyse récursive conçoit les fonctions sur les réels à l'aide d'une machine opérant sur des entiers, le modèle BSS traite ses données pas à pas.

Il est cependant possible de concevoir des modèles qui travaillent en temps continu. Cela comporte malheureusement des risques comme l'apparition de phénomènes physiquement irréalistes. Nous discuterons de ces risques à l'aide d'un exemple simple dans la section 1.3.5. Ces modèles présentent souvent la capacité de calculer plus que les machines de Turing, mise en évidence par la possibilité de résoudre le problème de la halte des machines de Turing discrètes. Ce trait est exprimé dans [Sta01] qui montre que les modèles super-Turing sont une réalité.

Nous présentons maintenant plusieurs modèles continus qui ont été considérés comme des modèles de calcul dans la littérature. Il ne s'agit que d'un survol de quelques modèles : nous ne présenterons que quelques traits de base de chacun ; nous ne présentons pas tous les modèles de calcul continus, par exemple, nous ne parlons pas ici des réseaux de neurones.

1.3.1 General Purpose Analog Computer

Le General Purpose Analog Computer (GPAC) sera présenté plus en détail dans la section 2.3. Le GPAC modélise des machines ayant réellement existé appelées analyseurs différentiels. Le fonctionnement de ces machines est assez comparable à des circuits électriques : on dispose de quelques composants de base (multiplicateur, additionneur, intégrateur) et de constantes et on peut les interconnecter. Les « fils » reliant ces composants véhiculent une quantité mesurable qui varie en fonction du temps (une intensité par exemple). Ce sont ces fonctions du temps qui constituent les fonctions générées par le GPAC. On peut établir une notion de calcul en étudiant la valeur d'un paramètre en fonction du temps et des entrées que peuvent constituer les constantes.

De telles machines peuvent être réalisées à l'aide de composants électroniques, ou bien de composants mécaniques. Les valeurs de la fonction sont alors représentées par la rotation d'une roue ou la translation d'une règle. Pour se convaincre de la réalité de ces machines, le lecteur pourra consulter `http://www.meccano.us/differential_analyzers/robinson_da/` qui montre des analyseurs différentiels réalisés avec des pièces de Meccano^{TM}.

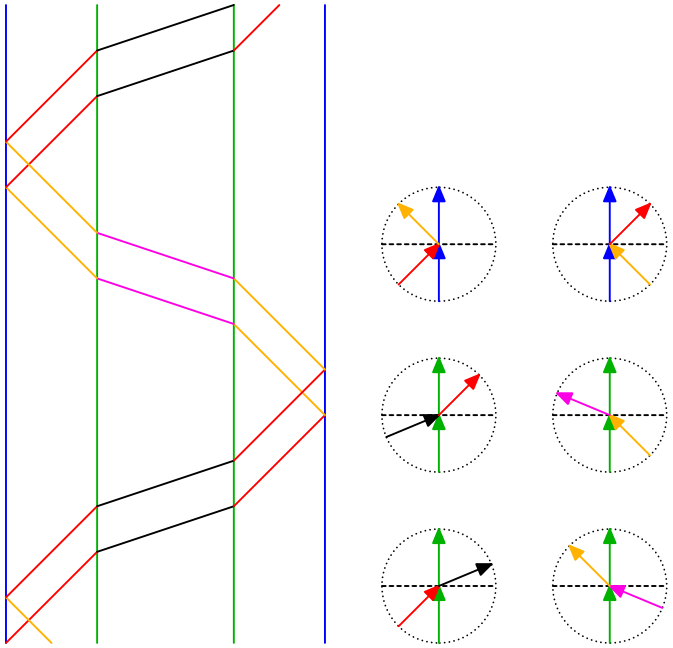

FIG. 1.3: Une machine à signaux

1.3.2 Machines à signaux

Les machines à signaux sont une adaptation des automates cellulaires à un domaine continu. De nombreux programmes sur les automates cellulaires font intervenir la notion de signaux : un signal est un état qui se propage dans une direction à chaque pas de calcul. Les étapes intéressantes du calcul sont alors celles où deux signaux ou plus se rencontrent, cas où l'application des règles peut faire apparaître de nouveaux états qui pourront éventuellement se propager. Le modèle des machines à signaux est présenté et étudié dans [DL03, DL05].

Dans le cadre des machines à signaux, l'espace de calcul n'est plus une grille discrète mais la ligne réelle (ou éventuellement $\mathbb{R}^n$). Les « états » sont alors représentés par une couleur et une direction. L'équivalent des règles discrètes est l'énumération des différents cas de collisions possibles entre ces signaux et ce qui résulte de ces collisions (il peut y avoir disparition de certains des signaux arrivant, création de nouveaux signaux, ou des combinaisons de ces possibilités). Un exemple de représentation d'une machine à signaux est présentée en figure 1.3. On peut y voir l'évolution d'un système en fonction des règles présentées. Les règles montrent les signaux se heurtant en bas, et les signaux qui résultent de la collision au dessus. La configuration du système à un instant donné est la coupe horizontale du diagramme à la hauteur correspondante. Le temps est croissant vers le haut.

Ce modèle permet simuler les machines à deux compteurs, il est donc au moins aussi puissant que les machines de Turing discrètes.

1.3.3 Systèmes dynamiques à temps continu

La sous-section 1.2.1 traitait des systèmes dynamiques à temps discret. Il s'agit de modèles qui reposent sur l'itération d'une fonction. Les systèmes dynamiques à temps continus remplacent l'itération pour décider de la configuration suivante par un opérateur différentiel indiquant la façon dont va varier la configuration.

Définition 1.32 (Systèmes dynamiques à temps continu) *On dénomme système dynamique à temps continu un couple (X, f) où $X \subset \mathbb{R}^d$ est l'espace du système et $f : X \to \mathbb{R}^d$ est la fonction d'évolution du système.*

La trajectoire d'un système (X, f) à partir d'un point $x_0 \in X$ est l'ensemble des points de la courbe différentielle $h : \mathbb{R} \to X$ définie par

$$\begin{aligned} h(0) &= x_0 \\ h'(t) &= f(h(t)) \end{aligned}$$

Définition 1.33 (Atteignabilité) *Étant donnés un système dynamique à temps continu (X, f) et un point $x \in X$, on dit que x est atteignable depuis x_0 s'il existe $t \in \mathbb{R}^+$ tel que $h(t) = x$ où h est la fonction définie précédemment.*

Un ensemble $T \subseteq X$ est dit atteignable à partir de x_0 s'il existe $t \in T$ atteignable depuis x_0.

On peut, de la même façon que pour les systèmes dynamiques à temps discret, définir une notion de langage reconnu, ou semi-reconnu. Il est toutefois naturel de réclamer des conditions fortes sur les ensembles d'acceptation pour que la reconnaissance soit effective. Par exemple, on peut imposer que l'ensemble d'acceptation A soit d'une certaine façon absorbant, c'est-à-dire qu'aucune trajectoire n'en sort. On peut également requérir qu'il existe une enveloppe de A dans laquelle toute trajectoire entrante va ensuite dans A pour éviter des trajectoires tangentes à A.

1.3.3.a Systèmes à dérivée constante par morceaux

Un système à dérivée constante par morceaux est un système dynamique dans lequel l'espace considéré est un $\mathbb{R}^d$ et la fonction d'évolution est constante par morceaux. Cela signifie que l'on peut découper l'espace en morceaux sur lesquels les trajectoires sont parallèles, dirigées par un vecteur vitesse constant, les trajectoires ne changeant de direction que lors du passage d'un morceau à un autre. [AM98, Bou99]

Définition 1.34 (Système à dérivée constante par morceaux) *Un système à dérivée constante par morceaux est un système dynamique à temps continu (X, f) où $X = \mathbb{R}^d$ et f est une fonction constante par morceaux.*

La figure 1.4 montre un système à dérivée constante par morceaux, les vecteurs vitesse dans chaque région et une portion de la trajectoire partant du point x_0.

Ce système est à temps continu, cependant, les points importants d'une trajectoire sont les transitions d'une région à une autre. Il est donc possible de définir un temps discret caractérisé par le nombre de frontières rencontrées.

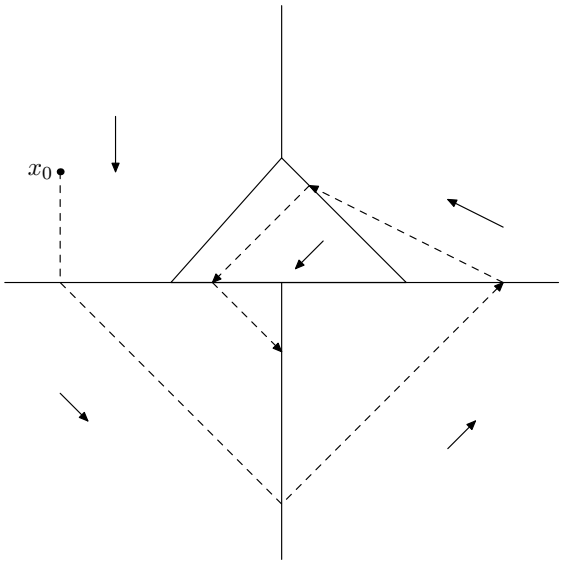

FIG. 1.4: Un système à dérivée constante par morceaux

1.3.3.b Problème des n corps

Dans le cadre de la mécanique newtonienne, les mouvements de planètes sont régis par des équations de la forme

$$\begin{aligned} x_1'' &= M_2 \frac{x_2 - x_1}{\|x_2 - x_1\|^3} + M_3 \frac{x_3 - x_1}{\|x_3 - x_1\|^3} \\ x_2' &= M_1 \frac{x_1 - x_2}{\|x_1 - x_2\|^3} + M_3 \frac{x_3 - x_2}{\|x_3 - x_2\|^3} \\ x_3' &= M_1 \frac{x_1 - x_3}{\|x_1 - x_3\|^3} + M_2 \frac{x_2 - x_3}{\|x_2 - x_3\|^3}. \end{aligned}$$

Dans le cas de 2 corps, on sait que le déplacement de x_2 par rapport à x_1 est une conique dont x_1 est un foyer. Dans le cas de 3 corps ou plus, on ne sait prévoir les trajectoires des astres que dans de rares cas particuliers.

Warren D. Smith propose dans [Smi06] de considérer le problème des n corps comme un modèle de calcul. L'encodage d'une entrée se fait en choisissant n puis en plaçant les n corps à l'instant initial et en imposant leurs vitesses initiales. On peut alors décider que l'entrée est acceptée si le système s'effondre, ou simplement si l'un des corps pénètre dans une zone prédéfinie, que l'entrée est refusée si le système explose, ou si un des corps sort d'une zone « sûre.»

Il s'agit donc d'un système dynamique à temps continu. Il est à noter que ce système ne peut être simulé de façon générale par une machine de Turing. Le problème des n corps est donc un modèle super-Turing.

1.3.4 Fonctions $\mathbb{R}$-récursives

Les fonctions $\mathbb{R}$-récursives sont définies comme étant des algèbres de fonctions sur les réels. Elles sont donc comparables aux fonctions discrètes récursives. En fait, ces fonctions ont été définies avec l'idée d'imiter les fonctions discrètes récursives. Nous l'avons classée ici dans les modèles de calcul continus, mais le temps n'est pas à proprement parler un paramètre de ces fonctions, il n'est donc pas très clair qu'elles travaillent en temps réel.

La classe $\mathscr{G}$ décrite par Cristopher Moore dans [Moo96] est définie comme contenant 0, 1, les projections et stable par composition, intégration et recherche de zéro. Cette classe contient les fonctions générées par GPAC, mais aussi des fonctions qui ne sont pas Turing-calculables : il est par exemple possible de résoudre le problème de la halte des machines de Turing à l'aide de fonctions $\mathbb{R}$-récursives.

Nous décrirons cette classe de fonctions plus en détails, ainsi que des classes qui s'en inspirent dans la section 2.2. L'un des objectifs de cette thèse est de caractériser les fonctions calculables au sens de l'analyse récursive par des sous-classes de fonctions $\mathbb{R}$-récursives.

1.3.5 Remarque physique

Ces modèles sont souvent affligés de caractéristiques que la physique réprouve. Nous allons ici expliquer le paradoxe de Zénon pour les modèles de calcul sur les réels. Empêcher l'apparition de ce phénomène est un point crucial pour parvenir à des modèles physiquement raisonnables. Nous montrerons ensuite que le calcul d'une intégrale est tout à fait faisable en physique et montrerons en particulier une réalisation électronique et une réalisation mécanique d'intégrateurs.

1.3.5.a Paradoxe de Zénon

Parmi les modèles entrevus dans cette partie, certains comportent des tares physiquement impardonnables : mener à bien un calcul peut éventuellement se faire en un temps fini mais requérir une énergie infinie. Pour illustrer ceci, présentons les machines de Turing accélérantes.

Une machine de Turing accélérante fonctionne exactement comme une machine de Turing à une exception près : chaque étape de calcul ne prend pas le même temps. Le premier pas de calcul (on appelle pas de calcul chaque application de la fonction de transition) est effectué en temps $\frac{1}{2}$, le second en temps $\frac{1}{4}$, de façon générale, le n-ième en temps 2^{-n}. Ainsi, tout ensemble récursivement énumérable est calculable par une machine de Turing accélérante en temps 1 : soit la machine a atteint l'état d'acceptation avant le temps 1, donc le calcul s'est fini de façon normale, soit au temps 1, le calcul ne s'est pas arrêté, mais tous les entiers ayant été exploré, on sait que l'entrée ne fait pas partie de l'ensemble testé. On a donc décidé cet ensemble. Cette construction n'est pas sans rappeler le paradoxe de la flèche de Zénon d'Élée : la flèche parcourt d'abord la moitié de la distance la séparant de la cible, puis la moitié de ce qui reste, et cela indéfiniment, mais comme ces étapes prennent de moins en moins de temps, la flèche peut tout de même atteindre la cible en temps fini.

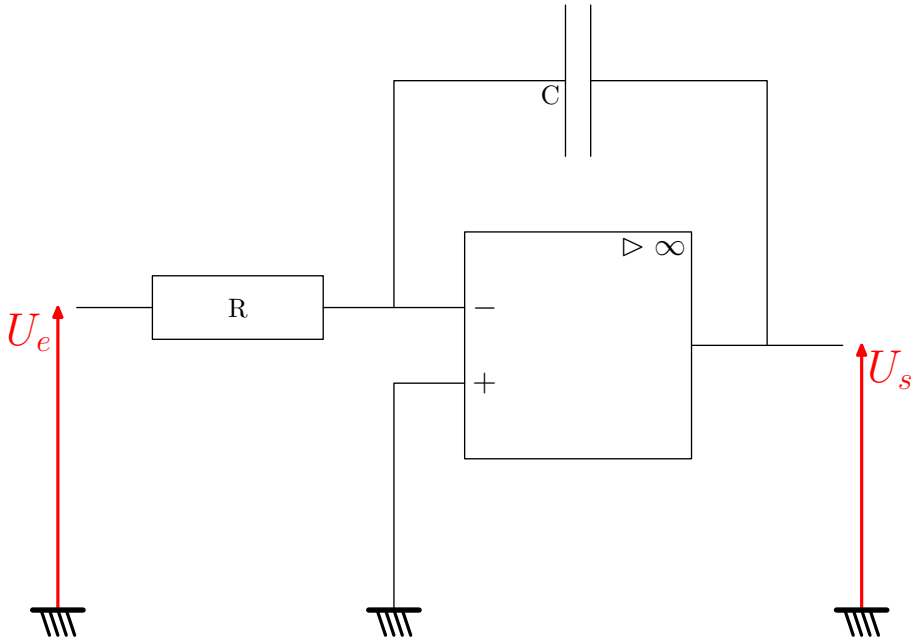

FIG. 1.5: Schéma d'un intégrateur électronique : $U_s = K \int_0^t U_e$

Une machine de Turing accélérante résout sans difficulté le problème de la halte des machines de Turing. Cependant, une infinité de pas de calculs ont potentiellement été réalisés pour arriver à ce résultat, la tête de lecture a pu faire une infinité de déplacements en un temps fini, or chaque déplacement coûte autant d'énergie (en fait, intuitivement, plus le déplacement est rapide, plus il devrait consommer d'énergie) donc la quantité d'énergie consommée par cette machine n'est pas finie.

Pourtant ce modèle simple de machine semble physiquement plausible : dans le cadre de la relativité générale, si une machine de Turing est envoyée dans un trou noir, elle peut parcourir un temps infini pendant un temps qui sera fini pour un observateur [Hog92, EN02], et l'observateur peut recevoir de cette machine des messages lui signalant si oui ou non la machine s'est arrêtée.

Les machines à signaux présentent également ce phénomène de zénonisme : un calcul prenant un temps infini est nécessairement inclus dans un morceau de plan s'évasant au cours du temps ou de largeur constante. En jouant sur les directions des signaux, il est possible de forcer le calcul à s'effectuer dans un triangle, c'est-à-dire en temps fini.

1.3.5.b Intégration physique

Le problème des n corps est par essence un problème physique. Mais le General Purpose Analog Computer ou les fonctions $\mathbb{R}$-récursives ne semblent pas intuitivement simples à réaliser de façon physique. Considérons les constituants de base du GPAC : multiplication, addition, intégration. En électronique comme en mécanique, additionner entre

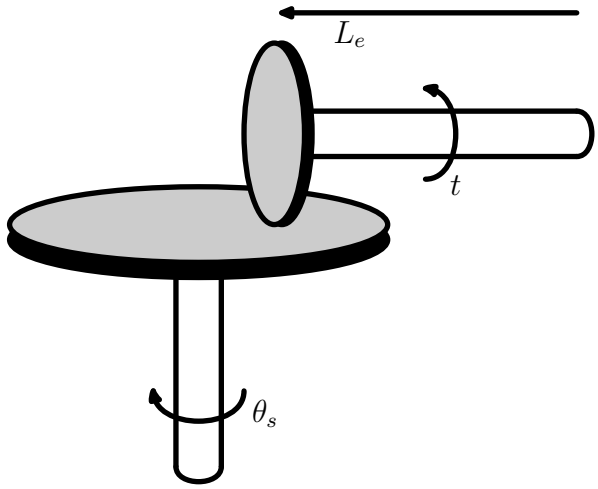

FIG. 1.6: Schéma d'un intégrateur mécanique : $\theta_s = K \int_0^t L_e$

eux deux signaux n'est pas compliqué. Multiplier un signal par une constante est également simple. Par contre, l'intégration d'un signal est *a priori* moins évidente. Cela est cependant possible : les figures 1.5 et 1.6 montrent des montages intégrateurs dans les domaines électronique et mécanique respectivement. À l'aide d'un intégrateur, il est possible d'obtenir la multiplication de deux signaux. Par contre, l'opérateur de recherche de zéro utilisé par les fonctions $\mathbb{R}$-récursives ne possède pas de contrepartie physique simple.

2 Compléments sur trois modèles de calcul sur les réels

> Peu à peu à cette hallucination succéda un regard moins égaré et moins grossissant. Le réel se faisait jour autour de lui, lui heurtait les yeux, lui heurtait les pieds
>
> *(Victor Hugo)*

Nous allons dans cette partie présenter plus en pronfondeur trois modèles de calcul sur les réels : l'analyse récursive que l'on peut faire remonter à la mi-vingtième siècle, les fonctions $\mathbb{R}$-récursives introduites récemment par Cristopher Moore et le General Purpose Analog Computer présenté par Claude Shannon en 1941.

2.1 Analyse récursive

L'idée de base de l'analyse récursive consiste à encoder un réel par une suite convergente de rationnels, et de faire travailler une machine classique sur cette entrée. On peut attribuer la paternité de cette idée à Alan Turing [Tur36], Daniel Lacombe [Lac55] et Andrzej Grzegorczyk [Grz57]. Nous utiliserons ici des définitions inspirées principalement d'ouvrages plus récents, en particulier celui de Klaus Weihrauch, [Wei00] qui présente l'analyse récursive sous l'angle de la calculabilité et le livre de Ker-I Ko, [Ko91] qui s'intéresse aux problèmes de complexité.

2.1.1 Définitions

Dans le cas discret, les machines travaillent sur une entrée représentée par un mot fini sur un alphabet fini (typiquement $\Sigma = \{0, 1\}$ ou $\Sigma = \{0, 1, 2, \ldots, 9\}$). Une fonction discrète est alors une fonction de Σ^* vers Σ^*. On capture ainsi bien les fonctions de $\mathbb{N}$ vers $\mathbb{N}$ car $\mathbb{N}$ et Σ^* sont liés par une bijection calculable simple.

Calculer sur les réels peut être vu comme calculer sur une suite représentant le réel plutôt que sur un mot fini. On note Σ^ω l'ensemble des suites d'éléments de Σ. On peut remarquer qu'en notations mathématiques classiques on peut noter cet ensemble $\Sigma^{\mathbb{N}}$: l'ensemble des suites indexées par $\mathbb{N}$ d'éléments de Σ. Par la suite, on utilisera indifféremment $\mathbb{N}$ pour Σ^* et $\mathbb{N}^{\mathbb{N}}$ pour Σ^ω. Nous appelons fonctionnelles les fonctions $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ par opposition aux fonctions $\mathbb{N} \to \mathbb{N}$.

Définition 2.1 (Fonctionnelle calculable) *Soit* $Map : \mathbb{N}^{\mathbb{N}} \times \mathbb{N} \to \mathbb{N}$ *qui a une suite* X *et un entier* n *associe le codage des* n *premiers termes de la suite* X*.*

Une fonctionnelle est calculable si elle peut être écrite comme

$$X \mapsto (\phi(Map(X, n)))_{n\in\mathbb{N}}$$

avec ϕ *récursive.*

Par la suite, nous passerons sous silence la fonction Map et appellerons de la même façon les sous-classes de fonctions récursives et la classe de fonctionnelles correspondante.

Définition 2.2 (Notation, Représentation) *Définissons les notions de notation et de représentation d'un ensemble :*

- *Une notation d'un ensemble* M *est une fonction surjective partielle* $\nu : \Sigma^* \to M$*.*
- *Une représentation d'un ensemble* M *est une fonction surjective partielle* $\delta : \Sigma^\omega \to M$*.*

Définissons quelques notations simples d'ensembles dénombrables indispensables : $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$.

Définition 2.3 (Notation de $\mathbb{N}$) *Soit* $\Sigma = \{0, 1\}$*.*

Soit

$$\nu_{\mathbb{N}} : \begin{cases} \{0\} \cup 1\Sigma^* \subseteq \Sigma^* & \to & \mathbb{N} \\ (a_k, \ldots, a_0) & \mapsto & \sum_{i=0}^{k} a_i 2^i. \end{cases}$$

$\nu_{\mathbb{N}}$ *est une notation de* $\mathbb{N}$*.*

Intuitivement, $\nu_{\mathbb{N}}$ *associe au codage en binaire d'un entier naturel cet entier lui-même.*

Définition 2.4 (Notation de $\mathbb{Z}$) *Soit* $\nu_{\mathbb{Z}} : \{0\} \cup 1\Sigma^* \cup -1\Sigma^* \to \mathbb{Z}$ *telle que si* $w \in \mathcal{D}om(\nu_{\mathbb{N}})$*,* $\nu_{\mathbb{Z}}(w) = \nu_{\mathbb{N}}(w)$*, et sinon,* $\nu_{\mathbb{Z}}(-w) = -\nu_{\mathbb{N}}(w)$*.*

Intuitivement, un entier relatif est soit un entier naturel (et dans ce cas, $\nu_{\mathbb{Z}}(w) = \nu_{\mathbb{N}}(w)$*), soit l'opposé d'un entier naturel.*

Définition 2.5 (Notation de $\mathbb{Q}$) *Soit* $\nu_{\mathbb{Q}}$ *définie par*

$$\nu_{\mathbb{Q}} : \begin{cases} \{u/v; u \in \mathcal{D}om(\nu_{\mathbb{Z}}), v \in \mathcal{D}om(\nu_{\mathbb{N}}) - \{0\}\} & \to & \mathbb{Q} \\ u/v & \to & \frac{\nu_{\mathbb{Z}}(u)}{\nu_{\mathbb{N}}(v)}. \end{cases}$$

Ce qui veut dire qu'on associe à un mot dénotant un entier relatif suivi d'un / représentant la barre de fraction suivi d'un mot dénotant un entier naturel non nul, le rationnel représentant le quotient du relatif par le naturel. On remarque que $\nu_{\mathbb{Q}}$ *n'est pas injective : plusieurs mots peuvent dénoter le même nombre (par exemple,* $\nu_{\mathbb{Q}}(1/10) = \nu_{\mathbb{Q}}(10/100) = 0,5$*).*

L'ensemble $\mathbb{R}$ n'étant pas dénombrable, il n'y a pas de notation de $\mathbb{R}$. Il existe cependant des représentations de $\mathbb{R}$. On peut décrire un réel x par l'ensemble des intervalles ouverts à bornes rationnelles le contenant : $\{]a,b[; a,b \in \mathbb{Q}, a < x < b\}$; ou bien par l'ensemble des rationnels plus petits que lui : $\{a \in \mathbb{Q}; a < x\}$; ou encore par l'ensemble des rationnels plus grands que lui : $\{a \in \mathbb{Q}; a > x\}$.

Définition 2.6 (Cubes rationnels) *Pour $n \geq 1$, on définit*
- *la norme utilisée : la norme infinie c'est-à-dire $\|(a_1, \ldots, a_n)\| = \max\{|a_i|\}$.*
- *$Cb^{(n)} = \{B(a,r); a \in \mathbb{Q}^n, r \in \mathbb{Q}, r > 0\}$. $Cb^{(n)}$ est l'ensemble des boules ouvertes rationnelles (pour la norme infinie, une boule a la forme d'un cube).*
- *$I^{(n)} : \prod_{i=0}^{n} \mathcal{D}om(\nu_{\mathbb{Q}}) \to Cb^{(n)}$. $I^{(n)}$ définie par*

$$I^{(n)}(v_1, \ldots, v_n, w) = B((\nu_{\mathbb{Q}}(v_1), \ldots, \nu_{\mathbb{Q}}(v_n)), \nu_{\mathbb{Q}}(w))$$

est une notation de $Cb^{(n)}$.

Nous avons vu dans la partie 1.1 qu'il est possible d'encoder 2 entiers ou plus à l'aide d'un seul de façon élémentaire. Notons $\langle ., \ldots, . \rangle : (\Sigma^*)^* \to \Sigma^*$ une fonction permettant cela. Notons également $v \lhd w$ si v est un sous-mot de w.

Définition 2.7 (Représentations de $\mathbb{R}$) *Soient les quatre représentations de $\mathbb{R}$ suivantes :*
- *ρ caractérisée par $\rho(p) = x \Leftrightarrow \{J \in Cb^{(1)}; x \in J\} = \{I^{(1)}(v,w); \langle v, w \rangle \lhd p\}$. En d'autres termes, un nombre réel est représenté par l'ensemble des intervalles ouverts rationnels qui le contiennent.*
- *$\rho_<$ caractérisée par $\rho(p) = x \Leftrightarrow \{a \in \mathbb{Q}; a < x\} = \{\nu_{\mathbb{Q}}(w); w \lhd p\}$. Un réel est représenté par l'ensemble des rationnels qui lui sont inférieurs.*
- *$\rho_>$: un réel est représenté par l'ensemble des rationnels qui lui sont supérieurs.*
- *la représentation de Cauchy : ρ_C caractérisée par*

$$\rho_C(p) = x \Leftrightarrow \begin{cases} \text{il existe } (w_i) \in \mathcal{D}om(\nu_{\mathbb{Q}})^{\mathbb{N}} \text{ telle que } p = w_0, w_1, \ldots \\ \text{et } \forall i, |\nu_{\mathbb{Q}}(w_i) - x| < 2^{-i}. \end{cases}$$

La représentation de Cauchy est équivalente à la représentation standard ρ ainsi qu'il est montré dans [Wei00, Lemme 4.1.6].

On dit de deux représentations ρ_1 et ρ_2 de M qu'elles sont équivalentes si elles peuvent se traduire l'une en l'autre, c'est-à-dire s'il existe une fonction calculable $f : \Sigma^\omega \to \Sigma^\omega$ telle que

$$\forall y \in \mathcal{D}om(\rho_1), \rho_1(y) = \rho_2(f(y))$$

et une fonction calculable $g : \Sigma^\omega \to \Sigma^\omega$ telle que

$$\forall y \in \mathcal{D}om(\rho_2), \rho_2(y) = \rho_1(g(y)).$$

Rappelons que la calculabilité des fonctions de Σ^ω est définie par 2.1.

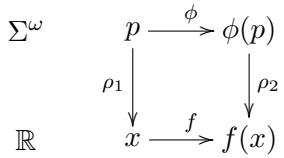

FIG. 2.1: ϕ est une (ρ_1, ρ_2)-représentation de f

Proposition 2.8 ([Wei00]) *Les représentations ρ et ρ_C sont équivalentes.*

Cela signifie que l'on peut utiliser indifféremment la représentation standard ρ ou la représentation de Cauchy.

Définition 2.9 (Nombre réel calculable) *Un nombre réel x est ρ-calculable si et seulement si il existe $g : \mathbb{N} \to \Sigma^*$ récursive telle que*

$$|x - \nu_{\mathbb{Q}}(g(n))| \leq 2^{-n}.$$

Il existe d'autres caractérisations équivalentes des nombres ρ-calculables par exemple la caractérisation suivante adaptée de [PER89].

Proposition 2.10 *Un nombre x est ρ-calculable si et seulement si il existe des fonctions récursives s, a, b, $e : \mathbb{N} \to \mathbb{N}$ telles que*

$$\forall N \in \mathbb{N}, \left| x - (-1)^{s(k)} \frac{a(k)}{b(k)} \right| \leq \frac{1}{2^N} \text{ pour tout } k \geq e(N).$$

Et nous pouvons maintenant définir les fonctions récursivement calculables. Plus précisément, nous définissons ce qu'est pour deux représentations ρ_1 et ρ_2 de $\mathbb{R}$ une fonction (ρ_1, ρ_2)-calculable.

Définition 2.11 (Fonction récursivement calculable) *Soient ρ_1 et ρ_2 des représentations de $\mathbb{R}$. Une fonction $f : \mathbb{R} \to \mathbb{R}$ est (ρ_1, ρ_2)-calculable s'il existe $\phi : \Sigma^\omega \to \Sigma^\omega$ récursive qui pour tout $x \in \mathcal{D}om(f)$ associe à tout p tel que $\rho_1(p) = x$ un $\phi(p)$ vérifiant $\rho_2(\phi(p)) = f(x)$.*

On appellera un tel ϕ une (ρ_1, ρ_2)-représentation de f.

La figure 2.1 présente schématiquement le rapport entre une fonction calculable et sa représentation : p est une ρ_1-représentation dans Σ^ω de $x \in \mathbb{R}$. Et $\phi(p)$ est une ρ_2-représentation de $f(x)$. On dit de ϕ qu'elle est une (ρ_1, ρ_2)-représentation de f ou encore qu'elle calcule f.

2.1.2 Définitions adaptées

Pour la suite, pour éviter de manipuler des notations trop lourdes, nous allons simplifier légèrement ces définitions, au prix parfois de légers abus de notations. Tout d'abord, nous allons confondre $\mathbb{N}$ et Σ^* et donc nous dispenser de $\nu_{\mathbb{N}}$.

Nous allons utiliser la représentation de Cauchy comme représentation des nombres réels.

Définition 2.12 (Représentation d'un réel) *Une suite d'entiers* $(x_i)_{i\in\mathbb{N}}$ représente un réel x *si* $\forall i \in \mathbb{N}, |\nu_{\mathbb{Q}}(x_i) - x| < 2^{-i}$. *On notera ceci* $(x_i) \rightsquigarrow x$.

Une fonction récursivement calculable sera alors une fonction (ρ_C, ρ_C)-calculable.

On définit les fonctions calculables sur les réels comme les fonctions qui à toute suite représentant un réel x associent *via* une fonction discrète calculable une suite représentant son image $f(x)$. On peut ainsi définir les fonctions élémentairement calculables, $\mathscr{E}_n$-calculables, ou plus généralement $\mathcal{C}$-calculable pour toute classe $\mathcal{C}$ de fonctions discrètes.

Définition 2.13 (Fonction récursivement calculable) *Une fonction* $f : \mathcal{D} \to \mathbb{R}$ *où* $\mathcal{D} \subset \mathbb{R}^k$ *est* récursivement calculable *(respectivement :* élémentairement calculable *;* $\mathscr{E}_n$-calculable *;* $\mathcal{C}$-calculable*) s'il existe* $\phi : (\mathbb{N}^k)^{\mathbb{N}} \times \mathbb{N} \to \mathbb{N}$ *récursive (resp. : élémentaire ;* $\mathscr{E}_n$ *;* $\mathcal{C}$*) telle que* $\forall \vec{x} \in \mathcal{D}$, $\forall X \in (\mathbb{N}^k)^{\mathbb{N}}$, $X \rightsquigarrow \vec{x} \Rightarrow \phi(X) \rightsquigarrow f(\vec{x})$.

On dira que ϕ *calcule* f.

Les fonctions élémentairement calculables ont été étudiées en particulier dans [Zho97].

On notera $\mathcal{R}ec(\mathbb{R})$ l'ensemble des fonctions récursivement calculables, $\mathscr{E}(\mathbb{R})$ l'ensemble des fonctions élémentairement calculables, et plus généralement $\mathcal{C}(\mathbb{R})$, pour $\mathcal{C}$ une classe de fonctions discrètes, l'ensemble des fonctions sur les réels f telles qu'il existe $\phi \in \mathcal{C}$ calculant f.

Définition 2.14 (Module de continuité) *Soit* $f \in \mathcal{R}ec(\mathbb{R})$*, on dit que* $m : \mathbb{N} \to \mathbb{N}$ *est un module de continuité de* f *si* $\forall x, y, \forall n \in \mathbb{N}, \|x - y\| < 2^{-m(n)} \Rightarrow \|f(x) - f(y)\| < 2^{-n}$.

On peut également définir une mesure de complexité sur les fonctions récursivement calculables.

Définition 2.15 (Complexité) *Soit une fonction récursivement calculable* $f : G \to \mathbb{R}$. *On dit que* $t : G \times \mathbb{N} \to \mathbb{N}$ *est une borne de complexité de* f *s'il existe* ϕ *calculant* f *telle que*

$$\forall x \in G, \forall n > 0, \forall X \rightsquigarrow x, t(x, n) \geq \{T((\phi(X))_n)\}$$

où $T((\phi(X))_n)$ *est le temps de calcul de* $(\phi(X))_n$.

Définition 2.16 (Complexité uniforme) $t' : \mathbb{N} \to \mathbb{N}$ *est une borne de complexité uniforme si* $\forall x \in G \cap [-2^n, 2^n], t(x, n) \leq t'(n)$

En d'autres termes, la complexité d'une fonction est le nombre $t(n)$ de bits de l'entrée qu'il est nécessaire de lire pour donner les n premiers bits de l'image.

2.1.3 Propriétés

Montrons quelques propriétés de l'analyse récursive. Tout d'abord quelques fonctions de base qui appartiennent à la classe des fonctions élémentairement calculables.

Proposition 2.17 (Th. 4.3.2 de [Wei00]) *Les fonctions* $+$, $-$, $\times$, $x \mapsto e^x$, sin, cos, $x \mapsto 1/x$ *sont élémentairement calculables.*

Démonstration : Il suffit de constater que la preuve de calculabilité de ces fonctions sur des intervalles fermés ne fait intervenir que des fonctions élémentaires □

Exemple 2.18 *La fonction* $+$ *est élémentairement calculable : soient deux suites* (a_n) *et* (b_n) *représentant respectivement les réels* a *et* b*. On a alors pour tout* $i \in \mathbb{N}$, $|\nu_{\mathbb{Q}}(a_i)-a| < 2^{-i}$ *et* $|\nu_{\mathbb{Q}}(b_i) - b| < 2^{-i}$*. Donc* $|(\nu_{\mathbb{Q}}(a_{i+1}) + \nu_{\mathbb{Q}}(b_{i+1})) - (a + b)| < 2 \times 2^{-(i+1)} = 2^{-i}$*. L'encodage étant une bijection élémentaire, on en tire de façon élémentaire une suite qui représente* $a + b$*.*

De par leur définition, les fonctions récursivement calculables sont continues.

Proposition 2.19 (Th. 4.3.1 de [Wei00]) *Toutes les fonctions récursivement calculables sont continues.*

Démonstration : Le principe de la preuve est le suivant : soit $f : \mathcal{G} \to \mathbb{R}$ une fonction élémentairement calculable, supposons-la calculée par ϕ. Soit $x \in \mathcal{G}$, soit $X \rightsquigarrow x$. Soit $\varepsilon > 0$. Il existe $n \in \mathbb{N}$ tel que $2^{-n} < \varepsilon$. Le calcul des n premiers termes de la suite $(\phi(X))$ se fait en un temps fini $t(X, n)$ qui majore le nombre d'éléments de la suite X qui ont été lus pour faire ce calcul. Tout $y \in \mathcal{G}$ tel que $\|x - y\| < 2^{-t(X,n)}$ peut être représenté par une suite Y commençant par les $t(X, n)$ premiers termes de X. Les n premiers termes de $\phi(Y)$ coïncident donc nécessairement avec les n premiers termes de $\phi(X)$. Cela signifie que $\|f(y) - f(x)\| \leq 2^{-n} < \varepsilon$.

Donc f est continue. □

On peut obtenir un résultat plus fort si l'on se place sur un compact : les fonctions élémentairement calculables sont uniformément continues et leur module de continuité est élémentaire.

Proposition 2.20 *Soit* $f \in \mathscr{E}(\mathbb{R})$ *définie sur un produit d'intervalles compacts. Alors,* f *possède un module de continuité dans* $\mathscr{E}(\mathbb{N})$*.*

Démonstration : La preuve de la proposition 2.19 montre que pour tout x, pour tout $\varepsilon > 0$, on peut exhiber un voisinage ouvert de x sur lequel $|f(y) - f(x)| < \varepsilon$.

Ici, nous voulons montrer que sur un compact, on peut borner uniformément le nombre de bits de l'entrée à voir en fonction de la précision voulue.

Notons K le compact où est définie f et ϕ la fonctionnelle qui calcule f. On a

$$\forall x \in K, \forall X \rightsquigarrow x, |\phi(X, n) - f(x)| < 2^{-n}.$$

Notons $T(X,n)$ le temps de calcul de $\phi(X,n)$. Comme $TIME(\mathscr{E}(\mathbb{N})) = \mathscr{E}(\mathbb{N})$, la fonction T est une fonctionnelle élémentaire. Comme si deux nombres sont proches à $2^{-T(X,n)}$ près, ils peuvent être représentés par une suite ayant les mêmes $T(X,n)$ premiers termes, on a

$$\forall x, y \in K, X \rightsquigarrow x, |x-y| < 2^{-T(X,n)} \Rightarrow |f(x)-f(y)| < 2^{-n}.$$

T étant élémentaire, on peut écrire qu'il existe e fonction itérée de l'exponentielle telle que $\forall x, n, X \rightsquigarrow x, T(X,n) \leq e(n,X)$. En réalité, seuls les $T(X,n)$ premiers éléments de X sont lisibles pour calculer $T(X,n)$, donc il est possible de dire que $T(X,n) \leq e(n, X_{T(X,n)})$. Or, il est possible de majorer par une constante $k2^n$ l'ensemble des représentants d'éléments de K à $T(X,n)$ près, il suffit pour cela de choisir une bonne représentation de K en constatant qu'il est possible de recouvrir K à l'aide de $k2^n$ boules de rayon $2^{-T(X,n)}$. On peut donc majorer $T(X,n)$ par $e(n,k)$. Cette fonction $e(n,k)$ est un module de continuité de f et est élémentaire (c'est une itérée de la fonction exponentielle). □

Proposition 2.21 *Soit f une fonction récursivement calculable. Soit t une borne de complexité uniforme pour f. Alors, t est un module de continuité pour f.*

Démonstration : Trivial. □

La fonction Γ d'Euler est récursivement calculable.

Définition 2.22 (Fonction Gamma d'Euler) *La fonction Γ est définie par la formule :*

$$\Gamma(x) = \int_0^{+\infty} t^{x-1} \mathrm{e}^{-t}\, \mathrm{d}t.$$

Cette fonction est définie et de classe $\mathscr{C}^\infty$ sur $\mathbb{R}^{>0}$ et à valeurs dans $\mathbb{R}$. Elle peut être vue comme une extension de la fonction factorielle aux réels : $\forall n \in \mathbb{N}, \Gamma(n+1) = n!$.

Proposition 2.23 *La fonction Γ d'Euler est récursivement calculable.*

Démonstration : Soit $y = \exp(x+1)$. Approchons $\Gamma(x)$ par $g(x,u) = \int_0^u t^{x-1}e^{-t}\,\mathrm{d}t$. $g(x,u)$ est récursivement calculable et tend vers $\Gamma(x)$ quand u tend vers $+\infty$. Calculons l'erreur faite en approchant $\Gamma(x)$ par $g(x,y)$.

$$\begin{array}{rcll} |\Gamma(x) - g(x,y)| & \leq & \int_y^{+\infty} t^{x-1}\mathrm{e}^{-t}\,\mathrm{d}t & \\ & \leq & \int_y^{+\infty} \frac{\mathrm{d}t}{t^2} & \text{si } y \geq \mathrm{e}^{x+1} \text{ car } t \geq \mathrm{e}^{x+1} \Rightarrow t^{x-1}\mathrm{e}^{-t} \leq 1/t^2 \\ & \leq & \frac{1}{y}. & \end{array}$$

On peut donc approcher $\Gamma(x)$ à la précision 2^{-i} pour tout i par $g(x,y)$ en choisissant $y \geq \max(\mathrm{e}^{x+1}, 2^i)$. □

2.2 Fonctions $\mathbb{R}$-récursives

Dans le monde discret, les fonctions récursives permettent de caractériser les fonctions calculables (par machine de Turing, par automate à deux piles, et en général par tous les modèles suffisamment puissants) de façon algébrique et à ce titre indépendamment des machines. Cristopher Moore a défini dans [Moo96], par analogie avec les fonctions discrètes récursives, une classe de fonctions dites $\mathbb{R}$-récursives. Par la suite, Manuel Campagnolo, Cristopher Moore et José Félix Costa ont défini [CMC00b, CMC02, Cam01] des sous-classes permettant de caractériser des fonctions plus raisonnables. José Félix Costa et Jerzy Mycka ont établi de nouvelles définitions pour certains des opérateurs utilisés [Myc03a, Myc03b, MC04, MC06, CM05] et des classes de fonctions qu'ils ont comparées avec les fonctions $\mathbb{R}$-récursives.

2.2.1 La classe $\mathscr{G}$

Dans cette section, nous rappelons les définitions et des résultats de Cristopher Moore dans [Moo96]. Certaines de ces définitions ne sont pas complètement bien définies. Nous présenterons ensuite des modifications dues à Jerzy Mycka et José Félix Costa dans le paragraphe 2.2.1.d, et à Manuel Campagnolo, Cristopher Moore et José Félix Costa dans la sous-section 2.2.2. Ces modifications ont entre autres buts celui de rendre plus propres ces définitions.

2.2.1.a Définitions

Nous reprenons la notation introduite par Peter Clote dans [Clo98] et présentée dans la définition 1.10 pour représenter de façon concise une classe de fonctions définie par clôture.

La classe $\mathscr{G}$ introduite par Moore dans [Moo96] est définie par imitation de la classe $\mathcal{R}ec(\mathbb{N})$. Nous allons maintenant définir les analogues continus des fonctions et opérateurs discrets qui composent la définition des fonctions récursives discrètes.

La constante 0 n'a pas lieu d'être modifiée. Le successeur va être remplacé par une fonction d'addition, en réalité, l'obtention de cette fonction se fera à l'aide des autres opérateurs, nous allons cependant avoir besoin de la constante 1.

Les projections restent des projections sur les réels, la composition elle non plus n'a pas besoin d'une définition spécifique pour être valide pour des fonctions sur les réels.

La minimisation va être remplacée par un opérateur de recherche de zéro sur les réels au lieu des entiers, ce qui pose deux problèmes : d'une part, les réels n'ont pas de minimum à partir duquel démarrer la recherche, il faut donc faire une recherche croissante et une recherche décroissante à partir de zéro et retourner la racine dont la valeur absolue est minimale ; d'autre part, l'ensemble des valeurs pour laquelle une fonction vaut 0 peut ne pas avoir de minimum, et alors en prenant la borne inférieure de cet ensemble, il est possible que le « plus petit » zéro ne soit pas une racine de la fonction. On peut penser par exemple à la fonction de Kronecker qui vaut 0 partout sauf en 0. La borne inférieure de l'ensemble de ses racines positives est cependant 0.

L'analogue réel de la récursion primitive choisi est un opérateur d'intégration. Rappelons que l'opérateur discret de récursion primitive fabrique une fonction $h : \mathbb{N}^{n+1} \to \mathbb{N}^m$ à partir de deux fonctions $f : \mathbb{N}^n \to \mathbb{N}^m$ et $g : \mathbb{N}^{n+m} \to \mathbb{N}^m$ définie par $h(\vec{x}, 0) = f(\vec{x})$ et $h(\vec{x}, k+1) = g(\vec{x}, h(\vec{x}, k))$. On pourrait alors écrire

$$\frac{h(\vec{x}, k+\varepsilon) - h(\vec{x}, k)}{\varepsilon} = \tilde{g}(\vec{x}, h(\vec{x}, k))$$

avec $\varepsilon = 1$ et $\tilde{g}(\vec{x}, \vec{y}) = g(\vec{x}, \vec{y}) - \vec{y}$.

L'analogie qui existe donc entre le schéma de récursion primitive et un problème de Cauchy a incité Cristopher Moore à proposer un opérateur d'intégration qui prend deux fonctions f et g en arguments et renvoie la fonction h solution du problème de Cauchy

$$\left\{ \begin{array}{rcl} h(\vec{x}, 0) & = & f(\vec{x}) \\ \frac{\partial h}{\partial t}(\vec{x}, t) & = & g(\vec{x}, h(\vec{x}, t)). \end{array} \right.$$

Définition 2.24 ($\mathscr{G}$) *On définit la classe $\mathscr{G}$ comme suit :*

$$\mathscr{G} = \left[0, +, U; \mathrm{COMP}, \int, \mu_{\mathbb{R}}\right]$$

où

- *U représente les projections : $\forall \vec{x} = (x_1, \ldots, x_n), i \in \{1, .., n\}, U_i^n(\vec{x}) = x_i$;*
- *$\int$: étant donnés f et g, $h = \int(f, g)$ est la solution de l'équation différentielle*

$$\left\{ \begin{array}{rcl} h(\vec{x}, 0) & = & f(\vec{x}) \\ \frac{\partial h}{\partial t}(\vec{x}, t) & = & g(\vec{x}, h(\vec{x}, t)) \end{array} \right.$$

- *$\mu_{\mathbb{R}}$: étant donnée une fonction $f : \mathbb{R}^{n+1} \to \mathbb{R}$, posons $y_+(\vec{x}) = \inf\{t > 0; f(\vec{x}, t) = 0\}$ et $y_-(\vec{x}) = \sup\{t < 0; f(\vec{x}, t) = 0\}$. Si y_+ ou y_- existe, on définit*

$$\mu_{\mathbb{R}}(f)(\vec{x}) = \left\{ \begin{array}{l} y_-(\vec{x}) \text{ si } |y_-(\vec{x})| \leq y_+(\vec{x}) \\ y_+(\vec{x}) \text{ sinon} \end{array} \right. .$$

Remarquons que l'opérateur $\mu_{\mathbb{R}}$ (opérateur de recherche de zéro réel ou opérateur de minimisation) n'est pas toujours défini (de même que l'opérateur de minimisation discret) : s'il n'existe aucun t tel que $f(\vec{x}, t) = 0$, $\mu_{\mathbb{R}}(f)(\vec{x})$ n'est pas défini.

Remarquons également que dans la définition $\int$, peuvent se présenter des cas désagréables de solution multiples, et dans ce cas, rien ne précise comment se comporte cet opérateur.

2.2.1.b Propriétés

Cette classe contient les fonctions usuelles $+$, $\times$, $x \mapsto 1/x$, ainsi que les fonctions sin, cos, exp, $|x|$

Proposition 2.25 ([Moo96]) *$+$, $\times$, $x \mapsto 1/x$ appartiennent à $\mathscr{G}$.*

Démonstration : Exhibons des constructions des fonctions f_+, $f_\times$ et inv telles que $f_+(x,y) = x+y$, $f_\times(x,y) = x-y$, $inv(x) = 1/x$:

- f_+ est définie par $f_+ = \int(x \mapsto x, (x,h) \mapsto 1)$ en effet, $f_+(x,0) = x$ et $\frac{\partial f_+}{\partial y}(x,y) = 1$;
- $f_\times$ est définie par $f_\times = \int(x \mapsto 0, (x,h) \mapsto x)$. On a bien ainsi $f_\times(x,0) = 0$ et $\frac{\partial f_\times}{\partial y}(x,y) = x$;
- pour définir inv, nous avons tout d'abord besoin de -1. -1 est la racine de la fonction $x \mapsto x+1$ qui est obtenue en composant f_+ et 1.

  $$-1 = \mu_{\mathbb{R}}(t \mapsto t+1).$$

  Notons que -1 est une fonction d'arité nulle. Posons maintenant

  $$g = \int(1, h \mapsto f_\times(-1, f_\times(h,h))).$$

  g va donc vérifier $g(0) = 1$ et $\frac{\partial g}{\partial y}(y) = -g(y)^2$. On a alors

  $$inv(x) = \mathrm{COMP}(g, x \mapsto x + (-1)).$$

  Remarquons que inv n'étant pas définie en 0 et l'opérateur $\int$ réclamant une valeur en 0, il était nécessaire d'utiliser une fonction intermédiaire. La fonction inv ainsi construite est définie uniquement sur $]0, +\infty[$.

Les fonctions d'addition, de multiplication et d'inversion sont donc dans $\mathscr{G}$. □

Proposition 2.26 ([Moo96]) *exp, cos, sin, $|x|$ appartiennent à $\mathscr{G}$*

Démonstration : La fonction exp est connue pour vérifier $\exp = \frac{\partial \exp}{\partial x}$. On peut donc définir exp comme $\int(f,g)$ avec $f() = 1$ et $g(y) = y$.

Les fonctions trigonométriques cos et sin se définissent simultanément par

$$\begin{pmatrix}\sin\\ \cos\end{pmatrix}(0) = \begin{pmatrix}0\\ 1\end{pmatrix} \text{ et } \frac{\partial\begin{pmatrix}\sin\\ \cos\end{pmatrix}}{\partial x}(x) = \begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix} \times \begin{pmatrix}\sin\\ \cos\end{pmatrix}(y)$$

La définition de $|x|$ se fait à l'aide de l'opérateur $\mu_{\mathbb{R}}$: posons $f(x,y) = x^2 - y^2$, alors, $|x| = -\mu_{\mathbb{R}}(f)(x)$. □

Proposition 2.27 *$\mathscr{G}$ est stable par passage à la primitive.*

Démonstration : Ce résultat est trivial : pour $f \in \mathscr{G}$, si 0 appartient à l'ensemble de définition de f, il est possible de définir la primitive de f valant 0 en 0 à l'aide de l'opérateur $\int$.

□

On peut en déduire que $\mathscr{G}$ contient également la fonction ln : primitive de *inv* valant 0 en 1.

Cette classe contient les fonctions caractéristiques d'un certain nombre d'ensembles classiques, par exemple $\mathbb{Q}$, ou celles d'ensembles non-décidables.

Proposition 2.28 ([Moo96]) *La fonction caractéristique des rationnels :*

$$\chi_{\mathbb{Q}} : \begin{cases} \mathbb{R} & \to & \mathbb{R} \\ x \in \mathbb{Q} & \mapsto & 1 \\ x \notin \mathbb{Q} & \mapsto & 0 \end{cases}$$

appartient à $\mathscr{G}$.

Démonstration : Pour montrer que $\chi_{\mathbb{Q}}$ est $\mathbb{R}$-récursive, nous allons définir $\chi_{\mathbb{Z}}$, fonction caractéristique des entiers relatifs. Définissons tout d'abord une fonction dite fonction de Kronecker δ qui vaut 1 en 0 et vaut 0 partout ailleurs.

$$\delta(x) = 1 - \mu_y((x^2 + y^2)(y - 1))$$

En effet, quoi qu'il arrive, $y = 1$ est une racine de l'expression $(x^2 + y^2)(y - 1)$, et $y = 0$ en est une racine si et seulement si $x = 0$. Comme il n'y a pas d'autres racines, μ retourne 1 si $x = 0$ et 0 si $x \neq 0$. Et finalement on obtient bien $\delta(x) = 1$ si $x = 0$ et $\delta(x) = 0$ sinon.

On peut caractériser les entiers relatifs par le fait qu'ils sont les racines de la fonction $x \mapsto \sin(\pi x)$. En d'autres termes,

$$\sin(\pi x) = 0 \Leftrightarrow x \in \mathbb{Z}.$$

On peut donc construire $\chi_{\mathbb{Z}}$ comme suit :

$$\chi_{\mathbb{Z}}(x) = \delta(\sin(\pi x)).$$

On peut caractériser les rationnels par le fait qu'il existe un entier naturel q (le dénominateur) dont le produit avec le nombre en question est un entier relatif (le numérateur).

$$x \in \mathbb{Q} \quad \Leftrightarrow \quad \exists q \in \mathbb{N}^*; xq \in \mathbb{Z}$$

Posons

$$g(y) = 1 - \frac{1}{y}.$$

La fonction g est définie sur $[1, +\infty[$ (et $g^{-1}(x) = \frac{1}{1-y}$ définie sur $[0, 1[$). Considérons l'expression

$$(\chi_{\mathbb{Z}}(g^{-1}(y))\chi_{\mathbb{Z}}(xg^{-1}(y)) - 1)(y - 1).$$

Elle possède une racine $y = 1$ (en convenant que $0 \times \infty = 0$) et une racine $0 \leqslant y < 1$ si et seulement si x est rationnel.

Donc $\chi_{\mathbb{Q}}(x) = 1 - \delta(\mu_y[(\chi_{\mathbb{Z}}(g^{-1}(|y|))\chi_{\mathbb{Z}}(xg^{-1}(|y|)) - 1)(y-1)] - 1)$. □

Ce résultat est déconcertant : la fonction caractéristique des rationnels est discontinue en tout point, ce qui la rend difficile à appréhender.

Nous allons maintenant montrer qu'il est possible d'encoder l'itération dans une fonction $\mathbb{R}$-récursive, ce résultat va nous permettre de montrer qu'il est possible de résoudre le problème de la halte des machines de Turing par une fonction $\mathbb{R}$-récursive.

Lemme 2.29 *Soit $f \in \mathscr{G}$. Nommons F la fonction itérée de f, c'est-à-dire la fonction telle que $F(t, \vec{x}_0) = f^{[t]}(\vec{x}_0)$. F appartient à $\mathscr{G}$.*

Démonstration : On a déjà montré dans la proposition 2.26 que la fonction valeur absolue est dans $\mathscr{G}$ et que la fonction de Kronecker δ aussi. Montrons maintenant que la fonction signe sgn appartient aussi à $\mathscr{G}$: soit

$$g : x, y \mapsto yx - |x|,$$

cette fonction est dans $\mathscr{G}$ et

$$sgn = \mu_{\mathbb{R}}(g).$$

Enfin, la fonction θ qui vaut 0 pour un argument strictement négatif et 1 pour un argument positif peut être définie comme

$$\theta(x) = \frac{sgn(x) + \delta(x) + 1}{2}.$$

On définit des fonctions d'horloge r et s comme suit :

$$s(t) = \theta(\sin(\pi t))$$

$$r(t) = \int (0, t \mapsto 2s(t) - 1)$$

s est une fonction en créneaux de période 2. Elle vaut 1 sur les intervalles de la forme $[2n, 2n+1]$ et 0 sur les $]2n+1, 2n+2[$. r est une fonction en triangle 2-périodique valant 0 en les entiers pairs, 1 en les entiers impairs. Définissons y_1 et y_2 par les équations différentielles suivantes (qui peuvent être encodées par un opérateur d'intégration) :

$$\begin{aligned}
y_1(0) &= \vec{x}_0 \\
y_2(0) &= \vec{x}_0 \\
\frac{\partial y_1}{\partial t}(t) &= (f(y_2(t)) - y_2(t))s(t) \\
\frac{\partial y_2}{\partial t}(t) &= \frac{(y_1(t) - y_2(t))}{r(t)}(1 - s(t))
\end{aligned}$$

Notons que la méthode utilisée est similaire à celle de Branicky [Bra95] que nous allons de nouveau utiliser dans le cadre de la proposition 5.7.

En posant $F(n, \vec{x}_0) = y_2(\lfloor 2n \rfloor)$, on obtient bien la fonction d'itération voulue. □

Proposition 2.30 ([Moo96]) *La fonction χ_{halt} définie par*

$$\forall n, p \in \mathbb{N}, \chi_{halt}(\langle n, p\rangle) = \begin{cases} 1 & \text{si } \phi_n \text{ s'arrête sur l'entrée } p \\ 0 & \text{sinon} \end{cases}$$

(où ϕ_n est définie comme dans la proposition 1.21) appartient à $\mathscr{G}$.

Démonstration : En suivant [KM99], on voit qu'il est possible de représenter une machine de Turing comme une fonction de $[0,1]^2$ dans $[0,1]^2$. Plus simplement, on peut intuitivement encoder le ruban de la machine par une paire d'entiers (les parties respectivement à gauche et à droite de la tête de lecture) représentant les parties décimales des coordonnées d'un point de $[0,1]^2$ et l'état par la troisième coordonnée $\frac{1}{q+1}$ où q est l'état.

La fonction de transition de la machine de Turing est alors une fonction $[0,1]^3 \to [0,1]^3$ affine par morceaux. Le décalage du ruban vers la gauche est par exemple simulé par $(x, y, q) \mapsto (Bx \ mod \ 1, \frac{Y}{B} + \lfloor \frac{Bx}{B} \rfloor, q')$ où B est la base de calcul.

En considérant une machine de Turing universelle et la fonction f_u définie comme proposé ci-dessus, en posant F_u la fonction itérée de f_u, savoir si ϕ_n termine sur l'entrée p consiste à savoir s'il existe t tel que la troisième composante de $F_u(t, (0, \langle n, p\rangle, \frac{1}{q_i}))$ soit l'état final q_f. Notons $G(t) = F_u(t, (0, \langle n, p\rangle, \frac{1}{q_i}))$ et u_3 la projection suivant la troisième composante.

$$\chi_{halt}(\langle n, p\rangle) = 1 - \delta\left(\mu_{\mathbb{R}}\left[z \mapsto \left(u_3(G(\tan(z))) - \frac{1}{q_f}\right)(z - \pi/2)\right] - \pi/2\right).$$

En effet, s'il existe un t pour lequel l'état final est atteint, $u_3(G(\tan(z))) - \frac{1}{q_f}$ s'annulera pour un $z < \pi/2$. S'il n'en existe pas, le plus petit z annulant l'expression sera $\pi/2$. □

On peut remarquer que la méthode de compression du temps utilisée (l'utilisation de la fonction tan pour couvrir l'ensemble des réels) est une incarnation du phénomène de Zénon décrit dans le paragraphe 1.3.5.a.

On a donc dans $\mathcal{G}$ une fonction discontinue en tout point ($\chi_{\mathbb{Q}}$) donc difficile à manipuler dans le cadre de l'analyse, et une fonction qui encode le problème de la halte des machines de Turing. Ces fonctions sont $\mathbb{R}$-récursives mais elles ne sont pas récursivement calculables au sens de l'analyse récursive.

Il apparaît que l'opérateur $\mu_{\mathbb{R}}$ possède une puissance excessive. Il permet de réaliser une opération qui ne semble pas physiquement réalisable. Il s'agit d'un opérateur qui fait apparaître le phénomène de zénonisme discuté dans le paragraphe 1.3.5.

Il est possible de construire une hiérarchie suivant le nombre de $\mu_{\mathbb{R}}$ imbriqués. Cela permet de voir à quel niveau des fonctions indésirables apparaissent.

2.2.1.c La $\mu_{\mathbb{R}}$-hiérarchie

Définissons le $\mu_{\mathbb{R}}$-degré d'une fonction appartenant à $\mathscr{G}$ comme étant le nombre minimal de $\mu_{\mathbb{R}}$ imbriqués nécessaires pour la définir.

Définition 2.31 ($\mu_{\mathbb{R}}$-degré) *On note $M_{x_i}(f)$ le nombre de $\mu_{\mathbb{R}}$ imbriqués relativement à la variable x_i d'une fonction f par récurrence structurelle sur la définition considérée de f.*

- $M_x(0) = M_x(1) = 0$
- $M_x(f(g_1, g_2, \ldots, g_k)) = \max_j(M_{x_j}(f) + M_x g_j)$
- $M_x(\int(f, g)) = \max(M_x(f), M_x(g), M_h(g))$
- $M_x(\mu_{\mathbb{R}}(f)) = 1 + \max(M_x(f), M_y(f))$

Le $\mu_{\mathbb{R}}$-degré de f est alors le minimum sur les définitions possibles de f de

$$\max_i M_{x_i}(f).$$

L'affirmation suivante issue de [Moo96] n'est pas vraie si on ne précise pas la notion de solution d'équation différentielle : on peut en effet obtenir des fonctions qui ne sont pas de classe $\mathcal{C}^1$ si on accepte les différentes solutions obtenues lors de la résolution d'un problème de Cauchy où apparaissent plusieurs solutions.

Affirmation 2.32 ([Moo96]) *Les fonctions de $\mu_{\mathbb{R}}$-degré 0 sont analytiques.*

On peut déduire de cette affirmation que la fonction valeur absolue n'est pas de $\mu_{\mathbb{R}}$-degré 0 mais de $\mu_{\mathbb{R}}$-degré 1.

Proposition 2.33 ([Moo96]) *Pour tout $j \in \mathbb{N}$, il existe une fonction de $\mu_{\mathbb{R}}$-degré j.*

Cela signifie que l'on peut établir une hiérarchie stricte basée sur le $\mu_{\mathbb{R}}$-degré.

2.2.1.d Définitions alternatives

L'opérateur de recherche de zéro $\mu_{\mathbb{R}}$ est certes l'adaptation de l'opérateur classique discret de recherche de zéro, mais il n'est pas très naturel en tant qu'opérateur sur les réels. Pour pallier ce problème entre autres, Jerzy Mycka et José Félix Costa ont proposé dans [Myc03a, Myc03b, MC05] des définitions de classes de fonctions sans cet opérateur $\mu_{\mathbb{R}}$ mais avec des opérateurs de limite.

Définition 2.34 (Opérateurs de limite) *Soit $f : \mathbb{R}^{k+1} \to \mathbb{R}^n$. Soit $\vec{x} \in \mathbb{R}^k$. On définit les fonctions h, h_i, h_s de la façon suivante :*

- $h(\vec{x}) = \lim_{y \to +\infty} f(\vec{x}, y)$
- $h_i(\vec{x}) = \liminf_{y \to +\infty} f(\vec{x}, y)$
- $h_s(\vec{x}) = \limsup_{y \to +\infty} f(\vec{x}, y)$

On notera $h = Lim(f)$, $h_i = Liminf(f)$ et $h_s = Limsup(f)$.

Considérons la classe de fonctions $\mathscr{G}_l$ dérivée de la classe $\mathscr{G}$ en remplaçant l'opérateur $\mu_{\mathbb{R}}$ par les opérateurs $Liminf$ et $Limsup$ et en y ajoutant -1.

Définition 2.35 ($\mathscr{G}_l$)

$$\mathscr{G}_l = \left[0, 1, -1, U; \text{COMP}, \int, Liminf, Limsup\right].$$

Notons que la constante -1 a été ajoutée à la classe, en effet, elle peut être obtenue à l'aide de l'opérateur $\mu_{\mathbb{R}}$, mais les opérateurs de limites ou d'intégration ne permettent pas de changement de signe.

Cette classe $\mathscr{G}_l$ est en fait équivalente à la classe $\mathscr{G}$. Les opérateurs de limite et de recherche de zéro sont donc de puissance similaire.

De la même façon que pour l'opérateur $\mu_{\mathbb{R}}$, il est possible de définir une hiérarchie dans la classe $\mathscr{G}_l$ suivant le nombre d'opérateurs de limite imbriqués nécessaires pour définir les fonctions. Cependant, on ne sait si cette hiérarchie est stricte : on sait que le niveau 2 contient strictement le niveau 1 qui contient strictement le niveau 0, mais nous n'avons pas connaissance de preuve qu'il existe des fonctions dans un niveau strictement supérieur à 2 qui ne soient pas incluses dans le niveau 2 de cette hiérarchie.

La constante de Chaitin Ω est un exemple de nombre non-calculable. Elle est définie comme étant la somme $\sum_{p \in P} 2^{-|p|}$ où P est l'ensemble des programmes sans entrées terminant ayant la propriété qu'aucun préfixe d'un programme de P n'est dans P et $|p|$ est la taille du programme. Calculer cette constante implique de résoudre le problème de la halte.

Proposition 2.36 ([MC05]) *La constante Ω de Chaitin appartient à $\mathscr{G}_l$.*

2.2.2 La classe $\mathscr{L}$

2.2.2.a Définitions

On a vu que la puissance de l'opérateur $\mu_{\mathbb{R}}$ le rendait peu raisonnable en termes de réalisation physique. L'opérateur $\int$ quant à lui n'est pas défini de façon parfaitement formelle. En effet, rien ne précise si la fonction que l'on intègre doit être intégrable. Enfin, la classe $\mathscr{G}$ contient des fonctions déraisonnables à cause de ces opérateurs. Remplacer ces opérateurs par d'autres plus raisonnables et mieux définis est donc un objectif utile. Pour cela, considérons les opérateurs et classes définis par Manuel Campagnolo *et al.* dans [CMC00a, CMC00b, Cam01].

La fonction de Heaviside est une fonction seuil qui différencie les nombres négatifs des nombres positifs.

Définition 2.37 (Fonction de Heaviside) *Soit θ la fonction de Heaviside définie par*

$$\theta : \begin{array}{ccc} \mathbb{R} & \to & \mathbb{R} \\ x & \mapsto & \begin{cases} 1 \ si \ x \geq 0 \\ 0 \ sinon \end{cases} \end{array}.$$

On définit des versions lissées de cette fonction de la façon suivante :

$$\theta_i : \begin{array}{ccc} \mathbb{R} & \to & \mathbb{R} \\ x & \mapsto & x^i\theta(x) \end{array} .$$

Pour tout $i \in \mathbb{N}$, la fonction θ_{i+1} est de classe $\mathscr{C}^i$

Définition 2.38 (Opérateur d'intégration linéaire LI**)** *Soient* $f : \mathbb{R}^n \to \mathbb{R}$ *et* $g : \mathbb{R}^{n+1} \to \mathbb{R}$*. On définit* $h = \mathrm{LI}(f, g)$ *comme la solution sur le plus grand intervalle contenant* 0 *du problème de Cauchy* $\begin{cases} h(\vec{x}, 0) = f(\vec{x}) \\ \frac{\partial h}{\partial t}(\vec{x}, t) = g(\vec{x}, t)h(\vec{x}, t) \end{cases}$.

Par la suite, on emploiera θ_3 dans les définitions de classes de fonctions. On peut noter que l'on aurait pu choisir n'importe quel θ_k pour $k \geq 3$.

Définition 2.39 ($\mathscr{L}$)

$$\mathscr{L} = [0, 1, -1, \pi, \theta_3, U; \mathrm{COMP}, \mathrm{LI}]$$

Remarquons qu'outre -1 déjà ajouté aux fonctions de base pour la définition de $\mathscr{G}_l$, a été utilisée pour $\mathscr{L}$ la constante π. En effet, les fonctions cos et sin apparaissent naturellement dans la classe en utilisant l'opérateur d'intégration (ici linéaire), mais leur période 2π ne semble pas simple à obtenir sans utiliser de limite ou de minimisation. De façon informelle, on peut penser au problème de la quadrature du cercle : pour obtenir le nombre π, il est nécessaire d'utiliser un opérateur de limite.

Avant de montrer les propriétés intéressantes que possède cette classe $\mathscr{L}$, définissons la notation DP.

Définition 2.40 (DP : Discrete Part) *Étant donnée une classe de fonctions* $\mathcal{F}$*, on définit sa partie discrète de la façon suivante :*

$$DP(\mathcal{F}) = \{f_{|\mathbb{N}}; f \in \mathcal{F} \wedge f(\mathbb{N}) \subset \mathbb{N}\}.$$

Où $f_{|\mathbb{N}}$ *est la restriction de* f *aux entiers naturels.*

2.2.2.b Propriétés

Toutes les fonctions de la classe $\mathscr{L}$ sont de classe $\mathscr{C}^2$. En fait, si on n'avait pas inclus la fonction θ_3 dans la classe, toutes les fonctions seraient analytiques. En effet, l'opérateur d'intégration linéaire que nous avons défini préserve l'appartenance à $\mathscr{C}^i$.

Proposition 2.41 *L'opérateur* LI *préserve l'analyticité et l'appartenance à* $\mathscr{C}^i$

Démonstration : Procédons par récurrence sur i. Soient $f : \mathbb{R}^{n+1} \to \mathbb{R}$ et $g : \mathbb{R}^n \to \mathbb{R}$ deux fonctions de classe $\mathscr{C}^0$. Soit $h = \mathrm{LI}(f, g)$. Par construction, $\vec{x} \mapsto h(\vec{x}, 0)$ est continue et h est dérivable par rapport à sa $n+1$-ième variable donc en particulier continue.

Pour $i+1$, on peut calculer la i-différentielle de h : $D_i(h) = \frac{\partial D_{i-1}(h)}{\partial x_i} + \frac{\partial D_{i-1}(h)}{\partial t} = \frac{\partial D_{i-1}(h)}{\partial x_i} + D_{i-1}(\frac{\partial h}{\partial t})$. On constate que cette différentielle est continue (par application du cas $i = 0$). Donc h est de classe $\mathscr{C}^i$.

L'analyticité est également préservée par cet opérateur d'intégration, en effet si le développement de Taylor de la dérivée existe, il est égal à la dérivée du développement de Taylor. La préservation de $\mathscr{C}^\infty$ et l'analyticité de f et g permettent de conclure que h est également analytique. □

Les fonctions de base sont clairement de classe $\mathscr{C}^2$. L'opérateur de composition préserve la continuité, la différentiabilité et l'analyticité, donc on a bien le résultat :

Proposition 2.42 *Toutes les fonctions de $\mathscr{L}$ sont de classe $\mathscr{C}^2$.*

Cette classe a des propriétés intéressantes qui permettent de relier $\mathscr{L}$ avec $\mathscr{E}(\mathbb{R})$ ou $\mathscr{E}(\mathbb{N})$.

Proposition 2.43 ([CMC00b, CMC02, Cam01])

$$\mathscr{L} \subseteq \mathscr{E}(\mathbb{R})$$

Proposition 2.44 ([CMC00b, CMC02, Cam01])

$$\mathscr{E}(\mathbb{N}) \subseteq DP(\mathscr{L})$$

2.2.3 Les classes $\mathscr{L}_n$

Nous avons présenté dans la section 1.1.4 les classes $\mathscr{E}_n$ de la hiérarchie de Grzegorczyk. Elles sont basées sur des fonctions E_n qui ajoutent à chaque classe un degré d'itération.

2.2.3.a Définitions

Définissons des fonctions $\bar{E}_n$ qui étendent de façon $\mathscr{C}^2$ les fonctions E_n aux réels. Par exemple $\bar{E}_2(x) = 2^x$, et pour $n \geq 3$,

$$\bar{E}_n(x) = E_n(\lfloor x \rfloor) + \frac{1}{2}\sin(2\pi(x - \lfloor x \rfloor) + \pi)\left(E_n(\lfloor x + 1 \rfloor) - E_n(\lfloor x \rfloor)\right).$$

On pourrait également obtenir de telles fonctions $\bar{E}_n$ à l'aide d'intégrations non linéaires. Les fonctions définies par $\bar{E}'_{n+1} = \bar{E}_n \circ \bar{E}_n$ feraient aussi l'affaire (même si ce ne sont pas exactement des extensions réelles des fonctions E_n).

Définition 2.45 ($\mathscr{L}_n$)

$$\mathscr{L}_n = \left[0, 1, -1, \pi, \theta_3, U, \bar{E}_{n-1}; \mathrm{COMP}, \mathrm{LI}\right]$$

2.2.3.b Propriétés

Les résultats reliant $\mathscr{L}$ à $\mathscr{E}(\mathbb{N})$ se transposent aux classes $\mathscr{L}_n$ et aux classes $\mathscr{E}_n(\mathbb{N})$ de la hiérarchie de Grzegorczyk.

Proposition 2.46 ([Cam01])

$$\mathscr{L}_n \subseteq \mathscr{E}_n(\mathbb{R})$$

Proposition 2.47 ([Cam01])

$$\mathscr{E}_n(\mathbb{N}) \subseteq DP(\mathscr{L}_n)$$

2.3 General Purpose Analog Computer

Le troisième modèle que nous étudions est le General Purpose Analog Computer, que nous nommerons GPAC. Le GPAC est un modèle que Claude Shannon a présenté dans [Sha41]. Il s'agit d'un modèle mathématique d'une machine analogique : l'analyseur différentiel. L'analyseur différentiel a été initié par Lord Kelvin [TLK76] dans les années 1870 qui eu l'idée d'utiliser des intégrateurs pour résoudre des équations différentielles. Dans les années 1930, un analyseur différentiel fut conçu au Massachusetts Institute of Technology par une équipe menée par Vannevar Bush [Bus31], puis fut utilisé pour résoudre des équations différentielles intervenant dans des calculs balistiques ou d'architecture aéronautique.

2.3.1 Définitions

L'idée de ce modèle peut être expliquée en termes électroniques. On dispose de plusieurs sortes de composants : intégrateurs, additionneurs, multiplicateurs, constantes ; que l'on peut connecter les uns aux autres, y compris en faisant des boucles de rétroaction (*feedback connections* en anglais).

Définition 2.48 (composants du GPAC) *Voici les 5 composants pouvant apparaître dans un* GPAC *:*

- *intégrateur : composant doté de 2 entrées et une sortie, et réglable via 2 constantes. Pour des entrées f et g, et des constantes a et t_0, la sortie h est définie par*

$$h = \int_{t_0}^{t} f(u)\,\mathrm{d}g(u) + a;$$

- *multiplicateur par un scalaire : composant à une entrée f, une constante λ et une sortie λf ;*
- *additionneur : composant doté de 2 entrées f et g et d'une sortie $f + g$;*
- *multiplicateur : composant à 2 entrées f et g et une sortie $f \times g$;*

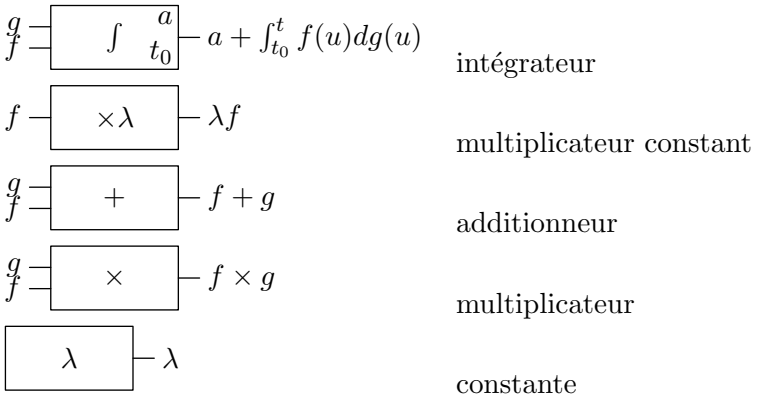

FIG. 2.2: Représentation des composants d'un GPAC.

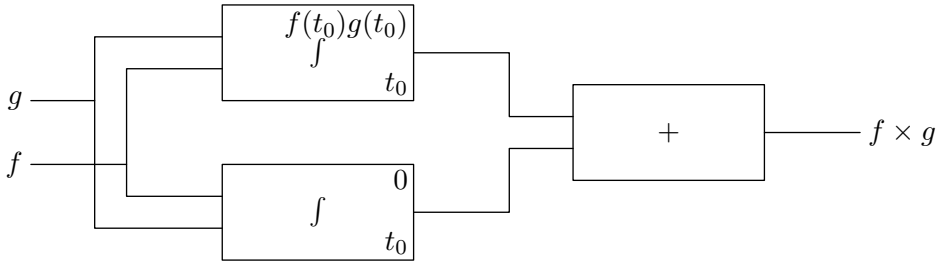

FIG. 2.3: Simulation d'un multiplicateur par des intégrateurs

– *constante : composant sans entrée doté d'une sortie* λ.

Ces composants peuvent être représentés à la manière de composants électroniques comme dans la figure 2.2.

On peut remarquer que le composant multiplication peut être simulé à l'aide de composants d'intégration en utilisant la formule d'intégration par partie :

$$f(t)g(t) = \int_{t_0}^{t} f(u)\,\mathrm{d}g(u) + \int_{t_0}^{t} g(u)\,\mathrm{d}f(u) + f(t_0)g(t_0).$$

Et donc le circuit représenté en figure 2.3 remplace un multiplicateur. Ce remplacement ne fonctionne cependant que si les deux intégrales sont définies. Cette méthode est donc parfois difficile à utiliser.

Exemple 2.49 *La fonction* exp *est très facilement définissable à l'aide d'un* GPAC. *Il suffit d'utiliser le fait que* $\exp' = \exp$ *et* $\exp(0) = 1$ *et l'on peut construire le circuit représenté en figure 2.4*

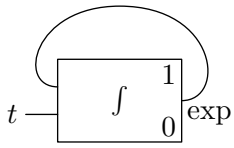

FIG. 2.4: Circuit calculant exp

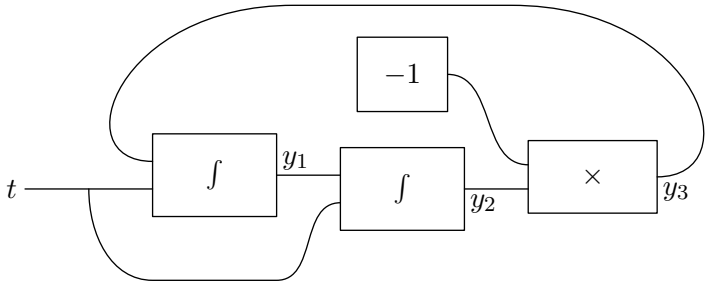

FIG. 2.5: Circuit calculant cos et sin.

Exemple 2.50 *Les fonctions* cos *et* sin *peuvent être calculées par un* GPAC *ainsi que l'illustre la figure 2.5. En posant comme conditions initiales (les constantes des intégrateurs)* $y_1(0) = 1$ *et* $y_2(0) = 0$*, on obtient* $y_1 = \cos$*,* $y_2 = \sin$*. En effet, le premier intégrateur impose* $y_1' = y_3$*, le deuxième intégrateur impose* $y_2' = y_1$ *et le multiplicateur dit que* $y_3 = -y_2$*. On a donc*

$$\begin{cases} y_1' &= -y_2 \\ y_2' &= y_1 \\ y_1(0) &= 1 \\ y_2(0) &= 0 \end{cases}$$

2.3.2 Propriétés

Dans l'article [Sha41] présentant cette modélisation des analyseurs différentiels, Claude Shannon prétend que les GPAC calculent exactement les fonctions différentiellement algébriques. Cette preuve n'était pas complètement valide, elle a été ensuite reprise, corrigée dans [PE74]. Puis cette deuxième preuve n'étant pas tout à fait satisfaisante, elle a de nouveau été corrigée dans [LR87].

Définition 2.51 *Une fonction différentiellement algébrique est une fonction solution d'une équation différentielle de la forme*

$$P(x, y, y', \ldots, y^{(n)}) = 0$$

où P est un polynôme en ses $n+2$ variables, c'est à dire une fonction de la forme

$$X_{-1}, X_0, \ldots, X_n \mapsto \sum_{\substack{i_{-1}+\cdots+i_n \leq p \\ p=0}}^{q} \left(\alpha_{i_{-1}\ldots i_n} \prod_{j=-1}^{n} X_{i_j}^{\beta_{i_j}} \right).$$

Même si cette preuve n'était pas parfaite, on peut déduire des arguments qui y sont utilisés la proposition suivante :

Proposition 2.52 *Les fonctions générées par* GPAC *sont analytiques.*

De cette proposition, on peut déduire que la fonction Γ d'Euler n'est pas générée par un GPAC : celle-ci n'est pas différentiellement algébrique.

Proposition 2.53 *La fonction Γ d'Euler ne peut être générée par un* GPAC.

2.3.3 Définitions adaptées

Suivant [Gra04], de nouvelles définitions plus propres de GPAC peuvent être posées. Les fonctions générées par GPAC sont alors définies comme les fonctions solutions de systèmes d'équations différentielles polynomiaux. Il est possible de lier ces systèmes avec des GPAC formant un circuit polynomial comme défini dans [GC03]. La relation avec les composants élémentaires et les circuits continue donc d'être valide.

Définition 2.54 (Systèmes d'équations différentielles polynomiaux) *On appelle système d'équations différentielles polynomial un problème de Cauchy de la forme*

$$\begin{cases} y' & = p(t, y) \\ y(t_0) & = y_0 \end{cases}$$

où y un vecteur et p est un vecteur de polynômes en les composantes de y et en t.

Les composantes y_i des solutions d'un tel système seront qualifiées de fonctions pCp (pour problème de Cauchy polynomial).

Définition 2.55 *Les fonctions générées par* GPAC *sont les fonctions pCp.*

La preuve que les fonctions GPAC-calculables sont différentiellement algébriques est alors facile : cela consiste à « aplatir » le système d'équations différentielles en un seul polynôme.

On a donc

$$f \text{ pCp} \Rightarrow f \text{ différentiellement algébrique} \Rightarrow f \text{ analytique} \Rightarrow f \in \mathscr{C}^\infty.$$

Pour la suite, nous utiliserons cette définition des GPAC, et donc montrer qu'une fonction est GPAC-calculable ne reposera pas sur la réalisation d'un circuit calculant la fonction mais sur l'exhibition d'un système d'équations différentielles polynomial.

Deuxième partie

Comparaison entre fonctions $\mathbb{R}$-récursives et analyse récursive

3 Caractérisation des fonctions récursives par des fonctions $\mathbb{R}$-récursives

> Celui qui veut nager dans l'océan de vérité, doit se réduire à zéro.
>
> *(Gandhi)*

Dans cette partie, nous allons utiliser les résultats de Manuel Campagnolo, Cristopher Moore et José Félix Costa [Cam01, CMC00b, CMC02] et les étendre. À partir d'un résultat caractérisant par des fonctions $\mathbb{R}$-récursives les fonctions discrètes élémentaires, nous allons obtenir une caractérisation des fonctions discrètes récursives, puis étendre ces caractérisations pour lier les fonctions élémentairement et récursivement calculables à des fonctions $\mathbb{R}$-récursives.

L'objet de ce chapitre est de démontrer le méta-théorème suivant :

Méta-théorème 1 *Il existe un opérateur* $!\mu$ *tel que* $DP(\mathscr{L}+!\mu) = \mathcal{R}ec(\mathbb{N})$.

En réalité, le résultat concernera les fonctions récursive préfixe-partielles ($\mathcal{PPR}ec(\mathbb{N})$).

Nous allons pour cela dans un premier temps définir un nouveau schéma μ noté UMU de recherche de zéro inspiré de résultats d'analyse récursive. La section 3.1 présentera la définition de ce nouvel opérateur, ainsi que des modifications des définitions de certains autres opérateurs pour les rendre compatibles avec UMU. Nous montrerons également quelques exemples d'utilisation de cet opérateur. Ensuite viendra la classe $\mathscr{L}+!\mu$ dans la section 3.2 où nous définirons cette classe appelée à capturer les fonctions récursives discrètes. Cette section contiendra également le résultat d'égalité entre les fonctions récursives discrètes et les restrictions aux entiers des fonctions de $\mathscr{L}+!\mu$ ainsi que la preuve de ce résultat. Nous présentons ensuite en section 3.3 des corollaires et applications de ce résultat (et de cette preuve) ainsi qu'une discussion sur d'autres schémas de minimisation réelle envisagés et qui permettraient d'obtenir un résultat similaire.

Les résultats présentés dans ce chapitre sont adaptés d'un travail réalisé en collaboration avec Olivier Bournez et qui a donné lieu aux publications [BH06, BH05b].

3.1 Un opérateur de minimisation réel

Nous avons vu dans le paragraphe 2.2.2 que la classe $\mathscr{L}$ permet de représenter l'ensemble des fonctions élémentaires discrètes. Dans le cas discret, les opérateurs permettant de passer des fonctions élémentaires aux fonctions récursives sont d'une part un opérateur de récursion primitive ou d'itération, qui est assez analogue à l'opérateur $\int$ et d'autre part l'opérateur de minimisation ou de recherche de zéro μ. Des résultats classiques sur

les fonctions récursives discrètes font apparaître que l'opérateur REC peut être simulé par l'opérateur μ. Donc μ suffit à passer des fonctions élémentaires discrètes aux fonctions récursives discrètes. En nous basant sur cette idée, nous allons donc proposer un schéma de recherche de zéro pour passer des fonctions élémentairement calculables aux fonctions récursivement calculables.

3.1.1 Définitions

L'opérateur de minimisation $\mu_{\mathbb{R}}$ permet de capturer des fonctions non récursivement calculables comme nous l'avons vu avec la proposition 2.30, nous allons pour y remédier définir un opérateur de recherche de zéro unique, incités en cela par [Wei00, Corollaire 6.3.5] et le théorème des fonctions implicites constitué ici par la proposition 3.1. L'énoncé de cette proposition est adaptée de [LFA74].

Proposition 3.1 (Théorème des fonctions implicites) *Soit $f : \mathcal{D}\times\mathcal{I} \to \mathbb{R}$ où $\mathcal{D}\times\mathcal{I}$ est un produit d'intervalles fermés de $\mathbb{R}^{k+1}$ une fonction de classe $\mathscr{C}^l$ avec $l \geq 1$.*

Supposons que pour tout $\vec{x} \in \mathcal{D}$,

1. *l'équation $f(\vec{x}, y) = 0$ ait une et une seule solution,*
2. *cette solution y_0 appartienne à $\overset{\circ}{\mathcal{I}}$ (l'intérieur de $\mathcal{I}$)*
3. *$\frac{\partial f}{\partial y}(\vec{x}, y_0) \neq 0$.*

Alors, la fonction $g : \mathbb{R}^k \to \mathbb{R}$ qui à $\vec{x} \in \mathcal{D}$ associe l'unique racine y_0 est définie sur $\mathcal{D}$ et est de classe $\mathscr{C}^l$.

On utilise ce théorème pour définir notre opérateur de recherche de zéro.

Définition 3.2 (Schéma UMU**)** *Soit une fonction différentiable $f : \mathcal{D} \times \mathcal{I} \to \mathbb{R}$ avec $\mathcal{D} \times \mathcal{I} \subset \mathbb{R}^{k+1}$ un produit d'intervalles fermés. Si*

1. *pour tout $\vec{x} \in \mathcal{D}$, la fonction $y \mapsto f(\vec{x}, y)$ est croissante*
2. *cette fonction possède une racine unique $y_0 \in \mathcal{I}$,*
3. *la racine y_0 appartient à $\overset{\circ}{\mathcal{I}}$*
4. *$\frac{\partial f}{\partial y}(\vec{x}, y_0) > 0$.*

Alors UMU(f) *est définie sur $\mathcal{D}$ par*

$$\mathrm{UMU}(f) : \left\{ \begin{array}{ccc} \mathcal{D} & \to & \mathbb{R} \\ \vec{x} & \mapsto & y_0 \text{ tel que } f(\vec{x}, y_0) = 0. \end{array} \right.$$

Si les conditions ne sont pas réunies, on ne définit pas UMU.

Pour prévenir l'apparition de fonctions indésirables qui pourrait survenir en utilisant des intégrations linéaires avec des minimisations, nous allons utiliser un opérateur d'intégration linéaire contrôlée à la place de l'opérateur LI.

Définition 3.3 (Intégration linéaire contrôlée) *Étant données g, h et c telles que h soit différentiable et $\|\frac{\partial h}{\partial y}(\vec{x}, y)\| < c(\vec{x}, y)$. Étant donné un fermé F, on définit la fonction $f = \mathrm{CLI}_F(g, h, c)$ comme la solution de l'équation différentielle*

$$\begin{cases} \frac{\partial f}{\partial y}(\vec{x}, y) = h(\vec{x}, y) f(\vec{x}, y) \\ f(\vec{x}, 0) = g(\vec{x}) \end{cases}$$

définie sur F.

Il est à remarquer que la solution maximale d'une telle équation différentielle est *a priori* définie sur un ouvert. Cependant, les fonctions définies par l'analyse récursive sur des ouverts présentent des difficultés, par exemple, elles ne sont pas stables par composition. Nous ne nous intéressons donc ici qu'aux fonctions définies sur des fermés (y compris des fermés à bornes infinies). Par la suite, nous omettrons de préciser le fermé considéré pour appliquer le schéma CLI pour avoir des notations plus concises.

D'une certaine façon, ce qui apparaît dans cette définition si l'on y impose de préciser le fermé sur lequel devra être définie la fonction obtenue par le schéma CLI est le fait que nous ne travaillons pas uniquement sur des fonctions mais sur des paires fonction, domaine. Par la suite, nous ne préciserons que si c'est nécessaire le domaine étudié, mais il faut garder à l'esprit qu'une fonction est toujours liée au domaine où on la définit.

Il est également bon de remarquer que l'application d'un opérateur peut faire changer le domaine de travail, ou produire une fonction qui ne sera définie que sur un sous-domaine de celui considéré, dans ce cas, sachant que si une fonction appartient à une de nos classes, la restriction de cette fonction à un sous-domaine appartiendra encore à la classe (sous réserve que ce sous-domaine vérifie quelques conditions, par exemple, on ne considère que des fonctions définies sur des produits d'intervalles fermés).

3.1.2 Utilisations de ces opérateurs

Dans cette sous-section, nous présentons quelques exemples de fonctions obtenues à l'aide de l'opérateur UMU. En particulier, nous montrons que les fonctions -1 et π qui étaient des fonctions de base de $\mathscr{L}$ peuvent être obtenue grâce à cet opérateur. Nous justifions également par l'exemple certaines hypothèses de la définition. Enfin, nous montrons un résultat sur les fonctions réciproques.

Il faut tout d'abord remarquer que la borne de contrôle utilisée dans la définition du schéma d'intégration linéaire contrôlée est appliquée à h (plus précisément à sa dérivée) et non à la solution de l'équation différentielle comme on le fait parfois dans le cas discret par exemple pour la définition de l'opérateur de récursion bornée dans [Ros84]. Ainsi, la fonction exp est toujours produite à l'aide de ce schéma : $\exp = \mathrm{CLI}(1, 1, 1)$. Ici, le fermé considéré est $]-\infty, +\infty[$

Concernant l'opérateur UMU, notons tout d'abord qu'il permet de produire les constantes -1, π assez simplement :

Exemple 3.4 *soit $f = x \mapsto 1 + x$, alors,* $\mathrm{UMU}(f) = -1$

Exemple 3.5 *soit*

$$f = (x, y) \mapsto (1 + x^2)y - 1$$

$\mathrm{UMU}(f) = x \mapsto \frac{1}{1+x^2}$ *qui est la dérivée de la fonction* arctan. *Donc*

$$\arctan = \mathrm{CLI}_{]-\infty,+\infty[}(0, \mathrm{UMU}(f), 1)$$

et $\pi = 4 \arctan(1)$.

La définition de UMU est très inspirée du théorème des fonctions implicites (rappelé en proposition 3.1) qui nous assure l'existence et l'appartenance à $\mathscr{C}^2$. De façon plus intuitive, considérons pour un $\vec{x}$ donné la fonction $y \mapsto f(\vec{x}, y)$. Nous requérons que la racine y_0 soit unique et que la fonction soit croissante, ainsi la recherche de cette racine sera « facile ». La dérivée en cette racine doit être strictement positive car sa nullité pourrait créer une discontinuité sur les dérivées de la fonction $\mathrm{UMU}(f)$.

Exemple 3.6 $f(x, y) = x - y^3$. *Pour tout* x, *il existe exactement un* y_0 *tel que* $f(x, y) = 0$ *(on peut noter* $y_0 = x^{1/3}$*), la fonction* $y \mapsto x - y^3$ *est bien croissante, mais sa dérivée en la racine est nulle et la fonction que donnerait* $\mathrm{UMU}(f)$ *n'est pas dérivable en* 0.

Montrons maintenant que si une fonction f est strictement croissante, sa réciproque peut être définie à l'aide de l'opérateur UMU.

Proposition 3.7 *Soit* f *une fonction différentiable strictement croissante de* $[a, b]$ *vers* $[c, d] = f([a, b])$. *La fonction* f^{-1} *définie sur* $[c, d]$ *par* $f^{-1} \circ f = Id$ *peut être définie en n'utilisant que* f, COMP *et* UMU.

Démonstration : Soit $g : x, y \mapsto f(y) - x$ définie sur $[c, d] \times [a, b]$. Pour $x \in [c, d]$ fixé, la fonction $y \mapsto g(x, y)$ est strictement croissante et possède une racine unique $y_0 \in [a, b]$. La dérivée de $y \mapsto g(x, y)$ en y_0 est strictement positive. On peut donc appliquer UMU à g. $\mathrm{UMU}(g)$ est définie sur $[c, d]$ et vérifie

$$\forall x \in [a, b], \mathrm{UMU}(g)(f(x)) = x$$

□

3.2 Classe capturant les fonctions $\mathbb{R}$-récursives

L'objectif de cette section est de présenter une classe de fonctions que nous nommons $\mathscr{L}+!\mu$ qui va caractériser les fonctions récursives préfixe-partielles discrètes à l'aide de l'opérateur UMU. Nous commençons en 3.2.1 par définir cette classe de fonctions et montrer que toutes les fonctions de $\mathscr{L}+!\mu$ sont de classe $\mathscr{C}^2$ et définies sur des produits d'intervalles fermés, puis dans la sous-section 3.2.2, nous énonçons et prouvons le résultat principal de ce chapitre, à savoir l'égalité entre l'ensemble des fonctions récursives discrètes et les restrictions aux entiers des fonctions de $\mathscr{L}+!\mu$, formalisant ainsi le méta-théorème 1.

3.2.1 Définitions et propriétés basiques

Nous pouvons maintenant définir une classe de fonctions contenant la classe $\mathscr{L}$ et capturant les fonctions $\mathbb{R}$-récursives.

Définition 3.8 ($\mathscr{L}+!\mu$) *On définit la classe $\mathscr{L}+!\mu$ de la façon suivante :*

$$\mathscr{L}+!\mu = [0, 1, U, \theta_3; \mathrm{COMP}, \mathrm{CLI}, \mathrm{UMU}].$$

Proposition 3.9 *La classe $\mathscr{L}$ est contenue dans $\mathscr{L}+!\mu$.*

Démonstration : Ce résultat découle de 2 choses : tout d'abord le fait que -1 et π appartiennent à $\mathscr{L}+!\mu$ (voir paragraphe 3.1.2) ; et ensuite le fait que l'on peut remplacer LI par CLI dans la définition de la classe $\mathscr{L}$. On a en effet,

$$\mathscr{L} = [0, 1, -1, \pi, U, \theta_3; \mathrm{COMP}, \mathrm{CLI}].$$

Pour ce second fait, on établit que toute fonction de $\mathscr{L}$ est bornée par une itérée de l'exponentielle : $\exp^{[d]}$ (où $\exp^{[0]} = x \mapsto x$ et $\exp^{[n+1]} = \exp \circ \exp^{[n]}$).

$\forall f \in \mathscr{L}$, $\exists d, A, B$ tels que $\forall x, \|f(x)\| < A \exp^{[d]}(B\|x\|)$ et de même pour la différentielle de f.

0, 1, -1 et U sont inférieurs à $2\exp(0)$.

$|\theta_3(x)| < \exp(|x|)$.

Si $\|f(x)\| < A_f \exp^{[d_f]}(B_f\|x\|)$ et $\|g(x)\| < A_g \exp^{[d_g]}(B_g\|x\|)$, alors

$$\mathrm{COMP}(f, g)(x) < A_f \exp^{[d_f+d_g]}(B_f A_g B_g \|x\|).$$

Et $\|\mathrm{LI}(f, g)(x)\| < A_g \exp^{[d_g+1+d_f]}(B_g A_f B_g \|x\|)$.

Maintenant, il suffit de remarquer que exp peut être obtenue avec CLI (voir paragraphe 3.1.2), que donc avec COMP, on peut produire les $\exp^{[d]}$ et donc que chaque utilisation de LI dans la classe $\mathscr{L}$ peut être remplacée par une utilisation de CLI. □

Lemme 3.10 *Les fonctions appartenant à $\mathscr{L}+!\mu$ sont définies sur des produits d'intervalles fermés.*

Démonstration :

- Les fonctions de base sont définies sur $\mathbb{R}$ ou $\mathbb{R}^n$;
- COMP : étant données f et g définies sur des produits d'intervalles fermés avec $g(\mathcal{D}om(g)) \subseteq \mathcal{D}om(f)$. Si $\mathrm{COMP}(f, g)$ existe, elle est définie sur $\mathcal{D}om(g)$;
- CLI : par définition, $\mathrm{CLI}(g, h, c)$ est définie sur un intervalle fermé ou produit d'intervalles fermés ;
- UMU : la définition de UMU fait que si f est définie sur $\mathcal{D}\times\mathcal{I}$ et vérifie les hypothèses pour que $\mathrm{UMU}(f)$ soit définie, alors $\mathrm{UMU}(f)$ est définie sur $\mathcal{D}$.

Les fonctions de $\mathscr{L}+!\mu$ sont donc définies sur des produits d'intervalles fermés. □

Lemme 3.11 *Les fonctions appartenant à $\mathscr{L}+!\mu$ sont de classe $\mathscr{C}^2$.*

Démonstration : Montrons ce résultat par récurrence structurelle :
- 0, 1 et U sont de classe $\mathscr{C}^\infty$ donc en particulier $\mathscr{C}^2$;
- θ_3 est de classe $\mathscr{C}^2$.
- si f et g sont de classe $\mathscr{C}^2$, alors la composition de ces 2 fonctions est également de classe $\mathscr{C}^2$.
- CLI(f, g, c) préserve $\mathscr{C}^2$ tout comme LI le fait (voir proposition 2.41)
- UMU préserve $\mathscr{C}^2$: en effet, on peut appliquer le théorème d'inversion local en tout point $\vec{x} \in \mathcal{D}$.

□

3.2.2 La classe $\mathscr{L}+!\mu$ capture les fonctions récursives

Dans cette partie, nous présentons le résultat principal du chapitre, à savoir que la classe $\mathscr{L}+!\mu$ que nous venons de définir représente d'une certaine façon une caractérisation continue des fonctions récursives discrètes. Ce résultat est l'objet du théorème 3.12 qui incarne le méta-théorème 1. Il montre que de même que la classe $\mathscr{L}$ caractérise les fonctions élémentaires :

$$DP(\mathscr{L}) = \mathscr{E}(\mathbb{N}),$$

la classe $\mathscr{L}+!\mu$ caractérise les fonctions récursives discrètes :

$$DP(\mathscr{L}+!\mu) = \mathcal{PPR}ec(\mathbb{N}).$$

Théorème 3.12 *Pour les fonctions définies sur un produit d'intervalles fermés à bornes rationnelles ou infinies,*

$$DP(\mathscr{L}+!\mu) = \mathcal{PPR}ec(\mathbb{N}).$$

3.2.2.a Sens direct de la preuve

Pour montrer le sens direct de ce théorème, à savoir l'inclusion de $DP(\mathscr{L}+!\mu)$ dans $\mathcal{PPR}ec(\mathbb{N})$, on montre en réalité que $\mathscr{L}+!\mu \subseteq \mathcal{R}ec(\mathbb{R})$.

Proposition 3.13 *Pour f définie sur un produit d'intervalles fermés à bornes infinies ou rationnelles,*

$$f \in \mathscr{L}+!\mu \Rightarrow f \in \mathcal{R}ec(\mathbb{R}).$$

Démonstration : On sait que $\mathscr{L} \subseteq \mathscr{E}(\mathbb{R})$. On sait de plus que les opérateurs COMP, LI préservent $\mathscr{E}(\mathbb{R})$.

Prouvons par récurrence structurelle sur $\mathscr{L}+!\mu$ que $\mathscr{L}+!\mu \subseteq \mathcal{R}ec(\mathbb{R})$

Les fonctions de base 0, 1, U et θ_3 sont élémentairement calculables donc appartiennent à $\mathcal{R}ec(\mathbb{R})$.

Considérons deux fonctions f et g appartenant à $\mathcal{R}ec(\mathbb{R})$ et telles que $\mathcal{I}m(g) \subseteq \mathcal{D}om(f)$, c'est-à-dire telles que $f \circ g$ ait un sens. Elles sont respectivement calculées par les fonctions ϕ_f et ϕ_g. Pour calculer COMP(f, g), il suffit d'utiliser $\phi_f \circ \phi_g$. En effet, pour $x \in \mathcal{D}om(g)$, $\forall X \rightsquigarrow x, \phi_g(X) \rightsquigarrow g(x)$ et donc $\phi_f(\phi_g(X)) \rightsquigarrow f \circ g(x)$.

Les cas de CLI et de UMU sont l'objet des lemmes 3.14 et 3.15. □

Pour la calculabilité du résultat d'une intégration linéaire contrôlée, nous utilisons la preuve de [Cam01, Proposition 4.3.5] qui montre dans le cas élémentairement calculable que le résultat d'une intégration linéaire est encore élémentairement calculable. Pour une étude plus générale de la calculabilité d'une fonction définie comme solution d'un problème de Cauchy linéaire, on pourra se reporter à [WZ06].

Lemme 3.14 *Étant données g, h et c appartenant à $\mathcal{R}ec(\mathbb{R})$, si* CLI$(g, h, c)$ *est définie, alors* CLI$(g, h, c) \in \mathcal{R}ec(\mathbb{R})$.

Démonstration : La preuve de ce lemme s'inspire de la preuve du fait que LI préserve $\mathscr{E}(\mathbb{R})$ que l'on peut trouver dans [Cam01]. Elle consiste à vérifier qu'en utilisant la méthode d'Euler de calcul numérique d'une intégrale, on exhibe une façon de calculer cette intégrale.

On se donne g, h et c calculées respectivement par φ_g, φ_h et φ_c. Et on veut une fonction récursive ψ telle que pour tout x, y et toute suite (x_i, y_i),

$$\psi((x_i, y_i)) \rightsquigarrow \text{CLI}(g, h, c)(x, y).$$

On considère donnés x, y et (x_i, y_i). Pour approcher $f(x, y)$, nous allons approcher $g(x)$ puis l'intégrale elle-même en coupant des tranches de plus en plus petites pour minimiser l'erreur. Dans le cas élémentaire, on s'aperçoit que le nombre de tranches et le calcul sur chaque tranche sont élémentaires (voir la preuve de [Cam01, Proposition 4.3.5]). Donc la fonction CLI(g, h, c) est élémentairement calculable. Dans le cas récursivement calculable, CLI(g, h, c) est récursivement calculable. □

Lemme 3.15 *Étant donnée $f \in \mathcal{R}ec(\mathbb{R})$, définie sur un fermé dont les extrémités sont rationnelles ou infinies, telle que* UMU(f) *soit définie, alors* UMU$(f) \in \mathcal{R}ec(\mathbb{R})$.

Démonstration : L'idée de base de cette preuve est d'utiliser [Wei00, Corollaire 6.3.9] qui montre qu'il est possible de calculer la racine unique d'une fonction sur un compact.

Dans un premier temps, nous allons donc exhiber un algorithme permettant restreindre l'intervalle de recherche du zéro à un compact, dans un deuxième temps, nous allons montrer un algorithme qui renverra des approximations à n'importe quelle précision de la racine.

1. Considérons $f \in \mathcal{R}ec(\mathbb{R})$ définie sur $\mathcal{D} \times \mathcal{I}$ avec $\mathcal{I} = [a, b]$ où a et b peuvent éventuellement être infinis. On suppose que 0 appartient à cet intervalle. On pourrait cependant remplacer 0 par n'importe quel rationnel appartenant à $[a, b]$.
   Soit $\vec{x} \in \mathcal{D}$. Soit y_0 l'unique racine de $y \mapsto f(\vec{x}, y)$. Comme nous avons imposé que cette fonction soit croissante, pour tout $y < y_0$, $f(x, y) < 0$ et pour tout $y > y_0$, $f(x, y) > 0$. Donc trouver un compact dans lequel se trouve y_0 consiste uniquement à prendre un compact de base et l'agrandir vers la gauche si les valeurs y sont positives, vers la droite si les valeurs y sont négatives.
   L'algorithme suivant permet de trouver un compact contenant y_0.

   $m = 1$
   Répéter
   Calculer $f_1 = f(\vec{x}, \min(b, m))$ à $\pm 2^{-m}$ près
   Calculer $f_2 = f(\vec{x}, \max(a, -m))$ à $\pm 2^{-m}$ près
   $m = m + 1$
   Jusqu'à ($f_1 > 2^{-m}$ et $f_2 < -2^{-m}$)
   Renvoyer m

   On a alors un intervalle compact $[\alpha, \beta] = [\max(a, -m), \min(b, m)]$ qui contient y_0.
2. Il reste à chercher la racine y_0 sur ce compact.
   Soit $n \in \mathbb{N}$. Nous cherchons un rationnel approchant y_0 à 2^{-n} près. Découpons $[\alpha, \beta]$ en 2^i segments $[y_j, y_{j+1}]$ pour $0 \leq j < 2^i$ avec $y_j = \alpha + \frac{\beta - \alpha}{2^i} j$. Soit pour chaque j, z_j une approximation de $f(\vec{x}, y_j)$ à 2^{-i} près.
   Pour que la racine y_0 soit dans l'intervalle $[z_j, z_{j+1}]$, il faut soit que $|z_j| < 2^{-i}$, soit que $|z_{j+1}| < 2^{-i}$ soit que $z_j z_{j+1} < 0$. Définissons j_- et j_+ comme suit :

$$j_- = \min \left\{ j; j \in \{0, .., 2^i - 1\} \text{ et } \left(|z_j| < 2^{-i} \text{ ou } |z_{j+1}| < 2^{-i} \text{ ou } z_j z_{j+1} < 0 \right) \right\}$$
$$j_+ = \max \left\{ j + 1; j \in \{0, .., 2^i - 1\} \text{ et } \left(|z_j| < 2^{-i} \text{ ou } |z_{j+1}| < 2^{-i} \text{ ou } z_j z_{j+1} < 0 \right) \right\}$$

   On est alors certain que $y_0 \in [z_{j_-}, z_{j_+}]$. Posons $m_i = z_{j_-}$ et $M_i = z_{j_+}$. On a des suites (m_i) et (M_i) qui encadrent la valeur recherchée. De plus, pour tout i, $m_i < M_i$.
   Les suites (m_i) et (M_i) étant à valeurs dans un compact ($[\alpha, \beta]$), on peut d'après le théorème de Bolzano-Weierstrass en extraire des suites convergentes $(m_{\phi(i)})$ et $(M_{\psi(i)})$. Appelons m^* et M^* les limites respectives de ces deux suites.
   Pour tout i, on a au moins l'une des trois inégalités suivantes :

$$|f(\vec{x}, m_i)| < |f(\vec{x}, m_i) - z_j| + |z_j| < 2^{-i} + 2^{-i}$$
$$|f(\vec{x}, m_i + 2^{-i})| < |f(\vec{x}, m_i + 2^{-i}) - z_{j+1}| + |z_{j+1}| < 2^{-i} + 2^{-i}$$
$$f(\vec{x}, m_i) f(\vec{x}, m_i + 2^{-i}) < 0$$

   On en déduit qu'il existe n_i tel que $n_i - m_i \leq 2^{-i}$ et $|f(\vec{x}, n_i)| < 2 \times 2^{-i}$. Comme $f(\vec{x}, .)$ est uniformément continue sur $[\alpha, \beta]$, il existe d tel que $\forall x, y \in [\alpha, \beta]$, $|f(x) -$

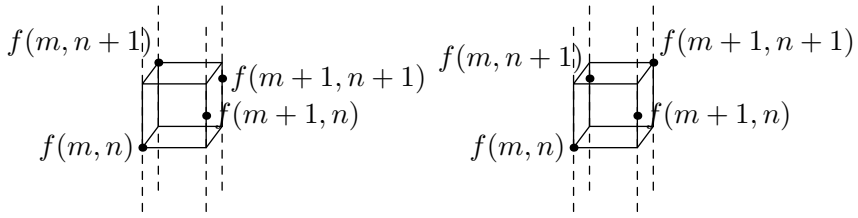

FIG. 3.1: Les parallélépipèdes canoniques pour différentes dispositions des points de base.

$f(y)| < d|x-y|$. Et donc $f(\vec{x}, m_i) < (2+d)2^{-i}$. On en déduit que $f(\vec{x}, m^\star) = 0$. De même $f(\vec{x}, M^\star) = 0$. Donc, $m^\star = y_0 = M^\star$, et il existe i tel que $M_i - m_i < 2^{-n}$. Donc l'algorithme suivant termine et renvoie une approximation de y_0 à 2^{-n} près.

$i = 0$
Répéter
    Calculer m_i et M_i
    $i = i + 1$
Jusqu'à $(M_i - m_i < 2^{-n})$
Renvoyer m_i

Ce qui termine la preuve. □

3.2.2.b Sens indirect de la preuve

La proposition 3.13 implique de façon immédiate le sens direct du théorème 3.12. Le sens indirect est l'objet de la proposition 3.17. Pour la prouver, nous utiliserons le lemme suivant :

Lemme 3.16 *Étant donnée $f : \mathbb{R}^2 \to \mathbb{R}^2 \in \mathscr{L}$, il existe une fonction $\tilde{f} : \mathbb{R}^2 \to \mathbb{R}^2$ appartenant également à $\mathscr{L}$ telle que*

$$\begin{array}{l} \forall m,n \in \mathbb{N} \\ \forall x \in [m, m+1[\\ \forall y \in [n, n+1[\end{array} \left\{ \begin{array}{l} \tilde{f}(m,n) = f(m,n) \\ \tilde{f}(x,y) \in [\min\{f(m,n), f(m+1,n), \\ \qquad\qquad f(m,n+1), f(m+1,n+1)\}, \\ \qquad\quad \max\{f(m,n), f(m+1,n), \\ \qquad\qquad f(m,n+1), f(m+1,n+1)\}] \end{array} \right.$$

On dit que la fonction $\tilde{f}$ est alors un représentant canonique de f. $\tilde{f}$ est égal à f sur les points de coordonnées entières et est incluse dans le plus petit parallélépipède à faces horizontales contenant les quatre points $(i, j, f(i,j))$ pour $i \in \{m, m+1\}$ et $j \in \{n, n+1\}$. La figure 3.1 représente des exemples de tels parallélépipèdes.

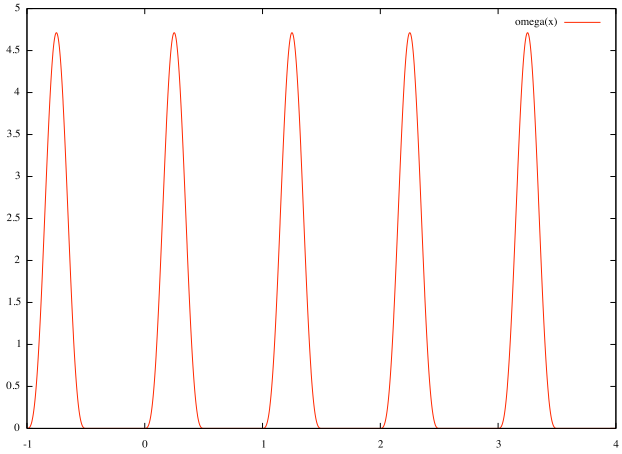

FIG. 3.2: Représentation graphique de la fonction ω.

Démonstration : Nous allons prouver ce lemme en construisant une fonction $\tilde{f}$ remplissant les conditions voulues.

Nous avons donc $f \in \mathscr{L}$. Posons $\zeta = \frac{3\pi}{2}$,

$$\omega : x \mapsto \zeta\theta_3(\sin(2\pi x))$$
$$\Omega : x \mapsto \int_0^x \omega(t)\,\mathrm{d}t$$
$$int : x \mapsto \Omega(x - \frac{1}{2})$$

La fonction *int* (représentée en figure 3.3) est une simulation de la fonction partie entière : étant donné $i \in \mathbb{N}$, pour tout $x \in [i, i + \frac{1}{2}]$, on a $int(x) = i = \lfloor x \rfloor$.

Posons $\Delta(i, y) = f(i, y + 1) - f(i, y)$. Pour tout $i \in \mathbb{N}$, $y \in \mathbb{R}$, nous avons

$$\omega(y)\Delta(i, int(y)) = \begin{cases} 0 & \text{si } y - \lfloor y \rfloor \geq 1/2 \\ \omega(y)\Delta(i, \lfloor y \rfloor) & \text{sinon} \end{cases}$$

Appelons G la solution du problème de Cauchy

$$\begin{cases} G(x, 0) = f(x, 0) \\ \frac{\partial G}{\partial y}(x, y) = \omega(y)\Delta(x, int(y)). \end{cases}$$

On montre facilement que pour tout $(i, j) \in \mathbb{N}^2$, on a $G(i, j) = f(i, j)$. De plus, par construction de la fonction G, pour tout i, $G(i, y)$ appartient à l'intervalle délimité par $G(i, \lfloor y \rfloor) = f(i, \lfloor y \rfloor)$ et $G(i, \lfloor y + 1 \rfloor) = f(i, \lfloor y + 1 \rfloor)$.

Soit $\nabla(x, y) = G(x+1, y) - G(x, y)$. Définissons maintenant $\tilde{f}$ comme étant la solution du problème de Cauchy suivant :

$$\begin{cases} \tilde{f}(0, y) = G(0, y) \\ \frac{\partial \tilde{f}}{\partial x}(x, y) = \omega(x)\nabla(int(x), y). \end{cases}$$

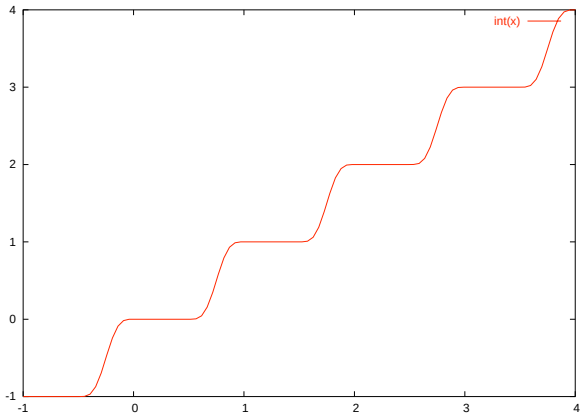

FIG. 3.3: Représentation graphique de la fonction *int*.

On montre simplement que pour tout $(i,j) \in \mathbb{N}^2$, on a $\tilde{f}(i,j) = G(i,j) = f(i,j)$. Et tout comme G, $\tilde{f}$ vérifie $\forall x, y$, $\tilde{f}(x,y)$ appartient à l'intervalle délimité par $\tilde{f}(x, \lfloor y \rfloor)$ et $\tilde{f}(x, \lfloor y+1 \rfloor)$. Mais de plus, pour tout x, y, $\tilde{f}(x,y)$ appartient à l'intervalle délimité par $\tilde{f}(\lfloor x \rfloor, y)$ et $\tilde{f}(\lfloor x+1 \rfloor, y)$. Ces 2 dernières propriétés suffisent pour établir que $\tilde{f}$ est un représentant canonique de f. □

Nous pouvons maintenant montrer le sens indirect du théorème 3.12. Il s'agit de la proposition suivante :

Proposition 3.17

$$\mathcal{PPR}ec(\mathbb{N}) \subseteq DP(\mathscr{L}+!\mu)$$

Démonstration : Soit ϕ une fonction récursive préfixe-partielle discrète. Nous allons construire une extension aux réels de ϕ appartenant à $\mathscr{L}+!\mu$.

Cette fonction ϕ peut être décomposée en $\phi = \chi \circ \mu(\psi)$ avec χ et ψ élémentaires. Cette propriété provient de [Odi92, Théorème I.7.3] en remarquant que les fonctions primitives récursives que produit la preuve sont en réalité élémentaires.

ϕ est définie sur un intervalle $\{0, n\}$, ce qui signifie que $z \mapsto \psi(m,z)$ possède une racine pour tout $m \in \{0, ..., n\}$. Nous allons donc construire une fonction $f \in \mathscr{L}+!\mu$ définie sur $[0, n]$.

Il suffirait donc intuitivement de prendre des extensions dans $\mathscr{L} \subset \mathscr{L}+!\mu$ de χ et ψ, d'appliquer UMU et de prendre la composée. Cependant, du fait de la définition de UMU, il faut adapter ψ et son extension.

Posons

$$\sigma(m,n) = \prod_{z<n} \psi(m,z).$$

On a bien σ élémentaire : elle est construite à l'aide d'une multiplication bornée. Appelons n_0 le plus petit n tel que $\psi(m,n) = 0$. $\sigma(m,.)$ est non nulle pour tout $n \leq n_0$ et nulle pour tout $n \geq n_0 + 1$.

Considérons

$$\kappa(m,n) = 1 \ominus (1 \ominus (1 \ominus \sigma(m,n) + \sigma(m,n+1))).$$

κ est elle aussi élémentaire, et vérifie :

$$\begin{array}{ll} \text{Pour } n \leq n_0 - 1, & \kappa(m,n) = 1 \\ \text{Pour } n = n_0, & \kappa(m,n) = 0 \\ \text{Pour } n \geq n_0 + 1, & \kappa(m,n) = 1. \end{array}$$

Construisons enfin une fonction ι qui sépare les n précédant la racine des n suivant la racine :

$$\iota(m,n) = 1 \ominus \kappa + 2 \times (1 \ominus \sigma(m,n)).$$

En résumé,

	$0 \ldots n_0 - 1$	n_0	$n_0 + 1 \ldots \infty$
ψ	> 0	0	?
σ	> 0	> 0	0
κ	1	0	1
ι	0	1	2

ι est élémentaire et pour tout m, il existe un unique n_0 tel que $\iota(m,n_0) = 1$. De plus, ce n_0 est aussi le plus petit n tel que $\psi(m,n) = 0$.

Soit $i : \mathbb{R}^2 \to \mathbb{R}$ une extension appartenant à $\mathscr{L}$ de ι. Soit $\tilde{i}$ un représentant canonique (construit à l'aide du lemme 3.16) de i. L'opérateur UMU ne peut pas encore être directement appliqué à la fonction $\tilde{i} - 1$: en effet, même si $\forall m \in \mathbb{N}$, il existe un unique $n_0 \in \mathbb{N}$ tel que $(\tilde{i} - 1)(m,n_0) = 0$, il peut exister plusieurs réels annulant cette fonction. De plus, la dérivée en la (les) racine(s) peut être nulle.

Nous savons que pour $m \in \mathbb{N}$, pour tout $y \leq n_0 - 1$, $(\tilde{i} - 1)(m,y) = -1$, pour $y \geq n_0 + 1$, $(\tilde{i} - 1)(m,y) = 1$, et pour $y \in [n_0 - 1, n_0]$, $(\tilde{i} - 1)(m,y) = \Omega(y - n_0)$. Nous allons maintenant appliquer un filtre « passe-haut » à cette fonction pour imposer l'unicité de la racine pour tout $m \in \mathbb{R}$ et assurer que la dérivée en cette racine est non nulle.

Soit $\mathcal{M}(x) = \theta_3(x+1)$. $\mathcal{M}$ filtre toute valeur inférieure à -1 : $\forall x \leq -1, \mathcal{M}(x) = 0$, et amplifie toute valeur supérieure à 0 : $\forall x \geq 0, \mathcal{M}(x) \geq 1$. Définissons $\tilde{g}$ comme la solution du problème de Cauchy suivant :

$$\begin{cases} \tilde{g}(x,0) = -1 \\ \frac{\partial \tilde{g}}{\partial y}(x,y) = \alpha \mathcal{M}(\tilde{i}(x,y) - 1) \end{cases}$$

avec $\alpha = \frac{1024}{2609}$. On peut vérifier que pour cette valeur de α, on a $\alpha \int_{-1}^{0} \mathcal{M}(\Omega(t))\,\mathrm{d}t = 1$. On peut en déduire que pour $m \in \mathbb{N}$, $\tilde{g}(m,y) = 0$ si et seulement si y est la plus petite racine y_0 de $n \mapsto \psi(m,n)$. De plus, la dérivée de $\tilde{g}$ en le point (m,y_0) est strictement

positive (elle vaut 1). Pour les m non entiers, la fonction $y \mapsto \tilde{g}(m,y)$ est croissante (car $\mathcal{M}$ est partout positive), elle vaut -1 en 0 et a une pente que l'on peut minorer par $\alpha > 0$ à partir d'un certain point (en effet, $\tilde{i} - 1$ vaut 1 à partir du moment où elle atteint 1 pour les deux entiers encadrant m), elle possède donc une racine, cette racine peut cependant ne pas être unique.

Définissons la fonction g comme la solution du problème de Cauchy suivant :

$$\begin{cases} g(x,0) = -1 \\ \frac{\partial g}{\partial y}(x,y) = \beta \mathcal{M}(\tilde{g}(x,y)) \end{cases}$$

avec β habilement choisi pour que $\beta\alpha \int_{-1}^{0} \mathcal{M}(\mathcal{M}(\Omega(t)))\,\mathrm{d}t = 1$ (il est possible d'écrire ce β sous la forme $\frac{a\pi^4}{b\pi^4+c\pi^2+d}$ avec $a,b,c,d \in \mathbb{Z}$). On a donc encore pour tout $m \in \mathbb{N}$, $g(m,y) = 0$ si et seulement si y est la plus petite racine de $n \mapsto \psi(m,n)$, et la dérivée en cette racine est non nulle.

Pour le cas $m \in \mathbb{R}-\mathbb{N}$, on a vu que $y \mapsto \tilde{g}(m,y)$ est croissante et $\mathcal{M}(x) > 0$ pour $x > -1$ et est nul pour $x \leq 1$, donc, tant que $\tilde{g}(m,y) \leq -1$, $g(m,y) = -1$ puis $y \mapsto g(m,y)$ est strictement croissante et on peut minorer sa pente par $\beta\alpha$ à partir d'un certain point.

La fonction g ainsi construite appartient à $\mathscr{L}$ et vérifie pour tout $x \in \mathbb{R}$, il existe un unique y_0 tel que $g(x,y) = 0$. De plus, $\frac{\partial g}{\partial y}(x,y_0) > 0$ et $y \mapsto g(x,y)$ est croissante. On peut donc appliquer UMU à cette fonction g. Comme on le désirait, $\forall x \in \mathbb{N}, \mathrm{UMU}(g)(x) = \mu(\psi)(x)$.

Soit $h \in \mathscr{L}$ une extension aux réels de χ. La fonction $h \circ \mathrm{UMU}(g)$ est une extension aux réels de $\phi = \chi \circ \mu(\psi)$ et appartient à $\mathscr{L}+!\mu$. □

3.3 Résultats concomitants

La classe $\mathscr{L}+!\mu$ fournit donc une caractérisation des extensions réelles des fonctions discrètes calculables. On a également obtenu une inclusion de $\mathscr{L}+!\mu$ dans l'ensemble des fonctions récursivement calculable. D'autres résultats découlent du théorème 3.12 ou de sa preuve. Dans cette section, nous allons tout d'abord montrer que l'inclusion de la classe $\mathscr{L}$ dans $\mathscr{L}+!\mu$ qui constituait la proposition 3.9 est en fait stricte. Nous allons également montrer un théorème de forme normale majorant le nombre de UMU imbriqués qui sont nécessaires pour définir une fonction de $\mathscr{L}+!\mu$. Nous allons ensuite montrer d'autres schémas envisageables pour remplacer UMU.

3.3.1 Inclusion stricte et forme normale

Proposition 3.18

$$\mathscr{L} \subsetneq \mathscr{L}+!\mu$$

Démonstration : $\mathscr{L}+!\mu$ contient par exemple une extension de la fonction d'Ackermann qui n'étant pas élémentaire n'a pas d'extension dans $\mathscr{L}$. □

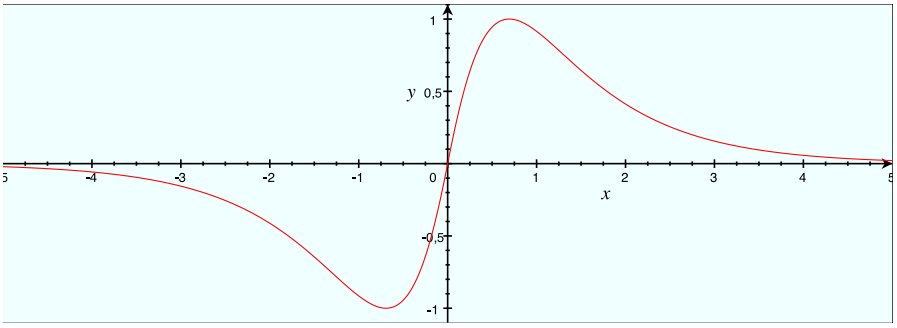

FIG. 3.4: Représentation graphique de la fonction η

Proposition 3.19 *Toute fonction appartenant à $\mathscr{L}+!\mu$ peut s'écrire avec au plus 3* UMU *imbriqués.*

Démonstration : La preuve de la proposition 3.17 montre que si on se donne -1 et π, il suffit d'un UMU pour définir toutes les fonctions de $\mathscr{L}+!\mu$ extensions de fonctions de $\mathcal{R}ec(\mathbb{N})$. Pour obtenir -1 et π, il suffit d'un autre UMU. Donc au final, 3 UMU imbriqués suffisent. □

3.3.2 Pourquoi la définition de UMU impose la croissance de la fonction

Pour définir UMU(f), nous imposons les conditions suivantes à f :

1. pour tout $\vec{x} \in \mathcal{D}$, la fonction $y \mapsto f(\vec{x}, y)$ est croissante
2. cette fonction possède une racine unique $y_0 \in \mathcal{I}$,
3. la racine y_0 appartient à $\overset{\circ}{\mathcal{I}}$
4. $\frac{\partial f}{\partial y}(\vec{x}, y_0) > 0$.

Nous avons déjà expliqué pourquoi nous requérons l'unicité de la racine et la dérivée strictement positive en cette racine. Mais la condition de croissance de $y \mapsto f(\vec{x}, y)$ ne semble à première vue pas indispensable. En effet, la recherche d'une racine unique sur un compact est calculable que la fonction soit ou non croissante. Le fait qui pose problème si l'on accepte des fonctions qui ne sont pas croissantes est que trouver un compact où se trouve la racine peut être difficile.

Considérons la fonction

$$\eta : x \mapsto \begin{cases} -\sin(\exp(x + \ln(\pi))) & \text{si } x \leq 0 \\ \sin(\exp(-x + \ln(\pi))) & \text{si } x \geq 0. \end{cases}$$

La fonction η est représentée en figure 3.4. Une fonction de la forme

$$x, y \mapsto \exp(-x)\eta(y)$$

présente bien des racines uniques pour tout x, mais l'algorithme de recherche d'un compact cherche un point $y > 0$ pour lequel $\exp(-x)\eta(y) > 2^{-y}$, or pour x grand (par exemple 1), il n'existe pas de tel y. Cet algorithme ne s'arrête donc pas si l'on omet l'hypothèse de croissance de $y \mapsto f(\vec{x}, y)$. En fait, intuitivement, pour tout algorithme permettant la recherche d'un compact sur lequel il y a nécessairement une racine, il y aura un x à partir duquel cet algorithme ne saura pas traiter soit le cas ci-dessus soit le cas où la racine tend vers $+\infty$ quand x tend vers $+\infty$.

Et donc, garantir la croissance est une hypothèse nécessaire pour obtenir le résultat escompté. Cependant, il serait possible de remplacer cette hypothèse par une autre comme le montre le paragraphe suivant.

3.3.3 Opérateur de minimum de fonction convexe

Le schéma de minimisation que nous avons choisi n'est pas le seul possible. On peut envisager d'utiliser un opérateur qui au lieu de calculer l'unique racine d'une fonction découvre le minimum unique d'une fonction convexe.

Définition 3.20 (μ_{min_convex}) *On définit le schéma μ_{min_convex} de recherche de minimum d'une fonction convexe de la façon suivante.*

Soit $f : \mathcal{D} \times \mathcal{I} \to \mathbb{R}$ une fonction telle que pour tout $\vec{x} \in \mathcal{D}$, la fonction $y \mapsto f(\vec{x}, y)$ est convexe et atteint un minimum. On note alors

$$\mu_{min\_convex}(f) : \left\{ \begin{array}{lcl} \mathcal{D} & \to & \mathbb{R} \\ \vec{x} & \mapsto & y_0 \in \mathcal{I} \text{ tel que } \forall y \in \mathcal{I} - \{y_0\}, f(\vec{x}, y) > f(\vec{x}, y_0) \end{array} \right.$$

Ce schéma est équivalent au schéma UMU appliqué à la dérivée de f.

Il est à remarquer que le fait qu'une fonction convexe soit unimodale est un des points importants de cette définition. Rappelons qu'une fonction unimodale est strictement décroissante, puis strictement croissante. Elle possède donc un unique minimum, et celui-ci est calculable. Cependant, la recherche de ce minimum est comparable à la recherche du zéro unique d'une fonction dont on n'impose pas la monotonie. La convexité, c'est-à-dire la monotonie de la dérivée est donc également utile.

On a les mêmes résultats avec ce schéma qu'avec l'opérateur UMU choisi :

Proposition 3.21 $\mathcal{R}ec(\mathbb{N}) \subseteq DP(\mathscr{L} + \mu_{min\_convex})$

Démonstration : On a vu dans la preuve de la proposition 3.17 comment montrer qu'il était possible de créer une fonction g à laquelle on peut appliquer UMU pour obtenir une extension d'une fonction donnée de la forme $\mu(\psi)$. En prenant une primitive de cette fonction g, on obtient une fonction convexe sur laquelle on peut appliquer μ_{min_convex} et obtenir le même résultat. □

Proposition 3.22 $\mathscr{L} + \mu_{min\_convex} \subseteq \mathcal{R}ec(\mathbb{R})$

Démonstration : Soit f une fonction de $\mathscr{L}$. Supposons que $\mu_{min\_convex}(f)$ soit définie. Alors, $\mu_{min\_convex}(f)$ est récursivement calculable : de la même façon que pour prouver la proposition 3.13, on peut exhiber un compact dont la pente soit décroissante à l'extrémité gauche et croissante à l'extrémité droite. Le minimum est donc dans ce compact. Et on peut réduire la largeur de ce compact jusqu'à la précision voulue. □

3.3.4 Opérateur de recherche de zéro sur un compact

On aurait pu envisager de ne pas imposer la croissance de la fonction dont on cherche la racine mais de donner à la place des bornes finies encadrant la valeur de la racine :

Définition 3.23 (μ_c) *On définit le schéma μ_c de minimisation sur un compact de la façon suivante :*

soient $f : \mathcal{D} \times \mathcal{I} \to \mathbb{R}$ et $m, M \in \overset{\circ}{\mathcal{I}}$ tels que pour tout $\vec{x} \in \mathcal{D}$, il existe un unique y_0 tel que $f(\vec{x}, y_0) = 0$ et ce y_0 appartient à $[m, M]$.

On note alors

$$\mu_c^{[m,M]}(f) : \left\{ \begin{array}{ccc} \mathcal{D} & \to & \mathbb{R} \\ \vec{x} & \mapsto & y_0 \text{ tel que } f(\vec{x}, y_0) = 0. \end{array} \right.$$

Dans cette définition, nous avons présenté m et M comme des constantes, mais il peut être plus intéressant de les considérer comme des fonctions qui à $\vec{x} \in \mathcal{D}$ associent des éléments de $\overset{\circ}{\mathcal{I}}$ respectivement inférieur et supérieur à la racine de $y \mapsto f(\vec{x}, y)$.

On voit assez simplement que l'inclusion $\mathscr{L} + \mu_c \subseteq \mathcal{R}ec(\mathbb{R})$ est vraie : l'étape que nous avons faite pour trouver les bornes d'un compact contenant la racine ne sont plus nécessaires, la fin de la preuve reste la même.

Proposition 3.24 $\mathscr{L} + \mu_c \subseteq \mathcal{R}ec(\mathbb{R})$

Démonstration : Dans la preuve de la proposition 3.13, nous avons procédé en deux temps : un pour trouver un compact sur lequel se trouvait la racine, un pour calculer la racine sur ce compact. Dans le cas du schéma μ_c, la première étape est déjà faite, il ne reste que la deuxième à effectuer. Il suffit pour cela de reprendre la preuve en question. □

Nous voudrions également avoir

$$\mathcal{R}ec(\mathbb{N}) \subseteq DP(\mathscr{L} + \mu_c).$$

Le travail effectué pour prouver l'inclusion $\mathcal{R}ec(\mathbb{N}) \subseteq DP(\mathscr{L}+!\mu)$ cependant ne semble pas pouvoir être adaptée pour ce schéma. En effet, la fonction d'Ackermann fait partie des fonctions récursives discrètes, donc une extension de cette fonction aux réels appartient à $\mathscr{L}+!\mu$. Néanmoins, cette fonction n'appartient pas à $\mathscr{L}$ (on peut le déduire de la propriété $DP(\mathscr{L}) = \mathscr{E}(\mathbb{N})$). Pour obtenir cette fonction à l'aide de fonctions de $\mathscr{L}$ et

de l'opérateur μ_c, il faut calculer un compact encadrant les valeurs de cette fonction qui est connue pour croître plus vite que toute fonction élémentaire. Il faut déjà avoir une fonction qui croit au moins aussi vite que la fonction d'Ackermann, et on ne peut l'obtenir que par un schéma μ.

4 Lien entre analyse récursive et fonctions $\mathbb{R}$-récursives

La simplicité n'a pas besoin d'être simple, mais du complexe resserré et synthétisé.

(Alfred Jarry)

Nous avons vu dans le chapitre précédent une caractérisation des fonctions récursives discrètes par une classe de fonctions définies sur des intervalles de $\mathbb{R}$. Le paragraphe 2.2.2 présentait une caractérisation des fonctions élémentaires et des fonctions appartenant à la hiérarchie de Grzegorczyk par des classes $\mathscr{L}$ et $\mathscr{L}_n$ de fonctions définies sur les réels. Ces résultats sont insatisfaisants dans le sens où comparer des ensembles bien connus de fonctions sur les entiers et des classes de fonctions sur les réels apporte une idée de ce que contiennent ces classes réelles mais en perdant ce qui en fait la spécificité. Un résultat désirable serait de pouvoir comparer ces classes aux fonctions calculables par des machines de type 2 ou par d'autres machines qui manipulent des réels. Dans ce chapitre, nous montrons des classes basées sur $\mathscr{L}$, $\mathscr{L}_n$ et $\mathscr{L}+!\mu$ qui vont caractériser les classes de fonctions calculables au sens de l'analyse récursive.

Des résultats de Manuel Campagnolo, José Félix Costa et Cristopher Moore nous montrent que

$$DP(\mathscr{L}) = \mathscr{E}(\mathbb{N})$$
$$DP(\mathscr{L}_n) = \mathscr{E}_n(\mathbb{N}).$$

Nous avons vu dans le chapitre précédent que

$$DP(\mathscr{L}+!\mu) = \mathcal{R}ec(\mathbb{N}).$$

$$\begin{array}{ccc} \mathscr{L}+!\mu & & \mathcal{R}ec(\mathbb{R}) \\ \downarrow{\scriptstyle DP} & & \downarrow{\scriptstyle DP} \\ DP(\mathscr{L}+!\mu) & = & \mathcal{R}ec(\mathbb{N}) \end{array}$$

FIG. 4.1: Lien entre $\mathscr{L}+!\mu$ et $\mathcal{R}ec(\mathbb{R})$

FIG. 4.2: Liens entre $\mathscr{L}$ et $\mathscr{E}(\mathbb{N})$, et entre $\mathscr{L}_n$ et $\mathscr{E}_n(\mathbb{R})$

Nous connaissons en fait déjà des résultats sans partie discrète : Manuel Campagnolo, José Félix Costa et Cristopher Moore ont en effet montré que : $\mathscr{L} \subseteq \mathscr{E}(\mathbb{R})$, $\mathscr{L} \subseteq \mathscr{E}_n(\mathbb{R})$. Et nous avons obtenu l'inclusion $\mathscr{L}+!\mu \subseteq \mathcal{R}ec(\mathbb{R})$.

Notre objectif est dans ce chapitre de créer un opérateur permettant de traduire les résultats précédents en des résultats sur des fonctions continues. Nous allons tout d'abord chercher à compléter le lien manquant dans la figure 4.1, c'est à dire à avoir une caractérisation algébrique des fonctions récursivement calculables. Puis nous vérifierons que cet opérateur permet aussi de traduire les caractérisations des fonctions élémentaires discrètes en une caractérisation des fonctions élémentairement calculables, et de même obtenir pour la hiérachie de Grzegorczyk une caractérisation algébrique non plus uniquement de fonctions $\mathbb{N} \to \mathbb{N}$ mais de fonctions $\mathbb{R} \to \mathbb{R}$. Nous voulons en quelque sorte compléter les schémas des figures 4.1 et 4.2, ce qui peut être résumé par les méta-théorèmes suivants :

Méta-théorème 2 *Il existe un opérateur L tel que*

$$\mathscr{L}+!\mu + L = \mathcal{R}ec(\mathbb{R})$$

Méta-théorème 3 *Cet opérateur L est tel que*

$$\mathscr{L} + L = \mathscr{E}(\mathbb{R})$$
$$\forall n \geq 3, \mathscr{L}_n + L = \mathscr{E}_n(\mathbb{R})$$

où la notation $\mathscr{C} + L$ avec $\mathscr{C}$ une algèbre de fonctions représente l'algèbre de fonctions obtenue en ajoutant L comme opérateur de clôture.

Ces résultats peuvent être considérés comme un pas vers une thèse de Church-Turing pour les fonctions sur les réels, ou moins ambitieusement, une caractérisation de ce qui est raisonnable pour les fonctions sur les réels. En effet, les fonctions produites par l'analyse récursive sont physiquement réalisables, tandis que les fonctions $\mathbb{R}$-récursives par exemple semblent ne pas respecter les principes fondamentaux de la thermodynamique comme nous en discutons brièvement en section 2.2.

L'idée fondamentale de la caractérisation des fonctions récursivement calculables et des sous classes de l'analyse récursive par des sous-classes de fonctions $\mathbb{R}$-récursives est

que ce qui différencie les machines de type 2 des machines de type 1 est un processus de limite : le nombre réel représenté par un ruban infini est la limite d'une suite de Cauchy. Ce processus de limite est cependant limité : il ne travaille que sur des suites convergeant rapidement ; et les résultats de Jerzy Mycka et José Félix Costa [MC05, MC04] montrent qu'un schéma général de limite est aussi puissant que le schéma $\mu_{\mathbb{R}}$ que nous voulions abolir.

Ces résultats sont le fruit d'un travail avec Olivier Bournez. Le méta-théorème 3 est présenté dans [BH04, BH05a]. Et le méta-théorème 2 vient de [BH06].

4.1 Définitions

Nous allons chercher pour opérateur L un opérateur de limite particulier. Notre opérateur de limite doit n'être défini que si la limite existe, est suffisamment lisse (de classe $\mathscr{C}^2$) et que cette limite ne croît pas nettement plus que la fonction d'origine.

On appelle polynôme en $x \in \mathbb{R}$ une fonction de la forme $\beta(x) = \sum_{i=0}^{n} a_i x^i$. On appelle polynôme en $\vec{x} \in \mathbb{R}^{k+1}$ une fonction de la forme $\beta(\vec{x}) = \sum_{i=0}^{n} \alpha_i x_{k+1}^i$ où les α_i sont des polynômes en $(x_1, \ldots, x_k)$. Les polynômes sont des fonctions dont nous connaissons l'appartenance à la classe $\mathscr{L}$ et donc aux autres classes que nous avons définies et que nous étudions, qui contiennent $\mathscr{L}$.

Voici notre candidat pour être l'opérateur L du méta-théorème :

Définition 4.1 (LIM_w) *Soit $f : \mathbb{R} \times \mathcal{D} \to \mathbb{R}^l$ où $\mathcal{D} \subseteq \mathbb{R}^{k+1}$ telle qu'il existe des polynômes $K : \mathcal{D} \to \mathbb{R}$ et $\beta : \mathcal{D} \to \mathbb{R}$ pour lesquels*

$$\forall t \geq \|\vec{x}\|, \left\| \frac{\partial f}{\partial t}(t, \vec{x}) \right\| \leq K(\vec{x}) 2^{-t\beta(\vec{x})}.$$

Soit I un produit d'intervalles fermés dans $\mathbb{R}^k$ sur lequel $\beta(\vec{x}) > 0$. Alors, pour tout $\vec{x} \in I$, $\lim_{t \to +\infty} f(t, \vec{x})$ existe. Si la fonction

$$F : \left\{ \begin{array}{lcl} I & \to & \mathbb{R}^l \\ \vec{x} & \mapsto & \lim_{t \to +\infty} f(t, \vec{x}) \end{array} \right.$$

est de classe $\mathscr{C}^2$, on définit $\mathrm{LIM}_w(f, K, \beta) = F$.

En d'autres termes, on impose que la fonction f soit dérivable, que cette dérivée soit bornée par une fonction qui décroît rapidement. La condition de borne sur la dérivée assure l'existence de la limite comme le montre la proposition suivante :

Proposition 4.2 *Soit $f : \mathbb{R} \to \mathbb{R}$ une fonction dérivable et $t_0 \in \mathbb{R}$ vérifiant*

$$\forall t \geq t_0, \left\| f'(t) \right\| \leq \lambda \mathrm{e}^{-\mu t}.$$

Alors, la fonction f converge en $+\infty$.

Démonstration : Soit f une fonction qui vérifie les conditions de la proposition. Soit $\varepsilon > 0$. Il existe $t_\varepsilon > t_0$ tel que $\frac{\lambda}{\mu}\mathrm{e}^{-\mu t_\varepsilon} < \varepsilon$. Or

$$\begin{aligned}\forall t \in \mathbb{R}, h > 0, |f(t+h) - f(t)| &\leq \int_t^{+\infty} |f'(u)| \,\mathrm{d}u \\ &\leq \frac{\lambda}{\mu}\mathrm{e}^{-\mu t}.\end{aligned}$$

Donc

$$\forall x, y \in]t_\varepsilon, +\infty[, |f(x) - f(y)| < \varepsilon.$$

Ce qui montre que f vérifie le critère de convergence de Cauchy pour les applications, elle converge donc. □

Définissons maintenant des classes de fonctions utilisant cet opérateur de limite.

Définition 4.3 ($\mathscr{L}^\star_{!\mu}$)

$$\mathscr{L}^\star_{!\mu} = [0, 1, U, \theta_3; \mathrm{COMP}, \mathrm{CLI}, \mathrm{UMU}, \mathrm{LIM}_w]$$

Et de façon plus générale :

Définition 4.4 *Étant donnée une classe $\mathcal{C} = [\mathcal{F}; \mathcal{O}]$, on notera $\mathcal{C}^\star = [\mathcal{F}; \mathcal{O}, \mathrm{LIM}_w]$.*

En particulier pour les classes $\mathscr{L}^\star$ et les $\mathscr{L}^\star_n$.

Définition 4.5 ($\mathscr{L}^\star$ et $\mathscr{L}^\star_n$)

$$\mathscr{L}^\star = [0, 1, -1, U, \theta_3; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w]$$

$$\mathscr{L}^\star_n = [0, 1, -1, U, \theta_3, \bar{E}_{n-1}; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w]$$

Il est à remarquer que nous n'avons pas inclus π dans ces classes alors que π est explicitement une fonction de base de $\mathscr{L}$ et $\mathscr{L}_n$. Nous verrons dans l'exemple 4.12 que π appartient quand même à ces classes étoilées.

Définition 4.6 (Module de continuité uniforme) *Étant donnée une fonction $f : \mathbb{R}^k \to \mathbb{R}^l$ définie sur un fermé. Une fonction $M : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ est un module de continuité uniforme de f si*

$$\forall K, i \in \mathbb{N}, \forall x, y \in [-K, K]^k, \|x - y\| < 2^{-M(K,i)} \Rightarrow \|f(x) - f(y)\| < 2^{-i}.$$

Si une fonction est définie sur un domaine compact, un module de continuité (voir définition 2.14) de cette fonction donne immédiatement un module de continuité uniforme.

4.2 Propriétés

L'opérateur LIM_w possède certaines propriétés qui nous seront utiles. Par exemple, les fonctions produites par LIM_w sont de classe $\mathscr{C}^2$, sont définies sur des intervalles fermés ou des produits d'intervalles fermés (mais pas nécessairement compacts). Nous montrerons également comment générer la fonction *inv* et la constante π à l'aide de cet opérateur, et sans UMU, prouvant ainsi l'inclusion stricte de $\mathscr{L}$ dans $\mathscr{L}^\star$.

Proposition 4.7 LIM_w *préserve* $\mathscr{C}^2$

Démonstration : Par définition, on n'accepte de prendre LIM_w d'une fonction que si le résultat est de classe $\mathscr{C}^2$. □

Proposition 4.8 *Si f est définie sur un produit d'intervalles fermés, qu'il existe β et K tels que $\mathrm{LIM}_w(f, K, \beta)$ soit définie, alors $\mathrm{LIM}_w(f, K, \beta)$ est définie sur un produit d'intervalles fermés.*

Démonstration : La condition de borne sur la dérivée de $f : \mathbb{R} \times \mathcal{D}$ par rapport à la première variable assure l'existence de la limite sur $\mathcal{D}$. Donc $\mathrm{LIM}_w(f, K, \beta)$ est définie sur $\mathcal{D}$ qui est un produit d'intervalles fermés, ou bien sur un produit d'intervalles fermés inclus dans $\mathcal{D}$ si l'on veut se restreindre à un domaine plus petit comme nous y autorise la définition. □

La borne imposée sur la dérivée pour définir LIM_w donne une idée de la vitesse de convergence et donc de l'erreur commise en prenant la valeur calculée pour un certain t pour approcher la limite. Le lemme suivant montre une borne de cette erreur.

Lemme 4.9 *Soit $F : \mathbb{R} \times \mathcal{D} \to \mathbb{R}^l$ une fonction de classe $\mathscr{C}^1$ avec $\mathcal{D} \subset \mathbb{R}^k$. Soient β et K des polynômes sur $\mathbb{R}^k$.*

Si pour tout $\vec{x} \in \mathcal{D}$, et tout $t \in \mathbb{R}$ on a

$$\left\| \frac{\partial F}{\partial t}(t, \vec{x}) \right\| \leq K(\vec{x}) 2^{-t\beta(\vec{x})},$$

alors, considérant $\Delta \subset \mathcal{D}$ un ensemble sur lequel $\beta(\vec{x}) > 0$, pour tout $\vec{x} \in \Delta$, $F(t, \vec{x})$ a une limite $L(\vec{x})$ quand t tend vers $+\infty$, et

$$\|F(t, \vec{x}) - L(\vec{x})\| \leq \frac{K(\vec{x}) 2^{-t\beta(\vec{x})}}{\beta(\vec{x})}.$$

Proposition 4.10 ([CMC00b, Cam01]) *Si f vérifie pour des entiers d, A et B tels que*

$$\forall x, \|f(x)\| < A \exp^{[d]}(B\|x\|),$$

et que l'on peut définir $\mathrm{LIM}_w(f, K, \beta)$*. Alors,* $\mathrm{LIM}_w(f, k, \beta)$ *vérifie une inégalité de la même forme :*

$$\|\mathrm{LIM}_w(f, K, \beta)\| < A' \exp^{[d']}(B'\|x\|).$$

Démonstration : Soit $h = \mathrm{LIM}_w(f, K, \beta)$.
Du lemme 4.9, on déduit que

$$\forall \vec{x}, \forall t \geq \|\vec{x}\|; \|h(\vec{x}) - f(t, \vec{x})\| \leq \frac{K(\vec{x})}{\beta(\vec{x})} 2^{-\beta(\vec{x})t}.$$

En particulier, il existe κ, C, D tels que pour $t = \|\vec{x}\|$,

$$\begin{aligned}
\|h(\vec{x})\| &\leq \|f(t, \vec{x})\| + \frac{K(\vec{x})}{\beta(\vec{x})} 2^{-\beta(\vec{x})t} \\
&\leq \|f(\|\vec{x}\|, \vec{x})\| + \frac{K(\vec{x})}{\beta(\vec{x})} 2^{-\beta(\vec{x})\|\vec{x}\|} \\
&< A \exp^{[d]}(B\|\vec{x}\|) + \frac{K(\vec{x})}{\beta(\vec{x})} \exp(-\beta(\vec{x})\|\vec{x}\|) \\
&< A \exp^{[d]}(B\|\vec{x}\|) + \kappa \exp(C\|\vec{x}\|) \exp(D\|\vec{x}\|) \\
&< A \exp^{[d]}(B\|\vec{x}\|) + \kappa \exp(C\|\vec{x}\|) \exp(D\|\vec{x}\|) \\
&< A' \exp^{[d]}(B'\|\vec{x}\|)
\end{aligned}$$

avec par exemple $A' = A + \kappa$ et $B' = B + C + D$. □

Montrons maintenant quelques exemples d'utilisation de cet opérateur de limite. Tout d'abord, la fonction inverse. Dans le cas des fonctions $\mathbb{R}$-récursives, on obtient la fonction inverse à l'aide du schéma $\mu_{\mathbb{R}}$. Dans le cas des fonctions récursivement calculables, la fonction inverse est obtenue gratuitement à l'aide de l'encodage : prendre l'inverse d'un rationnel dont on connaît le numérateur et le dénominateur est trivial. Dans le cas de $\mathscr{L}+!\mu$, on n'a pas *a priori* la fonction inverse, mais on peut l'obtenir à l'aide de l'opérateur de limite.

Exemple 4.11 *Soit* $E(t, x)$ *définie comme étant la primitive de* $\exp(-tx)$ *valant* 0 *en* $\vec{0}$*. E appartient à la classe* $\mathscr{L}$ *car* LI *permet de simuler la primitive. E vérifie alors*

$$E(t, x) = \begin{cases} \frac{1-\mathrm{e}^{-tx}}{x} & \text{for } x \neq 0 \\ t & \text{for } x = 0 \end{cases}$$

Et pour $x > 0$*,* $\lim_{t \to +\infty} E(t, x) = \frac{1}{x}$*. De plus, pour tout intervalle fermé inclus dans* $]0, +\infty[$*, il existe un polynôme* K *tel que* $\exists n_0, \forall t > n_0, \forall x, \left|\frac{\partial E}{\partial t}(t, x)\right| < K(x) \exp(-tx)$*. Et donc on peut définir* $inv : x \to 1/x$ *sur n'importe quel intervalle fermé inclus dans* $]0, +\infty[$ *comme étant* $\mathrm{LIM}_w(E, K, x \mapsto x)$*.*

Remarquons que ce résultat montre une fonction appartenant à $\mathscr{L}^\star$ mais pas à $\mathscr{L}$ qui ne contient que des fonctions totales.

La définition de $\mathscr{L}^\star$ ne fait pas apparaître π parmi les fonction de base. π est cependant bien dans la classe.

Exemple 4.12 *La fonction $x \mapsto 1 + x^2$ appartient à $\mathscr{L}$ et est à valeurs dans $[1, +\infty]$. La fonction $a : x \mapsto \frac{1}{1+x^2}$ appartient donc à $\mathscr{L}^\star$ (en composant avec la fonction inv définie sur $[1, +\infty[$). Et donc la primitive de a valant 0 en 0 y appartient également, or il s'agit de* arctan. *Finalement, $\pi = 4\arctan(1) \in \mathscr{L}^\star$*

On peut donc écrire

$$[0, 1, -1, U, \theta_3; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w] = [0, 1, -1, \pi, U, \theta_3; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w].$$

De ces deux exemples, on peut déduire un résultat intéressant :

Proposition 4.13

$$\mathscr{L} \subsetneq \mathscr{L}^\star$$
$$\forall n \geq 3, \mathscr{L}_n \subsetneq \mathscr{L}_n^\star$$

4.3 Résultats

Le premier et principal résultat de ce chapitre est l'égalité entre $\mathcal{R}ec(\mathbb{R})$ et $\mathscr{L}^\star_{!\mu}$ pour les fonctions $\mathscr{C}^2$ définies sur des produits d'intervalles fermés.

Nous montrerons également que les fonctions de $\mathscr{L}^\star$ sont exactement les fonctions $\mathscr{C}^2$ définies sur des fermés de $\mathscr{E}(\mathbb{R})$, et que de même, on peut caractériser les fonctions de $\mathscr{E}_n(\mathbb{R})$ à l'aide des classes $\mathscr{L}_n^\star$

4.3.1 Caractérisation de $\mathcal{R}ec(\mathbb{R})$

4.3.1.a Résultat

Le résultat de caractérisation de l'ensemble des fonctions récursivement calculables de façon algébrique est énoncé dans le théorème suivant.

Théorème 4.14 *Soit f une fonction de classe $\mathscr{C}^2$, définie sur un produit d'intervalles compacts à bornes rationnelles.*

$$f \in \mathscr{L}^\star_{!\mu} \Leftrightarrow f \in \mathcal{R}ec(\mathbb{R}).$$

Ce résultat est décomposé en deux propositions pour expliciter l'organisation des preuves. Les propositions 4.15 et 4.16 prouvent respectivement le sens direct ($\mathscr{L}^\star_{!\mu}$ est inclus dans $\mathcal{R}ec(\mathbb{R})$) et le sens indirect du théorème ($\mathscr{L}^\star_{!\mu}$ contient $\mathcal{R}ec(\mathbb{R})$) avec les bonnes hypothèses sur les fonctions.

4.3.1.b Sens direct de la preuve

Proposition 4.15 *Toute fonction appartenant à $\mathscr{L}^{\star}_{!\mu}$, définie sur un compact à bornes rationnelles appartient à $\mathcal{R}ec(\mathbb{R})$*

Démonstration : La preuve se fait par récurrence structurelle.

Nous avons déjà vu que 0, 1, U et θ_3 appartiennent à $\mathcal{R}ec(\mathbb{R})$, et que COMP, CLI et UMU préservent $\mathcal{R}ec(\mathbb{R})$.

Nous devons donc uniquement prouver que LIM_w préserve également $\mathcal{R}ec(\mathbb{R})$.

Soit f une fonction récursivement calculable. Il existe une fonctionnelle récursive ϕ qui la calcule. On suppose qu'il existe des polynômes K et β tels que $\mathrm{LIM}_w(f, K, \beta)$ existe. On veut alors montrer que $g = \mathrm{LIM}_w(f, K, \beta)$ est récursivement calculable.

On suppose pour la suite que f est définie sur $\mathbb{R} \times \mathcal{I}$ où $\mathcal{I}$ est un intervalle fermé de $\mathbb{R}$, et que $\beta(x) > 0$ pour $x \in \mathcal{I}$. Le cas général s'obtient facilement.

$\beta(x)$ étant un polynôme, il est calculable. Étant donné x, on peut donc construire un $N(x)$ tel que $1/\beta(x) < N(x)$.

Soit $(x_n) \rightsquigarrow x$. Pour tout $i, j \in \mathbb{N}$, on a $|\nu_{\mathbb{Q}}(\phi((i, x_n), j)) - f(i, x)| < 2^{-j}$.

D'autre part $|g(x) - f(i, x)| \leq KN2^{-\beta(x)i}$ d'après le lemme 4.9.

On peut donc estimer la différence entre $\nu_{\mathbb{Q}}(\phi((i, x_n), j))$ et la limite $g(x)$.

$$|\nu_{\mathbb{Q}}(\phi((i, x_n), j)) - g(x)| < 2^{-j} + KN2^{-\beta(x)i}$$

Prenons $j' \geq j + 1$ et $i' \geq N \times (j + 1 + \lceil \lg(KN) \rceil)$. On a alors $2^{-j'} \geq \frac{1}{2}2^{-j}$ et $KN2^{-\beta(x)i'} \leq \frac{1}{2}2^{-j}$. Donc la fonctionnelle ψ définie ci-après calcule la fonction g :

$$\psi : (x_n), j \mapsto \phi\left((\max(X, KN(j + 1 + \lceil \lg(KN) \rceil)), x_n), j + 1\right)$$

En effet, pour tout j, $\|\nu_{\mathbb{Q}}(\psi(x_n, j)) - g(x)\| \leq 2^{-j}$. □

4.3.1.c Sens indirect de la preuve

Nous allons maintenant montrer le sens indirect du théorème 4.14 : pour les fonctions $\mathscr{C}^2$ sur un compact l'appartenance à $\mathcal{R}ec(\mathbb{R})$ implique l'appartenance à $\mathscr{L}^{\star}_{!\mu}$.

Proposition 4.16 *Notons $\mathscr{C}^2(\mathbb{K})$ l'ensemble des fonctions définies sur un compact et de classe $\mathscr{C}^2$ sur ce compact.*

$$\mathcal{R}ec(\mathbb{R}) \cap \mathscr{C}^2(\mathbb{K}) \subseteq \mathscr{L}^{\star}_{!\mu}$$

Nous allons en fait montrer une proposition plus générale qui stipule que si A est une classe de fonctions discrètes et A contient $\mathscr{E}(\mathbb{N})$, et $\mathscr{A}$ une classe de fonctions sur les réels telle que $A \subseteq DP(\mathscr{A})$, alors $A(\mathbb{R}) \subseteq \mathscr{A}^{\star}$ pour les fonctions $\mathscr{C}^2$ définies sur un compact.

Définition 4.17 *Soit A une classe de fonctions sur les entiers.*

Une fonction $f : \mathbb{R}^k \to \mathbb{R}$ appartient à $A(\mathbb{R})$ si il existe une fonctionnelle $\phi \in A$ telle que pour tout $\vec{x} \in \mathcal{D}om(f)$, pour toute suite $X \rightsquigarrow x$, alors la suite $(\phi(X, j))_j \rightsquigarrow f(\vec{x})$.

Une fonction $f : \mathbb{R}^k \to \mathbb{R}^l$ avec $l > 1$ appartient à $A(\mathbb{R})$ si toutes ses projections appartiennent à $A(\mathbb{R})$.

Nous montrerons que la proposition suivante implique la proposition 4.16 en prenant $A = \mathcal{R}ec(\mathbb{N})$ et $\mathscr{A} = \mathscr{L}+!\mu$. Cette proposition nous permettra par la suite de montrer les inclusions de $\mathscr{E}(\mathbb{R})$ dans $\mathscr{L}^\star$ et des $\mathscr{E}_n(\mathbb{R})$ dans les classes $\mathscr{L}_n^\star$.

Proposition 4.18 *Soit A une classe de fonctions sur les entiers, close par composition et contenant $\mathscr{E}(\mathbb{N})$.*

Soit $\mathscr{A}$ une classe de fonctions sur les réels close par composition, prise de primitive et contenant $\mathscr{L}$.

Si $A \subseteq DP(\mathscr{A})$, alors pour les fonctions de classe $\mathscr{A}^2$, définies sur un compact et dont les dérivées ont un module de continuité dans A,

$$A(\mathbb{R}) \subseteq \mathscr{A} + \mathrm{LIM}_w.$$

Résultats intermédiaires Le lemme suivant montre que la dérivée d'une fonction $\mathscr{C}^2$ sur un compact calculable est calculable. Dans le cas où $A = \mathcal{R}ec(\mathbb{N})$, on retrouve [Wei00, Corollaire 6.4.8-2]. Remarquons que continûment dérivable ne suffit pas, la condition $\mathscr{C}^2$ est nécessaire comme le montre [Myh71] qui exhibe une fonction $\mathscr{C}^1$ sur un compact, récursivement calculable mais dont la dérivée n'est pas récursivement calculable.

Lemme 4.19 *Soit une fonction f de classe $\mathscr{C}^2$ définie sur un compact. Si $f \in A(\mathbb{R})$ pour un A stable pour la composition et contenant $\mathscr{E}(\mathbb{N})$, alors les dérivées partielles de f appartiennent également à $A(\mathbb{R})$*

Démonstration : Supposons f définie sur $[a, b]$ à valeurs dans $\mathbb{R}$. Le cas général peut facilement s'en déduire. Nous devons donc montrer que la dérivée de f est calculable au sens de l'analyse récursive en utilisant uniquement des fonctions de A.

f est de classe $\mathscr{C}^2$, donc f'' est continue et définie sur le compact $[a, b]$ donc f'' est bornée sur $[a, b]$ par une constante réelle M. À l'aide du théorème des valeurs intermédiaires, on obtient

$$\forall x, y \in [a, b] |f'(x) - f'(y)| < M|x - y|.$$

Soit $x \in [a, b]$. Soit $i \in \mathbb{N}$. Calculons une approximation de $f'(x)$ à 2^{-i} près :

Calculer un n tel que $M2^{-n} \leq 2^{-i-1}$
Construire y_1 un rationnel approchant $f(x)$ à 2^{-i-n-2}
Construire y_2 un rationnel approchant $f(x + 2^{-n})$ à 2^{-i-n-2}
Renvoyer $z = (y_2 - y_1)/2^{-n}$

On peut vérifier que z approche $f'(x)$ à 2^{-i} près. En effet, le théorème des valeurs intermédiaires affirme l'existence de $\chi \in [x, x+2^{-n}]$ tel que $f'(\chi) = \frac{f(x+2^{-n})-f(x)}{2^{-n}}$. Et l'erreur commise en approchant $f'(x)$ par z est bornée comme suit :

$$\begin{aligned}
\left|z - f'(x)\right| &= \left|\frac{y_2 - y_1}{2^{-n}} - f'(x)\right| \\
&\leq \left|\frac{y_2 - y_1}{2^{-n}} - f'(\chi)\right| + \left|f'(\chi) - f'(x)\right| \\
&\leq \left|\frac{y_2 - f(x+2^{-n})}{2^{-n}} - \frac{y_1 - f(x)}{2^{-n}}\right| + M2^{-n} \\
&\leq \left|\frac{y_2 - f(x+2^{-n})}{2^{-n}}\right| + \left|\frac{y_1 - f(x)}{2^{-n}}\right| + M2^{-n} \\
&\leq 2^{-i-2} + 2^{-i-2} + 2^{-i-1} \\
&\leq 2^{-i}
\end{aligned}$$

Outre le calcul de f, on n'a utilisé que la fonction exponentielle, la multiplication et la composition de fonctions. Donc ce calcul montre que $f' \in A(\mathbb{R})$. □

Nous devons également montrer que ces dérivées en plus d'être calculables ont un module de continuité dans A. Pour ce faire, nous nous contentons de montrer que, pour A vérifiant certaines conditions, toute fonction de classe $\mathscr{C}^1$ calculable dans $A(\mathbb{R})$ possède un module de continuité dans A. Dans le cas où $A = \mathcal{R}ec(\mathbb{N})$, on retrouve, ce lemme s'écrit « toute fonction récursivement calculable, $\mathscr{C}^1$ sur un compact possède un module de continuité calculable. »

Lemme 4.20 *Soit f une fonction de classe $\mathscr{C}^1$ définie sur un compact. Soit A une classe de fonctions discrètes contenant $\mathscr{E}(\mathbb{N})$. Si $f \in A(\mathbb{R})$, alors f possède un module de continuité dans A.*

Démonstration : Considérons la différentielle d'une telle fonction f. Étant continue et définie sur un compact, sa norme possède un maximum. Soit m un entier majorant la norme de la différentielle.

La fonction $i \mapsto m+i$ est un module de continuité de f en application du théorème de la moyenne. Il s'agit d'une fonction linéaire qui à ce titre appartient à $\mathscr{E}(\mathbb{N})$ et donc à A. □

La proposition 4.18 montre que pour une fonction f de classe $\mathscr{C}^2$ définies sur un compact,

$$f \in A(\mathbb{R}) \text{ et } f' \text{ a un module de continuité dans } A \Rightarrow f \in \mathscr{A}^\star.$$

En particulier,

$$f \in \mathcal{R}ec(\mathbb{R}) \text{ et } f' \text{ a un module de continuité dans } \mathcal{R}ec(\mathbb{N}) \Rightarrow f \in \mathscr{L}^\star_{!\mu}.$$

Si l'on considère une fonction $f \in \mathcal{R}ec(\mathbb{R})$ de classe $\mathscr{C}^2$ définie sur un compact, d'après le lemme 4.19, f' appartient à $\mathcal{R}ec(\mathbb{R})$. Le lemme 4.20 appliqué à f' montre que f' possède un module de continuité dans $\mathcal{R}ec(\mathbb{N})$. On en déduit donc que pour toute fonction f de classe $\mathscr{C}^2$ définie sur un compact,

$$f \in \mathcal{R}ec(\mathbb{R}) \Rightarrow f \in \mathscr{L}^{\star}_{!\mu}.$$

Preuve de la proposition 4.18 Les résultats précédents sont donc suffisants pour montrer la proposition 4.16.

Il nous reste alors à prouver la proposition 4.18. Pour ce faire, nous allons utiliser les lemmes suivants. Le lemme 4.21 peut être vu comme une généralisation du lemme 3.16 qui montrait que toute fonction de $\mathscr{L}$ possède une extension « canonique » également dans $\mathscr{L}$. Ici, nous allons montrer que toute fonction dans $\mathscr{A}$ possède une extension dans $\mathscr{A}$ canonique pour des cubes aussi petits que nécessaire et que cette extension soit suffisamment lisse (on voudra y appliquer l'opérateur de limite).

Lemme 4.21 *Soit $\mathscr{A}$ une algèbre de fonctions réelles contenant $\mathscr{E}(\mathbb{R})$, stable pour la composition et la prise de primitive. Soit $\varepsilon : \mathbb{R} \to \mathbb{R}$ une fonction de $\mathscr{A}$ décroissante et à valeurs strictement positives et telle que $\varepsilon(x) \xrightarrow[x\to+\infty]{} 0$.*

Étant donnée $f : \mathbb{R}^2 \to \mathbb{R}^l$ une fonction de $\mathscr{A}$. Il existe $F : \mathbb{R}^2 \to \mathbb{R}^l$ dans $\mathscr{A}$ telle que

1. *$\forall i \in \mathbb{N}$, $\forall x \in \mathbb{Z}\varepsilon(\lfloor i \rfloor)$, $F(i,x) = f(i,x)$*
2. *$\forall i \in \mathbb{N}$, $\forall x \in \mathbb{R}$, en notant $\lfloor x \rfloor_{\varepsilon_i} = \max\{y \in \mathbb{Z}\varepsilon(\lfloor i \rfloor); y < x\}$,*

$$\|F(i,x) - f(i, \lfloor x \rfloor_{\varepsilon_i})\| \leq \|f(i, \lfloor x \rfloor_{\varepsilon_i} + \varepsilon(\lfloor i \rfloor)) - f(i, \lfloor x \rfloor_{\varepsilon_i})\|$$

3. *$\forall i \in \mathbb{R}^*_+$, $\forall x \in \mathbb{R}$, en notant*

$$\begin{aligned} \varepsilon_i &= \varepsilon(\lfloor i \rfloor) \\ \varepsilon_{i+1} &= \varepsilon(\lfloor i+1 \rfloor) \\ \lfloor x \rfloor_{\varepsilon_i} &= \max\{y \in \mathbb{Z}\varepsilon_i, y < x\} \text{ (comme ci-dessus),} \end{aligned}$$

$$\begin{aligned} \left\| \frac{\partial F}{\partial i}(i,x) \right\| \leq & 5\|f(\lfloor i+1 \rfloor, \lfloor x \rfloor_{\varepsilon_i}) - f(\lfloor i \rfloor, \lfloor x \rfloor_{\varepsilon_i})\| \\ & + 25\|f(\lfloor i \rfloor, \lfloor x \rfloor_{\varepsilon_i} + \varepsilon_i) - f(\lfloor i \rfloor, \lfloor x \rfloor_{\varepsilon_i})\| \\ & + 25\|f(\lfloor i+1 \rfloor, \lfloor x \rfloor_{\varepsilon_{i+1}} + \varepsilon_{i+1}) - f(\lfloor i+1 \rfloor, \lfloor x \rfloor_{\varepsilon_{i+1}})\|. \end{aligned}$$

Démonstration : Ce lemme est une extension du lemme 3.16. Les deux premières conditions représentent une certaine canonicité sur un petit cube, la troisième impose que la fonction soit suffisamment lisse. Nous allons réutiliser les fonctions ω et *int* qui y ont été définies :

$$\zeta = \frac{3\pi}{2}$$
$$\omega : x \mapsto \zeta\theta_3(\sin(2\pi x))$$
$$\Omega : x \mapsto \int_0^x \omega(t)\,\mathrm{d}t$$
$$int : x \mapsto \Omega(x - \frac{1}{2})$$

Tout d'abord, nous allons fabriquer une fonction G dont le comportement sera connu et lisse suivant la deuxième variable.

Étant donnée $f : \mathbb{R}^2 \to \mathbb{R}^l \in \mathscr{A}$, on définit $\Delta : \mathbb{R}^2 \to \mathbb{R}^l$ comme étant le taux de variation de f par rapport à la deuxième variable sur un intervalle de longueur $\varepsilon(i)$:

$$\Delta(i, x) = f(i, x + \varepsilon(i)) - f(i, x).$$

On vérifie que pour tout i, x, on a

$$\frac{\omega(x/\varepsilon(i))}{\varepsilon(i)}\Delta(i, \varepsilon(i)int(x/\varepsilon(i))) = \begin{cases} 0 & \text{si } x - \lfloor x \rfloor_{\varepsilon(i)} \geq \frac{1}{2}\varepsilon(i) \\ \frac{\omega(x/\varepsilon(i))}{\varepsilon(i)}\Delta(i, \varepsilon(i)\lfloor x/\varepsilon(i)\rfloor) & \text{sinon} \end{cases}$$

En effet, si $x - \varepsilon(i)\lfloor x/\varepsilon(i)\rfloor \geq \frac{1}{2}\varepsilon(i)$, $\omega(x/\varepsilon(i)) = 0$; sinon, $int(x/\varepsilon(i)) = \lfloor x/\varepsilon(i)\rfloor$.

Δ appartient bien à $\mathscr{A}$ puisque f, ω, int, -1, la multiplication, l'addition et la composition sont dans $\mathscr{A}$.

Soit G la solution du problème de Cauchy suivant :

$$\begin{cases} G(i, 0) & = f(i, 0) \\ \frac{\partial G}{\partial x}(i, x) & = \frac{\omega(t/\varepsilon(i))}{\varepsilon(i)}\Delta(i, \varepsilon(i)int(x/\varepsilon(i))) \end{cases}$$

G appartient à la classe $\mathscr{A}$ et vérifie $\forall j \in \mathbb{Z}, G(i, j\varepsilon(i)) = f(i, j\varepsilon(i))$ en effet, par hypothèse, $G(i, 0) = f(i, 0)$ et si $G(i, j\varepsilon(i)) = f(i, j\varepsilon(i))$, alors,

$$\begin{aligned} G(i, (j+1)\varepsilon(i)) &= f(i, j\varepsilon(i)) + \int_{j\varepsilon(i)}^{(j+1)\varepsilon(i)} \frac{\omega(t/\varepsilon(i))}{\varepsilon(i)}\Delta(i, int(t/\varepsilon(i)))\,\mathrm{d}t \\ &= f(i, j\varepsilon(i)) + \int_{j\varepsilon(i)}^{(j+\frac{1}{2})\varepsilon(i)} \frac{\omega(t/\varepsilon(i))}{\varepsilon(i)}\Delta(i, int(t/\varepsilon(i)))\,\mathrm{d}t \\ &= f(i, j\varepsilon(i)) + \int_{j\varepsilon(i)}^{(j+\frac{1}{2})\varepsilon(i)} \frac{\omega(t/\varepsilon(i))}{\varepsilon(i)}\Delta(i, \varepsilon(i)\lceil t/\varepsilon(i)\rceil)\,\mathrm{d}t \\ &= f(i, j\varepsilon(i)) + \Delta(i, \varepsilon(i)j)\int_{j\varepsilon(i)}^{(j+\frac{1}{2})\varepsilon(i)} \frac{\omega(t/\varepsilon(i))}{\varepsilon(i)}\,\mathrm{d}t \\ &= f(i, j\varepsilon(i)) + \Delta(i, \varepsilon(i)j)\int_k^{k+1} \omega(u)\,\mathrm{d}u \\ &= f(i, j\varepsilon(i)) + \Delta(i, j\varepsilon(i)) = f(i, (j+1)\varepsilon(i)) \end{aligned}$$

De plus, ω est positive, donc sur l'intervalle $[j\varepsilon(i), (j+1)\varepsilon(i)]$, $x \mapsto G(i,x)$ est monotone donc reste dans les bornes fixées par $f(i, j\varepsilon(i))$ et $f(i,(j+1)\varepsilon(i))$. Et pour $i \in \mathbb{N}$,

$$\|G(i,x) - f(i, \lfloor x \rfloor_{\varepsilon_i})\| \leq \|f(i, \lfloor x \rfloor_{\varepsilon_{i+1}}) - f(i, \lfloor x \rfloor_{\varepsilon_i})\|$$

Nous allons maintenant construire F qui reprendra le comportement de G suivant la deuxième variable et qui sera en plus connue et lisse par rapport à la première variable.
Soit maintenant ∇ un taux de variation de G par rapport à la première variable :

$$\nabla(i,x) = G(i+1,x) - G(i,x).$$

Définissons F comme la solution du problème de Cauchy suivant

$$\begin{cases} F(0,x) & = G(0,x) \\ \frac{\partial F}{\partial i}(i,x) & = \omega(i)\nabla(int(i),x) \end{cases}$$

De même que pour Δ, on a

$$\omega(i)\nabla(int(i),x) = \begin{cases} 0 & \text{si } i - \lfloor i \rfloor \geq \frac{1}{2} \\ \omega(i)\nabla(\lfloor i \rfloor, x) & \text{sinon} \end{cases}$$

Et de la même façon que pour G (mais orthogonalement), on vérifie que $\forall i \in \mathbb{Z}, x \in \mathbb{R}$,

$$F(i,x) = G(i,x) = f(i,x).$$

On vérifie également que

$$\|F(i,x) - f(i, \lfloor x \rfloor_{\varepsilon_i})\| \leq \|f(i, \lfloor x \rfloor_{\varepsilon_i} + \varepsilon_i) - f(i, \lfloor x \rfloor_{\varepsilon_i})\|$$

Il nous reste à vérifier que cette fonction F vérifie également la troisième condition du lemme, à savoir la borne sur la dérivée. Rappelons que $\zeta = \frac{3\pi}{2}$ est la norme de ω.

$$\begin{aligned} \frac{\partial F}{\partial i}(i,x) &= \omega(i)\nabla(int(i),x) \\ \left\| \frac{\partial F}{\partial i}(i,x) \right\| &\leq \omega(i)\|\nabla(\lfloor i \rfloor, x)\| \\ &\leq \zeta \|f(\lfloor i+1 \rfloor, x) - f(\lfloor i \rfloor, x)\| \\ &\leq 5 \|f(\lfloor i+1 \rfloor, x) - f(\lfloor i \rfloor, x)\| \end{aligned} \tag{4.1}$$

Constatons que $\frac{\partial F}{\partial i}$ est dérivable par rapport à la deuxième variable. Et

$$\frac{\partial^2 F}{\partial x \partial i}(i,x) = \omega(i) \left(\frac{\partial G}{\partial i}(int(i+1),x) - \frac{\partial G}{\partial i}(int(i),x) \right)$$

D'après le théorème des valeurs intermédiaires, il existe $\chi \in]\lfloor x \rfloor_{\varepsilon_i}, x[$ tel que $\frac{\partial F}{\partial i}(i,x) = \frac{\partial F}{\partial i}(i, \lfloor x \rfloor_{\varepsilon_i}) + \frac{\partial^2 F}{\partial x \partial i}(i,\chi)\,(x - \lfloor x \rfloor_{\varepsilon_i})$. Or,

$$\begin{aligned}
\left\|\frac{\partial^2 F}{\partial x \partial i}(i,\chi)\right\| &\leq \zeta\left(\left\|\frac{\partial G}{\partial x}(\lfloor i+1\rfloor,\chi)\right\| + \left\|\frac{\partial G}{\partial x}(\lfloor i\rfloor,\chi)\right\|\right) \\
&\leq \zeta\left(\frac{\omega(\chi/\varepsilon_{i+1})}{\varepsilon_{i+1}}\|\Delta(\lfloor i+1\rfloor,\lfloor\chi\rfloor_{\varepsilon_{i+1}})\| + \frac{\omega(\chi/\varepsilon_i)}{\varepsilon_i}\|\Delta(\lfloor i\rfloor,\lfloor\chi\rfloor_{\varepsilon_i})\|\right) \\
&\leq \frac{\zeta^2}{\varepsilon_i}\left(\|\Delta(\lfloor i+1\rfloor,\lfloor x\rfloor_{\varepsilon_{i+1}})\| + \varepsilon_i\|\Delta(\lfloor i\rfloor,\lfloor x\rfloor_{\varepsilon_i})\|\right) \\
&\leq \frac{25}{\varepsilon_i} \quad \begin{array}{l}(\|f(\lfloor i+1\rfloor,\lfloor x\rfloor_{\varepsilon_{i+1}}+\varepsilon_{i+1}) - f(\lfloor i+1\rfloor,\lfloor x\rfloor_{\varepsilon_{i+1}})\| \\ \quad + \|f(\lfloor i\rfloor,\lfloor x\rfloor_{\varepsilon_i}+\varepsilon_i) - f(\lfloor i\rfloor,\lfloor x\rfloor_{\varepsilon_i})\|)\end{array} \qquad (4.2)
\end{aligned}$$

Donc, en utilisant l'équation obtenue grâce aux théorème des valeurs intermédiaires,

$$\begin{aligned}
\left\|\frac{\partial F}{\partial i}(i,x)\right\| &\leq \left\|\frac{\partial F}{\partial i}(i,\lfloor x\rfloor_{\varepsilon_i})\right\| + \left\|\frac{\partial^2 F}{\partial x\partial i}(i,\chi)\right\|(x-\lfloor x\rfloor_{\varepsilon_i}) \\
&\leq \left\|\frac{\partial F}{\partial i}(i,\lfloor x\rfloor_{\varepsilon_i})\right\| + \varepsilon_i\left\|\frac{\partial^2 F}{\partial x\partial i}(i,\chi)\right\|
\end{aligned}$$

Ce qui en réinjectant les majorations des dérivées première et seconde de F obtenues dans les équations 4.1 et 4.2 donne la borne voulue. □

Le lemme suivant montre que si l'on se donne la dérivée calculable dans A d'une fonction, cette fonction va appartenir à la classe $\mathscr{A}$ augmentée de l'opérateur LIM_w. Il utilise le lemme 4.21 pour étant donnée une fonction obtenir sa primitive en tant que limite d'une fonction de la classe.

Lemme 4.22 *Soit A classe de fonctions discrètes. Soit $\mathscr{A}$ algèbre de fonctions réelles telle que $A \subseteq DP(\mathscr{A})$, et $\mathscr{A}$ stable par composition et prise de primitive et contenant $\mathscr{L}$.*

Étant donnée $f : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$ fonction de classe $\mathscr{C}^1$ appartenant à $A(\mathbb{R})$ définie sur un intervalle fermé à bornes rationnelles ou infinies et contenant 0, possédant un module de continuité dans A, Alors, la primitive F de f valant 0 en 0 appartient à $\mathscr{A}^\star$

Démonstration : Soit $f : \mathcal{D} \to \mathbb{R}$ de classe $\mathscr{C}^1$ appartenant à $A(\mathbb{R})$ dotée d'un module de continuité $M_{\mathbb{N}} : \mathbb{N}^2 \to \mathbb{N}$, $m \in A$.

Soient $i, j \in \mathbb{N}$, soit $x_j = j2^{-M_{\mathbb{N}}(i+1,i)}$. Pour tout $x, y \in [x_j, x_{j+1}] \cap [-i-1, i+1]$, on a $|f(x) - f(y)| \leq 2^{-i}$.

Les $f(x_j)$ vont nous permettre de représenter une approximation de la fonction f. Calculons des entiers p_j et q_j tels que $|p_j 2^{-q_j} - f(x_j)| \leq 2^{-i}$. Des fonctions $p_{\mathbb{N}} : \mathbb{N}^2 \to \mathbb{N}$ et $q_{\mathbb{N}} : \mathbb{N}^2 \to \mathbb{N}$ qui à (i,j) associent respectivement p_j et q_j appartiennent à A.

Par hypothèse, $A \subseteq DP(\mathscr{A})$, donc $p_{\mathbb{N}}$, $q_{\mathbb{N}}$ et $M_{\mathbb{N}}$ ont des extensions p, q et M dans $\mathscr{A}$. On peut alors définir une fonction $g : \mathbb{R} \times \mathcal{D} \to \mathbb{R}$ définie par

$$g(i,x) = p(i, 2^{-M(i+1,i)}x)2^{-q(i,2^{-M(i+1,i)}x)}.$$

Par construction, on a pour tout i, j entiers

$$g(i, x_j) = p_j 2^{-q_j}.$$

Définissons une fonction $\varepsilon : i \mapsto 2^{-M(i+1,i)}$. On applique le lemme 4.21 à la fonction g. On obtient alors une fonction F qui appartient à l'algèbre de fonctions $\mathscr{A}$ et vérifie

$$F(i, x_j) = g(i, x_j) = p_j 2^{-q_j}.$$

On a également pour tout $(i, j) \in \mathbb{N}^2$,

$$\|F(i, x_j) - f(x_j)\| \leq 2^{-i}.$$

Par notre choix de la fonction ε, on impose que les x_j soit de la forme $j\varepsilon(i)$, c'est-à-dire peuvent être écrits comme étant des $\lfloor x \rfloor_{\varepsilon(i)}$ pour certains x (pour tous les x appartenant à $[x_j, x_{j+1}[$ pour être précis). On peut constater que la fonction F approche bien f dès que i est suffisamment grand. Pour tout $x \in \mathcal{D}$, et tout $i \geq \|x\|$, on a $[x_j, x_{j+1}] \subseteq [-(i+1), i+1]$ et

$$\begin{aligned}
\|F(i,x) - f(x)\| &\leq \|F(i,x) - F(i,x_j)\| + \|F(i,x_j) - g(i,x_j)\| \\
&\quad + \|g(i,x_j) - f(x_j)\| + \|f(x_j) - f(x)\| \\
&\leq \|F(i,x_{j+1}) - F(i,x_j)\| + 0 + 2^{-i} + 2^{-i} \\
&\leq \|g(i,x_{j+1}) - g(i,x_j)\| + 2 \times 2^{-i} \\
&\leq \|g(i,x_{j+1}) - f(x_{j+1})\| + \|f(x_{j+1}) - f(x_j)\| \\
&\quad + \|f(x_j) - g(i,x_j)\| + 2 \times 2^{-i} \\
&\leq 5 \times 2^{-i}
\end{aligned}$$

Nous allons maintenant intégrer la fonction F pour parvenir au fait que la primitive de f valant 0 en 0 appartient bien à l'algèbre de fonctions $\mathscr{A}^\star$.

Considérons la fonction $G \in \mathscr{A}$ définie comme étant la solution du problème de Cauchy suivant :

$$\begin{cases} G(i,0) = 0 \\ \frac{\partial G}{\partial x}(i,x) = F(i,x) \end{cases}$$

En d'autres termes, $G(i,x) = \int_0^x F(i,u)\,\mathrm{d}u$.

On a $\left\| \frac{\partial G}{\partial x}(i,x) - f(x) \right\| = \|F(i,x) - f(x)\| \leq 2 \times 2^{-i}$. Et, si $i \geq \|x\|$, on a

$$\left\| G(i,x) - \int_0^x f(u)\,\mathrm{d}u \right\| \leq 5 \times 2^{-i}\|x\|.$$

Ce qui signifie qu'à x fixé, la limite quand i tend vers $+\infty$ de $G(i,x)$ est la primitive de f valant 0 en 0. Pour montrer que cette primitive appartient à $\mathscr{A}^\star$, il ne reste plus qu'à montrer que 'on peut appliquer le schéma LIM_w à la fonction G.

Pour ce faire, on doit exhiber des polynômes β et K tels que pour t suffisamment grand, $\left\|\frac{\partial G}{\partial i}(i,x)\right\| \leq K(x)2^{-t\beta(x)}$.

On sait que $G(i,x) = \int_0^x F(i,u)\,\mathrm{d}u$. F est par construction différentiable, donc $\frac{\partial G}{\partial i} = \int_0^x \frac{\partial F}{\partial i}(i,u)\,\mathrm{d}u$. Essayons de borner $\frac{\partial G}{\partial i}$:

$$\begin{aligned}\left\|\frac{\partial G}{\partial i}(i,x)\right\| &\leq \int_0^x \left\|\frac{\partial F}{\partial i}(i,u)\right\| \mathrm{d}u \\ &\leq \|x\| \times \max_{[0,x]} \left\|\frac{\partial F}{\partial i}(i,\chi)\right\| \\ &\leq (x^2+1) \times \max_{[0,x]} \left\|\frac{\partial F}{\partial i}(i,\chi)\right\|\end{aligned}$$

Le lemme 4.21 nous donnait une majoration de $\left\|\frac{\partial F}{\partial i}(i,\chi)\right\|$:

$$\begin{aligned}\left\|\frac{\partial F}{\partial i}(i,\chi)\right\| \leq & 5\|g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_i}) - g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i})\| \\ & + 25\|g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i} + \varepsilon_i) - g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i})\| \\ & + 25\|g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_{i+1}} + \varepsilon_{i+1}) - g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_{i+1}})\|.\end{aligned}$$

Chacun des termes peut être borné par un multiple de 2^{-i} de la façon suivante :

$$\|g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_i}) - g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i})\| \leq 2 \times 2^{-i}$$

$$\begin{aligned}\|g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i} + \varepsilon_i) - g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i})\| &\leq \|g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i} + \varepsilon_i) - f(\lfloor\chi\rfloor_{\varepsilon_i} + \varepsilon_i)\| \\ &\quad + \|f(\lfloor\chi\rfloor_{\varepsilon_i} + \varepsilon_i) - f(\lfloor\chi\rfloor_{\varepsilon_i})\| \\ &\quad + \|f(\lfloor\chi\rfloor_{\varepsilon_i}) - g(\lfloor i\rfloor, \lfloor\chi\rfloor_{\varepsilon_i})\| \\ &\leq 3 \times 2^{-i}\end{aligned}$$

$$\|g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_{i+1}} + \varepsilon_{i+1}) - g(\lfloor i+1\rfloor, \lfloor\chi\rfloor_{\varepsilon_{i+1}})\| \leq 3 \times 2^{-i}$$

Et donc, si $i \geq \|x\|$, on a

$$\left\|\frac{\partial G}{\partial i}(i,x)\right\| \leq 160(x^2+1)2^{-i}.$$

On peut donc appliquer le schéma LIM_w à la fonction G pour obtenir la fonction $\int(f)$ primitive de f valant 0 en 0. Et $\int(f) \in \mathscr{A}^\star$ □

La preuve de la proposition 4.18 est maintenant immédiate.

Proposition 4.18 (Rappel) *Soit A une classe de fonctions sur les entiers, close par composition et contenant $\mathscr{E}^\mathscr{E}(\mathbb{N})$.*

Soit $\mathscr{A}$ une classe de fonctions sur les réels close par composition, prise de primitive et contenant $\mathscr{L}$.

Si $A \subseteq DP(\mathscr{A})$, alors pour les fonctions de classe $\mathscr{C}^2$, définies sur un compact et dont les dérivées ont un module de continuité dans A,

$$A(\mathbb{R}) \subseteq \mathscr{A}^\star.$$

Démonstration : Soit f une fonction de $A(\mathbb{R})$ de classe $\mathscr{C}^2$, définie sur un compact. La fonction f' est de classe $\mathscr{C}^1$ et a un module de continuité dans A. On peut donc appliquer le lemme 4.22 à f'.

On obtient alors une fonction $F \in \mathscr{A}^\star$ définie sur le même domaine que f et dont la dérivée est f'. Les fonctions F et f ne diffèrent que d'une constante $f(0)$. Or $f(0)$ est calculable dans $A(\mathbb{R})$, donc appartient à $\mathscr{A}$. Et finalement, f appartient à $\mathscr{A}^\star$. □

Nous avons donc montré une caractérisation des fonctions récursivement calculables par la classe $\mathscr{L}^\star_{!_\mu}$. Ces deux ensembles de fonctions coïncident sur les fonctions de classe $\mathscr{C}^2$ définies sur des compacts.

4.3.2 Caractérisation de $\mathscr{E}(\mathbb{R})$ et de la hiérarchie de Grzegorczyk

L'opérateur de limite LIM_w permet également d'obtenir simplement une caractérisation des fonctions élémentairement calculables puis des caractérisations des classes de la hiérarchie de Grzegorczyk contenant $\mathscr{E}(\mathbb{R})$ (c'est-à-dire les classes $\mathscr{E}_n(\mathbb{R})$ pour $n \geq 3$).

Théorème 4.23 *Pour une fonction f de classe $\mathscr{C}^2$ définies sur un compact,*

$$f \in \mathscr{L}^\star \Leftrightarrow f \in \mathscr{E}(\mathbb{R}).$$

Théorème 4.24 *Pour une fonction f de classe $\mathscr{C}^2$ définies sur un compact, pour $n \geq 3$,*

$$f \in \mathscr{L}^\star_n \Leftrightarrow f \in \mathscr{E}_n(\mathbb{R}).$$

Nous allons montrer ces 2 théorèmes en réutilisant les propositions et lemmes utilisés pour le théorème 4.14 :

4.3.2.a Sens directs des preuves

Proposition 4.25

$$\mathscr{L}^\star \subseteq \mathscr{E}(\mathbb{R}).$$

Démonstration : Nous savons depuis la partie 2.2.2 que 0, 1, -1, U et θ_3 appartiennent à $\mathscr{E}(\mathbb{R})$. Nous savons également que COMP et LI préservent $\mathscr{E}(\mathbb{R})$. Il reste donc à montrer que LIM_w préserve $\mathscr{E}(\mathbb{R})$.

Soient f une fonction de $\mathscr{E}(\mathbb{R})$, K et β des polynômes tels que $g = \mathrm{LIM}_w(f, K, \beta)$ soit définie. Soit ϕ une fonctionnelle élémentaire calculant f.

Reprenons la preuve de la proposition 4.15.

La fonctionnelle ψ définie ci-après calcule la fonction g :

$$\psi : (x_n), j \mapsto \phi\left((\max(X, KN(j+1+\lceil \lg(KN)\rceil)), x_n), j+1\right)$$

Et cette fonctionnelle est élémentaire : max est élémentaire, K et N peuvent être choisis en n'utilisant que des calculs élémentaires. □

Proposition 4.26

$$\mathscr{L}_n^\star \subseteq \mathscr{E}_n(\mathbb{R}).$$

Démonstration : La preuve de la proposition 4.25 est aussi valable pour cette proposition. □

4.3.2.b Sens indirects des preuves

Rappelons la proposition 4.18 qui dit qu'étant données une algèbre de fonctions $\mathscr{A}$ contient $\mathscr{L}$ et est stable par composition et prise de primitive, une classe de fonctions discrètes contenant $\mathscr{E}(\mathbb{N})$ et stable par composition, si $A \subseteq DP(\mathscr{A})$, alors pour les fonctions $\mathscr{C}^2$ définies sur des compacts, $A(\mathbb{R}) \subseteq \mathscr{A}^\star$. On peut facilement appliquer cette proposition pour obtenir les propositions qui nous intéressent :

Proposition 4.27 *Soit f une fonction de classe $\mathscr{C}^2$ définie sur un produit d'intervalles compacts,*

$$f \in \mathscr{E}(\mathbb{R}) \Rightarrow f \in \mathscr{L}^\star.$$

Démonstration : Il suffit d'appliquer la proposition 4.18 avec $\mathscr{A} = \mathscr{L}$ en remarquant que $\mathscr{L}$ contient $\mathscr{L}$ et est stable par composition et prise de primitive, que $\mathscr{E}(\mathbb{N})$ est stable par composition et que $\mathscr{E}(\mathbb{N}) \subseteq DP(\mathscr{L})$ (on a vu qu'il y avait égalité). □

Proposition 4.28 *Soit f une fonction de classe $\mathscr{C}^2$ définie sur un produit d'intervalles compacts, pour $n \geq 3$,*

$$f \in \mathscr{E}_n(\mathbb{R}) \Rightarrow f \in \mathscr{L}_n^\star.$$

Démonstration : Pour $n \geq 3$, $\mathscr{E}_n(\mathbb{N}) \supseteq \mathscr{E}(\mathbb{N})$ et $\mathscr{L}_n \supseteq \mathscr{L}$. On peut donc réappliquer la proposition 4.18. □

4.4 Autres résultats

4.4.1 Généralisation aux fonctions plus lisses

Dans ce qui précède, nous avons considéré uniquement des fonctions de classe $\mathscr{C}^2$, en effet, supposer que les fonctions sont suffisamment lisses est nécessaires pour certaines preuves, par exemple la stabilité des fonctions de $\mathcal{R}ec(\mathbb{R})$ pour l'opérateur d'intégration linéaire fait appel aux dérivées secondes.

Nos résultats et preuves sont cependant toujours valables si l'on remplace θ_3 par θ_k et $\mathscr{C}^2$ par $\mathscr{C}^{k-1}$ pour $k \geq 3$.

Posons $k \geq 3$, et notons

$$\begin{aligned}
\mathscr{L}+!\mu_k &= [0, 1, U, \theta_k; \mathrm{COMP}, \mathrm{CLI}, \mathrm{UMU}] \\
\mathscr{L}^\star_{!\mu_k} &= [0, 1, U, \theta_k; \mathrm{COMP}, \mathrm{CLI}, \mathrm{UMU}, \mathrm{LIM}_w] \\
\mathscr{L}^\star_k &= [0, 1, -1, U, \theta_k; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w] \\
\mathscr{L}^\star_{n\,k} &= [0, 1, -1, U, \theta_k, \bar{E}_{n-1}; \mathrm{COMP}, \mathrm{LI}, \mathrm{LIM}_w]
\end{aligned}$$

En modifiant la définition de LIM_w pour qu'elle ne crée que des fonctions de classe $\mathscr{C}^{k-1}$, on constate que les classes de fonctions que l'on obtient ne contiennent que des fonctions $\mathscr{C}^{k-1}$.

Les théorèmes que l'on obtient s'adaptent sans difficultés à ces classes :

$$DP(\mathscr{L}+!\mu_k) = \mathcal{R}ec(\mathbb{N})$$

Pour f fonction de classe $\mathscr{C}^{k-1}$ définie sur un compact,

$$\begin{aligned}
f \in \mathscr{L}^\star_{!\mu_k} &\Leftrightarrow \mathcal{R}ec(\mathbb{R}) \\
f \in \mathscr{L}^\star_k &\Leftrightarrow f \in \mathscr{E}(\mathbb{R}) \\
\forall n \geq 3, f \in \mathscr{L}^\star_{n\,k} &\Leftrightarrow f \in \mathscr{E}_n(\mathbb{R}).
\end{aligned}$$

4.4.2 Classe de fonctions primitives récursives

La proposition 4.18 peut être appliquée à une classe caractérisant les fonctions primitives discrètes. Grâce à cela, on peut obtenir une classe contenant l'ensemble des fonction primitivement récursive calculables (de façon plus fine que $\mathscr{L}^\star_{!\mu}$).

Dans [Moo96] est présenté un opérateur $\bar{I}$ qui permet de définir une classe $\bar{\mathscr{D}}$. $\bar{I}$ est défini de la façon suivante :

Définition 4.29 ($\bar{I}$) *Étant données les fonctions $f_1, \ldots, f_m$ d'arité n et $g_1, \ldots, g_{m+1}$ d'arité $n+m+1$. Étant donné un intervalle F contenant 0. S'il existe un ensemble dénombrables $S \subseteq F$ de points isolés et un ensemble de fonctions $h_1, \ldots, h_m$ tels que*

$$\begin{aligned}
h_i(x, 0) &= f_i(x) \\
\frac{\partial h_i}{\partial y}(x, y) &= g_i(x, y, \vec{h}(x, y)) \text{ pour tout } y \in F - S
\end{aligned}$$

Alors, h_1 est définie comme étant $\bar{I}(f, g)$.

Définition 4.30 ($\bar{\mathscr{D}}$)

$$\bar{\mathscr{D}} = \left[0, 1, -1, U; \mathrm{COMP}, \bar{I}\right]$$

Cette classe contient des extensions des fonctions primitives récursives. En d'autres termes,

$$\mathcal{PR}(\mathbb{N}) \subseteq DP(\bar{\mathscr{D}}).$$

On constate facilement que $(\bar{\mathscr{D}} + \theta_3)^\star$ contient $\mathscr{L}$ et est stable par composition et prise de primitive. On peut donc appliquer la proposition 4.18 à cette classe. Les lemmes 4.19 et 4.20 sont aussi valables pour cette classe. Donc

$$\mathcal{PR}(\mathbb{R}) \subseteq (\bar{\mathscr{D}} + \theta_3)^\star$$

4.4.3 Nombres calculables

Les nombres réels calculables peuvent être définis comme étant les images de 0 par des fonctions récursivement calculables. On peut donc définir les nombres récursivement (respectivement élémentairement) calculables à l'aide des classes ci-avant définies.

Définition 4.31 (Nombres récursivement calculables) *On définit l'ensemble des nombres récursivement calculables comme étant*

$$\mathcal{T} = \{f(0) \in \mathbb{R}; f \in \mathscr{L}^\star_{!\mu}\}.$$

Définition 4.32 (Nombres élémentairement calculables) *L'ensemble*

$$\mathcal{S} = \{f(0) \in \mathbb{R}; f \in \mathscr{L}^\star\}$$

est l'ensemble des nombres élémentairement calculables.

Nous allons maintenant étudier quelques propriétés de ces ensembles, en particulier, le fait qu'ils contiennent les nombres rationnels, les nombres algébriques, que ces ensembles sont des corps strictement inclus dans $\mathbb{R}$ et qu'ils s'agit de corps réels clos (mais pas de corps algébriquement clos). Les résultats présentés ici ne sont pas nouveaux : les classes de nombres calculables (au sens de l'analyse récursive) ont déjà été étudiées par exemple dans [Mül86] et cette notion recouvre celle que nous venons de définir, mais les démonstrations ici faites utilisent les opérateurs que nous étudions et sont concises, ce qui les rend intéressantes.

Proposition 4.33 *Les nombres élémentairement calculables sont récursivement calculables.*

Démonstration : Cette inclusion est triviale : si $x \in \mathcal{S}$, il existe $f \in \mathscr{L}^\star$ telle que $x = f(0)$. Or $\mathscr{L}^\star \subsetneq \mathscr{L}^\star_{!\mu}$, donc $f \in \mathscr{L}^\star_{!\mu}$ et $x \in \mathcal{T}$. □

Proposition 4.34 *Les ensembles $\mathcal{S}$ et $\mathcal{T}$ sont dénombrables.*

Démonstration : Par construction de $\mathscr{L}^\star$ et $\mathscr{L}^\star_{!\mu}$, on peut indexer les fonctions de ces classes par les entiers. Notons ϕ_n la fonction de $\mathscr{L}^\star_{!\mu}$ d'indice n. Alors, la fonction $n \mapsto \phi_n(0)$ est un indexage des éléments de $\mathcal{T}$. □

Proposition 4.35 *Les nombres rationnels sont élémentairement calculables.*

Démonstration : Il suffit pour montrer ce résultat de reprendre la preuve de [CMC02, Lemme 4.6]. Soit p/q un nombre rationnel. La fonction $x \mapsto 1$ appartient à la classe $\mathscr{L}^\star$. $\mathscr{L}^\star$ est stable par intégration. Donc la primitive q-ième de $x \mapsto 1$ valant 0 en 0 appartient à $\mathscr{L}^\star$. Or cette primitive est la fonction $F : x \mapsto \frac{1}{q!}x^q$. Notons $inc : x \mapsto x+1$ et $g : x \mapsto p \times (q-1)!$. Ces fonctions appartiennent à $\mathscr{L}^\star$ et $(g \times F \circ inc)(0) = \frac{p}{q}$. □

Proposition 4.36

$$\mathcal{T} \subsetneq \mathbb{R}$$

Démonstration : L'inclusion dans $\mathbb{R}$ est immédiate. Un argument de dénombrement permet d'établir que l'inclusion est stricte.

On pourrait aussi remarquer que la constante Ω de Chaitin n'appartient pas à $\mathcal{T}$, exhibant ainsi un élément de $\mathbb{R} - \mathcal{T}$. □

Nous avons donc pour l'instant montré que

$$\mathbb{Q} \subsetneq \mathcal{S} \subseteq \mathcal{T} \subsetneq \mathbb{R}.$$

L'inclusion de $\mathbb{Q}$ dans $\mathcal{S}$ est stricte : $\pi \in \mathcal{S}$ mais n'est pas rationnel.

Proposition 4.37 *Les nombres algébriques sont récursivement calculables.*

Démonstration : Soit ϕ un nombre algébrique. Il existe un polynôme $P(X)$ à coefficients rationnels dont ϕ est racine. P appartient à $\mathscr{L}^\star_{!\mu}$. Supposons que ϕ soit une racine de P de degré 1. Si ce n 'est pas le cas, on peut dériver P $d-1$ fois (où d est le degré de ϕ) et obtenir un polynôme à coefficients rationnels dont ϕ est racine de degré 1. On a donc $P'(\phi) \neq 0$. Supposons $P'(\phi) > 0$. Si ce n'est pas le cas, il suffit de considérer $-P$. Il existe des rationnels p et q tels que $p < \phi < q$ et $\forall x \in [p,q]$, $P'(x) > 0$. On peut appliquer UMU à la fonction f définie sur $0 \times [p,q]$ qui à $0, x$ associe $P(x)$. Et on obtient $g : 0 \mapsto \phi$. Donc $\phi \in \mathcal{T}$. □

Pour montrer que les nombres algébriques ne sont pas que récursivement calculables mais aussi élémentairement calculables, nous allons utiliser un lemme montrant que la recherche d'un zero unique est élémentaire si il existe un minorant strictement positif de la dérivée.

Lemme 4.38 *Soit $f : \mathcal{D} \times \mathcal{I} \to \mathbb{R}$ telle que*

1. *$\forall x \in \mathcal{D}$, $\exists g(x) > 0$ vérifiant*
$$\forall y \in \mathcal{I}, \frac{\partial f}{\partial y}(x, y) > g(x),$$
2. *$\forall x \in \mathcal{D}$, $\exists y_0 \in \mathcal{I}$ vérifiant*
$$f(x, y_0) = 0.$$

Alors, $\mathrm{UMU}(f)$ *est définie.*
De plus, si $f \in \mathscr{E}(\mathbb{R})$ et $g \in \mathscr{E}(\mathbb{R})$, alors $\mathrm{UMU}(f) \in \mathscr{E}(\mathbb{R})$ *aussi.*

Démonstration : Pour montrer ce résultat, nous allons écrire un algorithme qui étant donné x et n donne une approximation de y_0 tel que $f(x, y_0) = 0$ à 2^{-n} près.

Présentons d'abord une routine `intervalle` qui étant donnés x, $\alpha = g(x)$ et optionnellement y_1 et y_2 tels que la racine appartienne à $[y_1, y_2]$, calcule un nouvel intervalle $[y_1, y_2]$ de taille moitié moindre de l'intervalle d'origine si celui-ci était donné.

Si y_1 et y_2 existent
    $y = \frac{1}{2}(y_2 - y_1)$
Sinon
    $y \in \mathcal{I}$
Soit $k > 1 - \lg(\alpha)$
Calculons $z = f(x, y)$ à 2^{-k} près
Si $z > 2^{-k}$
    Si y_1 n'existe pas
        $y_1 = y - \frac{z+2^{-k}}{\alpha}$
    $y_2 = y$
Sinon Si $z < -2^{-k}$
    $y_1 = y$
    Si y_2 n'existe pas
        $y_2 = y + \frac{-z+2^{-k}}{\alpha}$
Sinon
    $y_1 = y - 2\frac{2^{-k}}{\alpha}$
    $y_2 = y + 2\frac{2^{-k}}{\alpha}$
Renvoyer (y_1, y_2)

L'algorithme suivant calcule donc une approximation de la racine à 2^{-n} près :

Soit $y \in \mathcal{I}$
Soit $\alpha = g(x)$
$(y_1, y_2) =$ `intervalle`(x, α)
$b = n + \lg(y_2 - y_1)$
Pour i allant de 1 à b
    $(y_1, y_2) =$ `intervalle`(x, α, y_1, y_2)
Renvoyer y_1

Remarquons qu'à tout moment, y_1 et y_2 sont des rationnels tels que $y_1 \leq y \leq y_2$. Chaque appel à la routine `intervalle` divise la taille de l'intervalle de recherche par 2, donc après $n+\lg(y_2-y_1)$, l'erreur commise est inférieure à $(y_2-y_1)2^{-n-\lg(y_2-y_1)} \leq 2^{-n}$. □

Proposition 4.39 *Les nombres algébriques sont élémentairement calculables.*

Démonstration : Soit ϕ un nombre algébrique. Reprenons la preuve de la proposition 4.37. Nous avons un polynôme P sur un segment $[p,q]$ tel que pour tout $y \in [p,q]$, $P'(y) > 0$. P et P' sont des polynômes, ils appartiennent donc à $\mathcal{L}$. Il existe un rationnel $\alpha > 0$ qui minore P' sur $[p,q]$. On peut appliquer le lemme 4.38 pour obtenir la racine ϕ de façon élémentaire. Donc ϕ appartient à $\mathcal{S}$. □

Proposition 4.40 *Les ensembles $\mathcal{S}$ et $\mathcal{T}$ sont des corps.*

Démonstration : $\mathcal{S}$ et $\mathcal{T}$ sont stables par multiplication et par division par un nombre non nul (voir exemple 4.11). $\mathcal{S}$ et $\mathcal{T}$ sont donc des corps. □

Proposition 4.41 *$\mathcal{S}$ et $\mathcal{T}$ ne sont pas algébriquement clos.*

Démonstration : i et $-i$, les racines de X^2+1 n'appartiennent pas à $\mathcal{S}$ ou $\mathcal{T}$. □

Proposition 4.42 *$\mathcal{S}$ et $\mathcal{T}$ sont des corps réel clos*

Démonstration : Cette proposition découle naturellement du travail fait pour montrer que les nombres algébriques sont inclus dans $\mathcal{S}$ et $\mathcal{T}$.

Tout polynôme de degré impair, à coefficient dans $\mathcal{S}$ ou dans $\mathcal{T}$ est un polynôme de degré impair à coefficients réels. On a vu qu'il était possible de calculer les racines réelles d'un polynôme dont les coefficients sont rationnels. La preuve n'utilise pas la rationnalité des coefficients mais leur calculabilité. Donc on montre que les racines réelles d'un polynômes à coefficients dans $\mathcal{S}$ sont dans $\mathcal{S}$. Et de même pour $\mathcal{T}$. □

4.4.4 Universalité

Les fonctions récursives discrètes sont dénombrables, ce qui signifie que l'on peut les indexer par les entiers. On peut ainsi numéroter chacune des fonctions récursives. D'autre part, il est possible d'encoder 2 entiers en un seul sans perte d'information. Grâce à ces deux faits, on a pu définir dans la proposition 1.25 une fonction récursive universelle discrète.

Dans le cas réel, on peut songer pour trouver une fonction qui représenterait deux réels en un seul de façon bijective à utiliser la courbe de Hilbert.

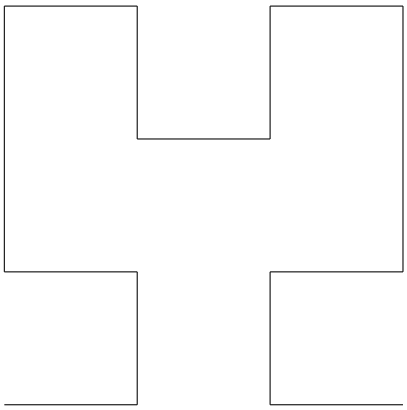

FIG. 4.3: Courbe de Hilbert de rang 2

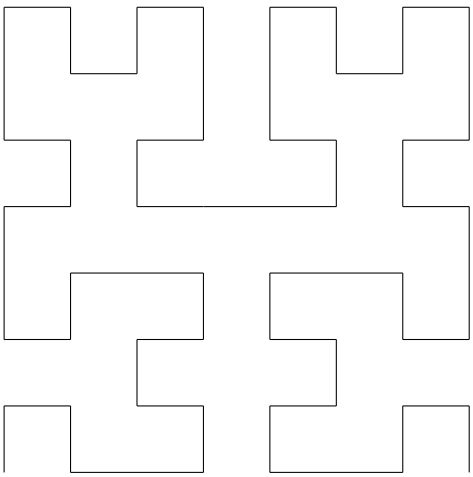

FIG. 4.4: Courbe de Hilbert de rang 3

La courbe de Hilbert (représentée dans les figures 4.3 et 4.4 pour les rangs 2 et 3) est une fonction de $[0,1]$ vers $[0,1]^2$. Sa limite est surjective (tout point du carré $[0,1]^2$ est atteint). La courbe de Hilbert semble donner un indice d'une façon de définir une bijection entre $\mathbb{R}^2$ et $\mathbb{R}$. Cependant, il n'existe pas de telle bijection...

Proposition 4.43 ([LFA74]) *Il n'existe pas de $\mathscr{C}^1$ difféomorphisme de $[0,1]^n$ sur $[0,1]^p$ si $n \neq p$.*

Démonstration : Nous allons donner deux preuves de ce résultat : une preuve pour $n = 1$ et $p = 2$ et une preuve générale :

- Soit $f : [0,1] \to [0,1]^2$ un $\mathscr{C}^1$-difféomorphisme. Soit ϕ sa réciproque.
  Soit X un sous-ensemble ouvert non vide de $[0,1]^2$ tel que $[0,1]^2 - X$ soit connexe. On peut par exemple choisir X comme la boule ouverte de centre $(\frac{1}{2}, \frac{1}{2})$ et de rayon $\frac{1}{4}$. Comme ϕ est continue, $\phi(X)$ est ouvert et $\phi([0,1]^2 - X)$ est connexe. Comme ϕ est bijective, $[0,1] - \phi(X) = \phi([0,1]^2 - X)$. Or $[0,1] - \phi(X)$ avec $\phi(X)$ ouvert non vide inclus dans $[0,1]$ ne peut être connexe : 0 et 1 appartiennent à $[0,1] - \phi(X)$ puisque $\phi(X)$ est ouvert, mais il existe $x \in [0,1]$ appartenant à $\phi(X)$ qui est non vide, empêchant $[0,1] - \phi(X)$ d'être connexe.
  Il y a donc contradiction.
- Considérons $f : [0,1]^p \to [0,1]^n$ et $\phi : [0,1]^n \to [0,1]^p$ deux fonctions de classe $\mathscr{C}^1$ telles que $f \circ \phi = Id_{[0,1]^n}$.
  Différencions l'expression $f \circ \phi = Id$. On notera $D(g)$ la différentielle de g. Rappelons que $D(g)$ est la fonction $\mathcal{D} \to \mathbb{R} \to \mathbb{R}$ qui à x associe $a \mapsto f'(x)a$

  $$\forall x \in [0,1]^n, D(f)(\phi(x)) \circ D(\phi)(x) = Id_{\mathbb{R}^n}$$

  Cela signifie que $D(f)(\phi(x))$ et $D(\phi)(x)$ sont réciproques. Par définition des différentielles, $D(f)(y)$ et $D(\phi)(x)$ sont des applications linéaires sur des espaces vectoriels de dimension finie. Le théorème du rang montre que tout morphisme bijectif sur des espaces de dimension finie associe deux espaces de même dimension. En d'autres termes, $n = p$

□

On ne va donc pas pouvoir définir de fonction universelle pour l'ensemble des fonctions récursives discrètes. On peut cependant construire un ensemble de fonctions dont chacune sera l'universelle d'une classe de fonction d'arité donnée.

En se plaçant dans le contexte de l'analyse récursive, et plus précisément des machines de type 2, il est assez facile d'imaginer des machines universelles. On peut pour illustrer ce point se référer à [Wei00, section 2.3]. Pour simplifier, on peut en déduire qu'il existe une machine de type 2, qui pour deux entrées n et x renvoie le résultat de la n-ième machine de type 2 sur l'entrée x. Dans la mesure où la description d'une machine de type 2 est analogue à la description d'une machine de Turing, l'existence de machines de Turing universelles suffit à montrer qu'il existe des machines de type 2 universelles.

Pour montrer l'existence d'une fonction universelle, il faut d'abord s'accorder sur un indexage des fonctions de $\mathscr{L}^\star_{!\mu}$. On prendra par exemple l'indexage standard tel qu'il est

proposé dans la preuve de la proposition 1.21. On notera f_n la n-ième fonction de $\mathscr{L}^\star_{!\mu}$ vis à vis de cet encodage.

Proposition 4.44 (Fonction universelle) *Il existe une fonction $u : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ vérifiant*

$$\forall n \in \mathbb{N}, \forall x \in \mathbb{R}, u(n, x) = f_n(x).$$

Démonstration : On a donc une machine $\mathcal{M}_u$ qui sur une entrée (n, x) travaille comme $\mathcal{M}_n$ sur l'entrée x. $\mathcal{M}_u$ est une machine de type 2, il existe donc une fonction de $\mathscr{L}^\star_{!\mu}$ que l'on appellera $f_m : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ qui à une entrée (n, x) associe le résultat du calcul de la n-ième machine de type 2 sur l'entrée x. Pour avoir une fonction universelle sur $\mathscr{L}^\star_{!\mu}$, il faut alors avoir une correspondance entre l'indexage sur $\mathscr{L}^\star_{!\mu}$ et celui sur les machines de type 2. Supposons que l'on ait un tel encodage ϕ, c'est-à-dire une fonction récursive telle que la fonction f_n soit calculée par la machine $\mathcal{M}_{\phi(n)}$. Alors, la fonction $f_u : n, x \mapsto f_m(\phi(n), x)$ est une fonction universelle de $\mathscr{L}^\star_{!\mu}$ puisque $\forall n, x, f_u(n, x) = f_m(\phi(n), x) = \mathcal{M}_u(\phi(n), x) = \mathcal{M}_{\phi(n)}(x) = f_n(x)$.

Or, nous avons montrer comment il était possible étant donnée une fonction de $\mathscr{L}^\star_{!\mu}$ de créer une fonctionnelle, c'est-à-dire une machine de type 2, qui la calcule. Cette construction peut se traduire en un encodage ϕ qui vérifie nos conditions. □

Ce résultat nous donne donc une machine universelle pour les fonctions d'arité 1. Cette machine calcule une fonction calculable d'arité 2. Notons la $u_1 : \mathbb{R}^2 \to \mathbb{R}$. Elle vérifie $\forall (n, t) \in \mathbb{N} \times \mathbb{R}$, $u_1(n, t) = f_n(t)$ où f_n est la n-ième fonction récursivement calculable d'arité 1 pour une numérotation fixée.

De la même façon, on définit des fonctions universelles pour les arités supérieures $u_i : \mathbb{R}^{i+1} \to \mathbb{R}$ telles que $\forall (n, \vec{x}) \in \mathbb{N} \times \mathbb{R}^i$, $u_i(n, \vec{x}) = f^i_n(\vec{x})$ où f^i_n est la n-ième fonction récursivement calculable d'arité i pour une numérotation fixée.

Remarquons que pour tout i, la fonction u_i est une fonction calculable d'arité $i + 1$, il existe donc un entier ι tel que $u_i = \vec{x} \mapsto u_{i+1}(\iota, \vec{x})$.

Nous pouvons écrire un théorème de point fixe similaire au théorème de Kleene pour les fonctions récursives discrètes.

Théorème 4.45 *Pour toute fonction $g \in \mathscr{L}^\star_{!\mu}$, il existe un réel n tel que $f_n = f_{g(n)}$*

Démonstration : La preuve est presque identique au cas discret.

On cherche n tel que

$$\forall y, u_1(n, y) = u_1(g(n), y).$$

Posons $\psi : x \mapsto g(u_1(x, x))$. Cette fonction appartient à $\mathscr{L}^\star_{!\mu}$. Il existe donc un entier γ tel que $\psi = f_\gamma$. Cela signifie que

$$\forall x, g(u_1(x, x)) = u_1(\gamma, x).$$

On peut en déduire que

$$\forall x, y, u_1(g(u_1(x, x)), y) = u_1(u_1(\gamma, x), y).$$

En particulier, pour $x = \gamma$, on a

$$\forall y, u_1(g(u_1(\gamma, \gamma)), y) = u_1(u_1(\gamma, \gamma), y).$$

Donc en posant $n = u_1(\gamma, \gamma)$, on a bien

$$\forall y, u_1(n, y) = u_1(g(n), y).$$

Ce qui termine la preuve. □

Il pourrait être intéressant de voir si les autres théorèmes de la théorie des fonctions récursives se traduisent dans cette classe $\mathscr{L}^{\star}_{!\mu}$. En particulier, si l'on voulait montrer que $\mathscr{L}^{\star}_{!\mu}$ est une système acceptable de programmation, nous devrions montrer trois choses :

1. $\mathscr{L}^{\star}_{!\mu}$ contient les fonctions récursives ;
2. $\mathscr{L}^{\star}_{!\mu}$ possède une fonction universelle ;
3. Il existe un théorème s-n-m pour les fonctions de $\mathscr{L}^{\star}_{!\mu}$.

Les deux premières propriétés ont été prouvées, malgré la réserve de l'absence d'une fonction universelle canonique mais d'une famille de fonctions universelles définies par arité. Le cas du théorème s-n-m est épineux : la preuve classique qui illustre l'analogie entre ce théorème et la notion d'évaluation partielle d'un programme ne peut être appliquée à un ensemble non-dénombrable. En fait, l'existence d'un tel théorème semble contredire la proposition 4.43 en exhibant une fonction permettant d'encoder deux réels en un seul.

Troisième partie

Comparaison entre General Purpose Analog Computer et analyse récursive

5 Comparaison entre GPAC et analyse récursive

> Il grandissimo libro dell'universo è scritto in lingua matematica, e i caratteri son triangoli, cerchi, ed altre figure geometriche, senza i quali mezzi è impossibile a intenderne umanamente parola; senza questi è un aggirarsi vanamente per un oscuro laberinto.
>
> *(Galileo Galilei)*

Nous avons vu dans la section 2.3 que le GPAC est moins puissant que les machines de type 2 puisque ces dernières calculent des fonctions qu'un GPAC ne peut générer, en particulier la fonction Γ d'Euler. Cependant, les modes de fonctionnement du GPAC et des machines de type 2 sont fondamentalement différents : les machines de type 2 donnent un résultat approché qui s'affine pour tendre asymptotiquement vers le résultat escompté tandis que le GPAC donne un résultat en temps réel, la valeur de $f(t)$ est atteinte au temps t. Il paraît donc naturel d'étudier ce que peut calculer le GPAC s'il lui est donné la possibilité d'approcher un résultat, de calculer avec un processus de limite.

Nous allons étendre les résultats de [Gra04] qui montrent qu'en utilisant une définition de GPAC-calculabilité faisant apparaître une limite permet de capturer la fonction Γ d'Euler. Nous allons ici montrer une équivalence entre les fonctions récursivement calculables (au sens de l'analyse récursive) et les fonctions GPAC-calculables (dans le sens générées par un GPAC avec limite).

L'objet de ce chapitre peut être résumé par le méta-théorème suivant :

Méta-théorème 4 *Une fonction est* GPAC*-calculable si et seulement si elle est récursivement calculable.*

Le contenu de ce chapitre reprend les résultats présentés dans [BCGH06], résultats obtenus par un travail en collaboration avec Olivier Bournez, Manuel Campagnolo et Daniel Graça.

5.1 Définitions

Nous allons définir la notion de fonction GPAC-calculable comme fonction pouvant être générée par un GPAC avec un processus de limite.

Définition 5.1 (GPAC-calculable) *On dit d'une fonction $f : [a, b] \to \mathbb{R}$ qu'elle est* GPAC*-calculable s'il existe un polynôme $p : \mathbb{R}^{n+1} \to \mathbb{R}^n$ à coefficients calculables et $n-1$ constantes calculables $\alpha_1, \ldots, \alpha_{n-1}$ tels que : le problème de Cauchy*

$$\begin{cases} y' = p(y, t) \\ y(0) = (\alpha_1, \ldots, \alpha_{n-1}, x) \end{cases}$$

possède une solution $y_1, \ldots, y_n$ définie sur $[0, +\infty[$ telle qu'il existe $i, j \in \{1, \ldots, n\}$ pour lesquels

$$\lim_{t \to +\infty} y_j(t) = 0$$
$$\forall t, |f(x) - y_i(t)| \leq y_j(t).$$

En d'autres termes, f est calculable si il existe des fonctions g et ε composantes de la solution d'une équation différentielle polynomiale vérifiant

$$\forall x, \forall t \in [0, +\infty[, |f(x) - g(x, t)| \leq \varepsilon(x, t)$$
$$\forall x, \lim_{t \to +\infty} \varepsilon(x, t) = 0$$

Une fonction est θ_j-GPAC-calculable si elle est calculable par un GPAC augmenté de la fonction θ_j.

Définition 5.2 (θ_j-GPAC-calculable) *Une fonction f est θ_j-*GPAC*-calculable s'il existe g et ε composantes de la solution d'un système d'équations différentielles de la forme*

$$\begin{cases} y' = p(t, y, \theta_j(y)) \\ y(0) = (\alpha_1, \ldots, \alpha_{n-1}, x). \end{cases}$$

5.2 Propriétés et résultat

L'ajout du processus de limite permet de capturer la fonction Γ d'Euler.

Proposition 5.3 ([Gra04]) *La fonction Γ est* GPAC*-calculable.*

Démonstration : Nous ne ferons ici qu'une esquisse de la preuve. Pour une preuve complète, on peut se référer à [Gra04].

Rappelons que $\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} \, \mathrm{d}t$. La fonction $f(x, u) = \int_0^u t^{x-1} e^{-t} \, \mathrm{d}t$ est engendrable par un GPAC sans limite. L'erreur faite tend vers 0 quand u tend vers $+\infty$ et peut être majorée par une fonction engendrable par un GPAC sans limite qui tend elle aussi vers 0. Par exemple la fonction $(x, u) \mapsto 1/u$ dont nous avons prouvé dans la proposition 2.23 qu'elle bornait effectivement l'erreur commise par cette approximation.

La fonction Γ est donc GPAC-calculable dans le sens que nous venons de définir. □

Théorème 5.4 *Une fonction définie sur un fermé est* GPAC*-calculable si et seulement si elle appartient à $\mathcal{R}ec(\mathbb{R})$*

Nous allons prouver ce résultat en deux temps : un pour chaque implication.

5.3 GPAC-calculable implique récursivement calculable

Nous allons tout d'abord montrer le sens direct du théorème 5.4 Pour ce faire, nous allons utiliser un résultat issu de [GZB06] qui montre la calculabilité par les machines de type 2 de fonctions pCp.

Proposition 5.5 *Soit $E \subseteq \mathbb{R}^{m+1}$ un ouvert récursivement énumérable. Soit $f : E \to \mathbb{R}^m$ une fonction analytique récursivement calculable. Soit $]\alpha, \beta[$ l'intervalle sur lequel est définie une solution maximale $x(t)$ du problème de cauchy :*

$$\begin{cases} x' & = f(t, x) \\ x(t_0) & = x_0 \end{cases}$$

avec (t_0, x_0) un point récursivement calculable de E. Alors, $]\alpha, \beta[$ est un ouvert récursivement énumérable et x est une fonction récursivement calculable sur $]\alpha, \beta[$.

Dans le système de Cauchy qui nous intéresse, la fonction f est un polynôme, donc il s'agit bien d'une fonction analytique récursivement calculable. L'ensemble E est choisi pour être récursivement calculable.

Proposition 5.6 *Soit $f : [a, b] \to \mathbb{R}$ une fonction GPAC-calculable. Alors, f est récursivement calculable.*

Démonstration : f étant GPAC-calculable, il existe un système d'équations différentielles

$$\begin{cases} y' & = & p(t, y) \\ y(0) & = & (\alpha, x) \end{cases}$$

dont la solution a deux composantes y_i et y_j vérifiant

$$\forall x, t, |f(x) - y_i(x, t)| \leq y_j(x, t)$$

$$\lim_{t \to +\infty} y_j(x, t) = 0.$$

L'ensemble E défini pour la proposition 5.5 va être l'ensemble $[a, b] \times]0, +\infty[$. Cet ensemble est récursivement énumérable. p étant un polynôme, il est récursivement calculable et analytique. La solution maximale du système est définie sur $]0, +\infty[$ par définition des fonctions GPAC-calculables. Donc y_i et y_j sont définies sur $]0, +\infty[$.

Pour calculer $f(x)$ à la précision 2^{-n} près, on peut appliquer l'algorithme suivant :

$t = 1$
$\bar{y_j} = 1$
Tant que $\bar{y_j} > 2^{-n-2}$
  Calculer $\bar{y_j} \approx y_j(x, t)$ à 2^{-n-2} près
  $t = t + 1$
Calculer $z \approx y_i(x, t)$ à 2^{-n-1} près
Renvoyer z

La boucle permet d'obtenir un t pour lequel l'erreur commise en approchant $f(x)$ par $y_i(x,t)$ est inférieure à 2^{-n-1}. La valeur z renvoyée est alors une bonne valeur approchée de $f(x)$ à 2^{-n} près.

Les calculs de $\bar{y_j}$ et z sont récursivement calculables d'après la proposition 5.5. Donc cet algorithme décrit une fonction récursivement calculable. □

Nous pouvons maintenant passer au sens indirect du théorème 5.4.

5.4 Les fonctions récursivement calculables sont θ_j-GPAC-calculables

Dans un premier temps, nous allons montrer le sens indirect du théorème 5.4 en nous contentant de montrer qu'il est possible pour une fonction récursivement calculable de fabriquer un θ_j-GPAC qui calcule cette fonction. Nous montrerons dans la section suivante qu'il est possible de s'affranchir des θ_j en contrôlant finement l'erreur commise à chaque stade de la construction.

Nous allons tout d'abord présenter des résultats qui nous seront utiles pour prouver que toute fonction récursivement calculable est θ_j-GPAC-calculable.

5.4.1 Prérequis

Il est possible à l'aide d'un système d'équations différentielles de simuler l'itération d'une fonction. La construction est dûe à Michael Branicky dans [Bra95].

Proposition 5.7 ([Bra95]) *Soit $f : \mathbb{N} \to \mathbb{N}$, soit $j \geq 2$ un entier. Soit $\tilde{f}$ une extension réelle de f. Il existe un système d'équations différentielles polynomiales dont une des composantes y_1 de la solution vérifie*

$$\begin{aligned} \forall t, &\quad y_1(t,1) = f(t) \\ \forall t, n \in \mathbb{N}, &\quad y_1(t,n+1) = f(y_1(t,n)) \end{aligned}$$

Démonstration : Pour réaliser l'itération d'une fonction dont l'extension peut ne pas être régulière, nous allons utiliser notre fonction *int* qui approche la partie entière. Quand la fonction *int* représente effectivement l'entier partie entière de n, nous allons obliger la fonction y_1 à croître de sa valeur en $\lfloor n \rfloor$ à $f(y_1(t, \lfloor n \rfloor))$. Pour ce faire, nous allons utiliser deux fonctions : y_1 qui simule la croissance et y_2 qui retient la valeur précédente puis se met à jour, comme le suggère la figure 5.1.

Nous disposons de la fonction θ_j. Donc la fonction *int* définie dans la preuve du lemme 3.16 est engendrable par un θ_j-GPAC. Nous allons nous en servir pour itérer sur la valeur entière et non sur une valeur réelle qui pourrait être plus ou moins proche et créer des

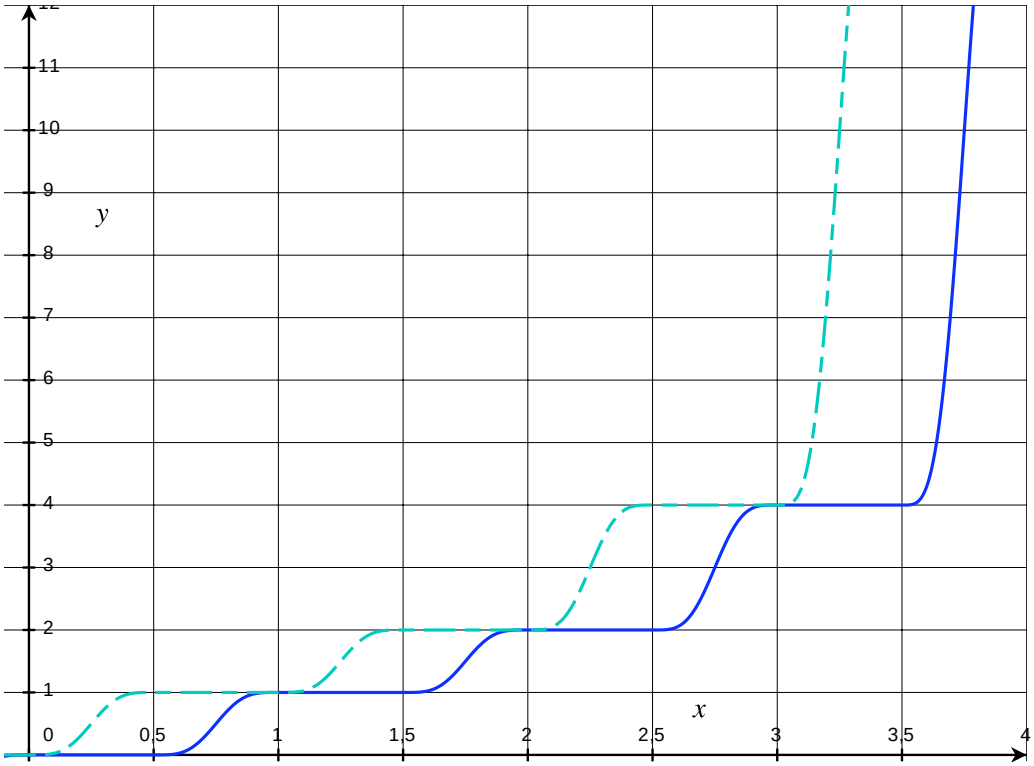

FIG. 5.1: Les fonction y_1 et y_2 simulent des itérations de la fonction $x \mapsto 2^x$

erreurs qui s'amplifieraient. Considérons le système d'équations différentielles suivant :

$$\begin{aligned}\frac{\partial y_1}{\partial t} &= \lambda_j(f(int(y_2)) - y_1)^3\theta_j(\sin(2\pi t))\\ \frac{\partial y_2}{\partial t} &= \lambda_j(int(y_1) - y_2)^3\theta_j(-\sin(2\pi t))\\ y_1(x,0) &= x\\ y_2(x,0) &= x\end{aligned}$$

Où λ_j est une constante choisie en fonction de j.

Sur les intervalles de la forme $[k, k+1/2]$ avec $k \in \mathbb{N}$, $\frac{\partial y_2}{\partial t} = 0$, donc y_2 reste fixé à une valeur n. Donc sur ces intervalles $\frac{\partial y_1}{\partial t} = \lambda_j(f(int(n)) - y_1)^3 \sin(2\pi t)^j$. Et avec un λ_j suffisamment grand (et indépendant de $f(int(n)) - y_1$), $y_1(k+1/2) = f(int(n))$. Sur les intervalles $[k+1/2, k+1]$, $\frac{\partial y_1}{\partial t} = 0$ donc seule y_2 change. De la même façon que y_1 l'a fait sur l'intervalle d'une demi-unité précédent, y_2 va atteindre $y_1(k+1/2)$. Et donc en $k+1$, $y_1(k+1) = y_2(k+1) = f(int(n)) = f(y_1(k))$.

Pour tout $t \in \mathbb{N}$, le fonctionnement est exact. Pour $t \notin \mathbb{N}$, les fonctions y_1 et y_2 sont définies, mais les mises à jour utilisant notre fonction de partie entière approchée, elles n'atteignent pas nécessairement des entiers. Ce n'est cependant pas un problème, notre seule préoccupation était que les fonctions y_1 et y_2 soient définies et qu'en les entiers, elles simulent l'itération. □

La construction précédente permet donc de simuler l'itération d'une fonction discrète, mais on peut aussi l'utiliser simplement pour rendre prévisible une fonction dont on ne connaît les valeurs qu'en les entiers. En effet, les fonctions obtenues par ce procédé ont la particularité d'être « canoniques »dans le sens où $y_1([n, n+1]) = [f(n), f(n+1)]$, c'est à dire de ne pas prendre des valeurs sortant de l'intervalle défini par les valeurs entières.

Nous allons maintenant donner un résultat concernant la simulation de machines de Turing par un GPAC. Et plus précisément, la simulation robuste d'une machine de Turing. Par robuste, on entend tolérante aux erreurs : si du bruit apparaît dans des limites raisonnables et distrait le GPAC, le résultat doit continuer d'être le bon.

Pour simuler une machine de Turing M, on code la configuration d'une machine de Turing par un triplet d'entiers naturels (q, r_-, r_+) représentant respectivement l'état, la partie du ruban à gauche de la tête de lecture et la partie du ruban à droite de la tête de lecture, celle-ci comprise.

On dit que la fonction $f_M : \mathbb{R}^4 \to \mathbb{R}^3$ simule M si $\forall t, q, r_-, r_+ \in \mathbb{N}$, $f(t, q, r_-, r_+)$ code la configuration de M sur l'entrée (q, r_-, r_+) au bout du temps t. La simulation est robuste si pour tout triplet $(q, r_-, r_+) \in \mathbb{R}^3$ tel qu'il existe $(\gamma, \rho_-, \rho_+) \in \mathbb{N}^3$ pour lequel $\|(q, r_-, r_+) - (\gamma, \rho_-, \rho_+)\|_\infty < \frac{1}{4}$, on a $\|f(t, q, r_-, r_+) - f(t, \gamma, \rho_-, \rho_+)\|_\infty < \frac{1}{4}$. En d'autres termes, étant donnée une marge d'erreur de $\frac{1}{4}$, la simulation ne dérive pas de plus de cette marge du résultat entier. En fait, il est possible de prendre n'importe quelle valeur d'erreur dans $]0, \frac{1}{2}[$. De façon évidente, toute valeur d'erreur autorisée supérieure à $\frac{1}{2}$ ne peut être acceptée puisque cela ne permettrait plus de discriminer 2 entiers.

Proposition 5.8 ([GCB05]) *Étant donnée une machine de Turing M, il existe une fonction $f_M : \mathbb{R}^4 \to \mathbb{R}^3$ solution d'une équation différentielle polynomiale qui simule M de façon robuste.*

5.4.2 Principe de la preuve

L'inclusion des fonctions récursivement calculables dans l'ensemble des fonctions GPAC-calculables constitue la proposition 5.12. L'idée de la preuve de cette proposition va consister, étant donnée une fonction f récursivement calculable à faire une boucle partant de $n = 0$ et incrémentant n à chaque pas de façon à améliorer la précision 2^{-n}. Dans le corps de la boucle, on commence par discrétiser l'entrée réelle, on simule le fonctionnement de la machine de Turing qui calcule la fonctionnelle représentant la fonction f, on stocke ensuite le résultat converti en rationnel dans une mémoire qui le conservera jusqu'à la fin du calcul du pas de la boucle suivant. Le fonctionnement est schématisé par la figure 5.2. Les flèches continues peuvent être considérées comme des fils de liaison entre les différents éléments du GPAC. La flèche pointillée représente une opération qui n'a lieu que dans le cas où le moment est venu d'incrémenter la précision. Les flèches discontinues signalent que la valeur qui transite est un entier.

Le lemme 5.9 montrera la réalisation d'un composant permettant d'effectuer la mémorisation. Le corps de la boucle, c'est-à-dire les étapes de calcul proprement dit : discrétisation, simulation de machine de Turing et calcul d'un rationnel approchant le résultat seront présentées dans le lemme 5.10. Le lemme 5.11 montrera comment réaliser une horloge annonçant si le calcul est terminé, c'est-à-dire si l'incrémentation de n doit être réalisée. Enfin, la proposition 5.12 regroupera les résultats pour terminer la preuve de l'inclusion des fonctions récursivement calculables dans l'ensemble des fonctions GPAC-calculables.

5.4.3 Résultats intermédiaires

Le lemme suivant montre qu'il est possible de mémoriser un résultat y et de le changer quand un signal c le demande. Une représentation schématique de ce composant est donnée en figure 5.3. Il s'agit d'un composant très simple, mais pour le réaliser, nous avons besoin d'un certain nombre d'hypothèses comme la constance de la fonction à mémoriser quand le signal c ordonne la mise en mémoire. Nous ne pouvons de plus mémoriser qu'une valeur entière car le processus peut être sujet à un certain bruit. L'énoncé du lemme suivant est donc quelque peu compliqué.

Les conditions que nous posons sont pour un certain $\varepsilon < 1/2$ d'une part que si le signal dépasse ε, il va atteindre $1 - \varepsilon$ sans repasser sous la barre ε et inversement s'il descend en dessous de $1 - \varepsilon$, il va atteindre ε sans repasser au dessus de $1 - \varepsilon$ et d'autre part que quand le signal dépasse ε, la valeur à stocker se fige sur une constante proche d'un entier. On obtient alors une sortie g qui après un temps d'adaptation de 1 va représenter la constante voulue (à savoir $\lfloor f(t) + 1/4 \rfloor$) et se fige sur cette valeur tant que le signal ne redépasse pas ε.

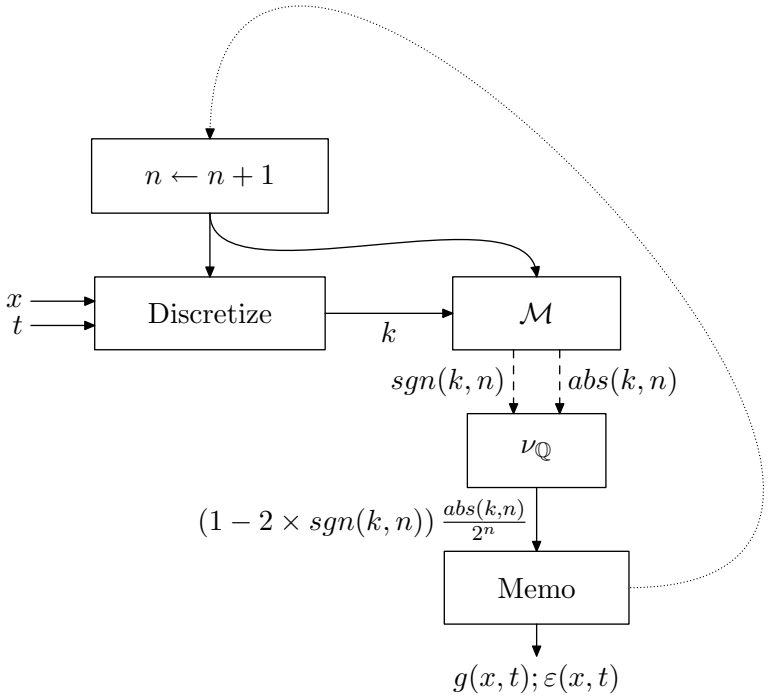

FIG. 5.2: Schéma du fonctionnement d'un GPAC calculant une fonction de $\mathcal{R}ec(\mathbb{R})$

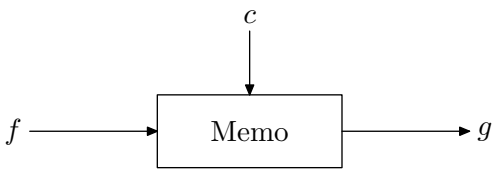

FIG. 5.3: Composant de mémorisation

Lemme 5.9 (Mémorisation) *Il existe un θ_j-GPAC qui pour des entrées $f : \mathbb{R} \to \mathbb{R}$ et $c : \mathbb{R} \to [0, 1]$ vérifiant pour un certain ε*

1. *si $\varepsilon < c(t) < 1-\varepsilon$, alors il existe u et v tels que $u < t < v$ et $c(u) = \varepsilon$, $c(v) = 1-\varepsilon$ et $\forall w \in]u, v[$, $c(w) \in]\varepsilon, 1-\varepsilon[$ (ou inversement, $c(v) = \varepsilon$ et $c(u) = 1-\varepsilon$),*
2. *sur tout intervalle tel que $c(t) > \varepsilon$, $f(t)$ est constante et de valeur dans un intervalle $[k - 1/4, k + 1/4]$ avec $k \in \mathbb{N}$*

donne une sortie g telle que

1. *si $c(t) = \varepsilon$ et il existe $\eta > 0$ et u tel que $\forall x \in]t-\eta, t[, c(x) < \varepsilon$, alors $g(t+1) = \lfloor f(t) + 1/4 \rfloor$,*
2. *si $c(t) = \varepsilon$ et il existe $\eta > 0$ et $u > t$ tels que $\forall x \in]t-\varepsilon, t[, c(x) > \varepsilon$, $c(u) = \varepsilon$ et $\forall v \in]t, u[, c(v) < \varepsilon$, alors $\forall v \in]t+1, u[$, $g(v) = g(t+1)$.*

Démonstration : Le principe est ici d'exhiber une équation différentielle pour g qui forcera g à être constante quand $c < 1 - \varepsilon$ et si c s'approche de 1, g fera un bond pour passer de son ancienne valeur à la valeur courante de f.

On suppose que sur les intervalles où $c(t) > 1 - \varepsilon$, $f(t)$ est constante de valeur mem.

Posons le système d'équations suivant :

$$y_1' = \lambda(int(mem) - y_1)^3\theta_j(\sin(2\pi t))\sigma_\varepsilon(c) + \lambda(y_2 - y_1)^3\theta_j(\sin(2\pi t))\sigma_{1-\varepsilon}(1-c) \quad (5.1)$$

$$y_2' = \lambda(y_1 - y_2)^3\theta_j(\sin(-2\pi t)) \quad (5.2)$$

Où $\sigma_\varepsilon(c)$ est une idéalisation de c telle que

$$\sigma_\varepsilon(x) = 0 \text{ si } x < 1 - \varepsilon$$
$$\sigma_\varepsilon(x) = \theta_j(\sin(2\pi(\mu t - \mu\nu))) \text{ si } x \geq 1 - \varepsilon$$
$$\text{où l'intervalle sur lequel } x \geq 1 - \varepsilon \text{ est de la forme } [\nu, \nu + \mu]$$

Les fonctions y_1 et y_2 permettent par un comportement inspiré de la construction de Branicky présentée en proposition 5.7 de réaliser une fonction g qui vérifie les conditions imposées. Les deux termes qui apparaissent dans l'équation pour y_1' sont respectivement la mise à jour quand le signal c est présent, c'est-à-dire quand $\sigma_\varepsilon(c) \approx 1$, et les ajustements pour corriger des éventuelles erreurs, mais uniquement en dehors des phases de mise à jour, c'est-à-dire quand $\sigma_{1-\varepsilon}(1-c) \approx 1$.

On a ainsi réalisé un composant permettant de mémoriser une valeur et de la mettre à jour à chaque fois qu'un signal l'ordonne. □

Un autre composant essentiel à la réalisation d'un GPAC calculant une fonction récursivement calculable est présenté dans le lemme suivant. Il s'agit d'un composant qui étant donnée une entrée x et une précision 2^{-n} donne un résultat précis à 2^{-n} près. Une représentation schématique de ce composant est donnée en figure 5.4.

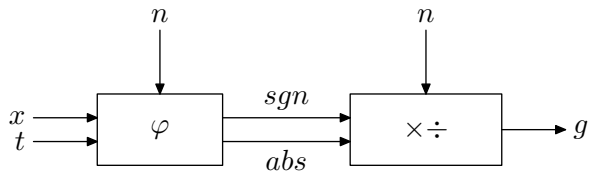

FIG. 5.4: Composant de calcul à précision donnée

Lemme 5.10 *Étant donnée une fonction $f : [a, b] \to \mathbb{R}$ récursivement calculable, il existe un* GPAC *qui associe aux entrées x et n une sortie approchant $f(x)$ à $2^{-\lfloor n \rfloor}$ près si $n - \lfloor n \rfloor \leq \frac{1}{2}$ après un certain temps.*

Démonstration : En utilisant la définition des fonctions récursivement calculables de [Ko91] qui est équivalente à celle que nous avons utilisée, on peut écrire que pour f récursivement calculable, il existe des fonctions discrètes calculables $abs : \mathbb{N}^2 \to \mathbb{N}$, $sgn : \mathbb{N}^2 \to \mathbb{N}$ et $m : \mathbb{N} \to \mathbb{N}$ vérifiant

$$\forall x \in \mathbb{R}, k, n \in \mathbb{N}, \left| \frac{k}{2^{m(n)}} - x \right| < 2^{-m(n)} \Rightarrow \left| \frac{(1 - 2sgn(k,n))abs(k,n)}{2^n} - f(x) \right| < 2^{-n}$$

Pour la clarté de la preuve, nous supposons pour l'instant que $n \in \mathbb{N}$. On a dans l'expression remplacé l'habituel $(-1)^{sgn(k,n)}$ par l'expression $(1 - 2sgn(k, n))$ qui lui est équivalente pour $sgn(k, n) \in \{0, 1\}$ et est définie de façon plus évidente sur tout l'intervalle $[0, 1]$.

Les fonctions m, sgn et abs étant calculables, il existe d'après la proposition 5.8 des GPAC qui les calculent. Le seul problème est alors de fournir à ces GPAC une entrée k qui convient en fonction de $m(n)$. En effet, si ce k est donné, on a des GPAC qui calculent sgn et abs ; il est facile pour un GPAC de calculer $\frac{(1-2sgn(k,n))abs(k,n)}{2^n}$ qui approche $f(x)$ à 2^{-n} près.

Étant donné n, choisir k tel que $\left| \frac{k}{2^{m(n)}} - x \right| < 2^{-m(n)}$ semble simple : il suffit de prendre $k = \lfloor x 2^{m(n)} \rfloor$. Malheureusement, la fonction partie entière n'est pas continue et donc pas θ_j-GPAC-calculable. Pour éviter ce problème, nous allons utiliser une fonction approchant la partie entière comme la fonction int définie dans la preuve du lemme 3.16. Cette définition suppose que $j = 3$. Dans les autres cas, il suffit de définir int_j comme suit :

$$int_j(x) = \frac{\int_0^x \theta_j(-\sin(2\pi(t + 1/4)))\,\mathrm{d}t}{\int_0^1 \theta_j(-\sin(2\pi(t + 1/4)))\,\mathrm{d}t}.$$

Cette fonction int (ou int_j) est égale à la fonction partie entière sur les intervalles de la forme $[n, n + \frac{1}{2}]$ avec n entier. Comme cette fonction ne renvoie un réel que dans un cas sur deux, une première idée propose d'utiliser une deuxième fonction décalée pour traiter les autres cas. Cependant, les cas tangents entre les cas où l'une des fonctions doit être utilisée et les autres cas (par exemple le cas de $\frac{1}{2}$) ne pourront être distingués par

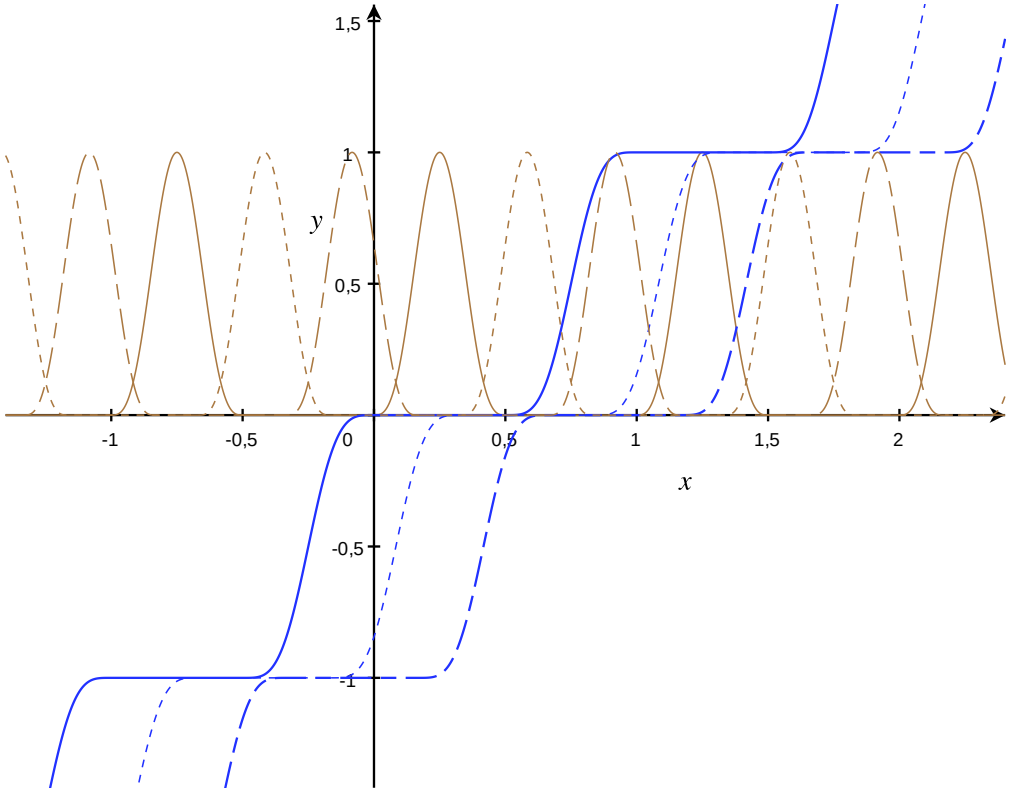

FIG. 5.5: Représentation graphique des fonctions r_0, r_1, r_2 et ω_0, ω_1, ω_2.

un processus continu et donc calculable. Nous allons donc en fait utiliser trois fonctions r_0, r_1 et r_2 définies par

$$r_i(x) = int\left(x - \frac{i}{3}\right).$$

Et pour savoir lequel des 3 candidats k choisir, nous allons utiliser des fonctions « détectrices » ω_0, ω_1 et ω_2 définie de façon similaire à la fonction ω de la preuve du lemme 3.16 :

$$\omega_i(x) = \theta_j(\sin(2\pi(x - i/3)))$$

On remarque que pour tout $i \in \{0, 1, 2\}$, si $\omega_i(x) > 0$, alors $r_i(x)$ est un entier. De plus, pour tout $x \in \mathbb{R}$, il existe un $i \in \{0, 1, 2\}$ tel que $\omega_i(x) > 0$. Donc pour tout x, nous sommes capables de calculer un entier qui vaut soit $\lfloor x \rfloor$ soit $\lfloor x \rfloor + 1$. Et donc les entrées passées à la simulation de machine de Turing donneront des résultats cohérents.

Appelons $\mathcal{U}$ un GPAC qui simule les machines de Turing qui sur les entrées k et n renvoient $abs(k, n)$ et $sgn(k, n)$. Comme nous avons trois entrées possibles pour k, nous allons dupliquer $\mathcal{U}$ en trois exemplaires $\mathcal{U}_0$, $\mathcal{U}_1$ et $\mathcal{U}_2$ liés respectivement à r_0, r_1 et r_2.

Nous avons alors des fonctions y_{sgn_0}, y_{abs_0}, y_{sgn_1}, y_{abs_1} et y_{sgn_2}, y_{abs_2}, sorties respectives des GPAC $\mathcal{U}_0$, $\mathcal{U}_1$ et $\mathcal{U}_2$. Pour tout x, au moins une de ces sorties est imprévisible : l'entrée

n'étant pas un entier, on ne peut savoir ce que va donner l'application de la simulation de machine de Turing sur cette entrée, mais ce résultat éventuellement incohérent va être multiplié par la fonction détectrice qui vaut 0 en ce x. On a vu que pour tout x, au moins un des $\omega_i(x)$ est strictement positif. Cela signifie que la somme des $\omega_i(x)$ est toujours strictement positive. On peut donc définir

$$y_{abs}(x,n,t) = \frac{\sum_{i=0}^{2} \omega_i(2^{m(n)}x) y_{abs_i}(2^{m(n)}x,t)}{\sum_{i=0}^{2} \omega_i(2^{m(n)}x)} \tag{5.3}$$

$$y_{sgn}(x,n,t) = \frac{\sum_{i=0}^{2} \omega_i(2^{m(n)}x) y_{sgn_i}(2^{m(n)}x,t)}{\sum_{i=0}^{2} \omega_i(2^{m(n)}x)} \tag{5.4}$$

On peut enfin poser

$$y_{approx}(x,n,t) = (1 - 2y_{sgn}(x,n,t))\frac{y_{abs}(x,n,t)}{2^n}.$$

Il est cependant à noter que ce résultat peut ne pas être le bon : r_2 renvoie parfois un résultat erroné. En effet, dans le cas où $2^{m(n)}x \in [k, k+1/6]$, $r_2(2^{m(n)}x) = \lfloor 2^{m(n)}x \rfloor - 1$. Néanmoins, l'erreur introduite ne modifie le résultat final qu'au plus d'un facteur 2^{-n} : on a en notant g la fonction approchée de f obtenue, $k = \lfloor 2^{m(n)}x \rfloor$, $k-1$ et k sont des bonnes valeurs pour approcher x, donc

$$\left| f\left(x - \frac{2^{m(n)}}{3}\right) - g(k-1,n) \right| < 2^{-n}$$

$$\left| f\left(x - \frac{2^{m(n)}}{3}\right) - g(k,n) \right| < 2^{-n}$$

et donc

$$|g(k,n) - g(k-1,n)| < 2^{-n+1}.$$

D'où finalement,

$$|f(x) - y_{approx}(x,n,t_f)| < \frac{3}{2^n}$$

L'erreur peut donc être corrigée en utilisant $n' = \lfloor n+2 \rfloor$ au lieu de n dans cette construction, et on obtient un GPAC (toutes nos constructions utilisent des fonctions et des opérateurs générées par équations différentielles polynomiales avec θ_j) dont la sortie converge en temps fini (le temps du calcul de la machine de Turing) vers une approximation à 2^{-n} près de $f(x)$. □

Nous allons maintenant montrer que l'on peut créer un composant horloge qui pour les entrées x et n annonce si le calcul d'un approximant de $f(x)$ à 2^{-n} près est terminé de façon certaine.

Lemme 5.11 *Il existe un θ_j-GPAC qui pour des entrées x et n renvoie en sortie une fonction $c(x,n,t) \in [0,1]$ qui vérifie*
- *pour tout x, n, il existe t_f tel que $c(x,n,t_f) = 1$,*
- *si $c(x,n,t) = 1$, alors le calcul d'un approximant de $f(x)$ à 2^{-n} est terminé.*

Démonstration : Nous avons vu dans la preuve du lemme précédent que l'on ne peut pas à l'aide d'un GPAC décider du k qui permettra de calculer une bonne approximation de $f(x)$. Cependant, il est possible de constater que l'on a dépassé cette valeur avec par exemple une fonction de la forme $\theta_j(1-\theta_j(x-2^{m(n)}k))$ qui vaut 0 tant que $x-2^{m(n)}k \geq 1$ et vaut 1 à partir du moment où $x \leq 2^{m(n)}k$. Donc si cette fonction atteint 1, on a au moins atteint le k cherché.

Pour réaliser notre horloge, nous allons implémenter dans un GPAC l'algorithme suivant :

$k = 0$
Tant que $\theta_j(1 - \theta_j(x - 2^{m(n)}k)) < 1$
Faire
    Réaliser le calcul de $approx(k,n)$
    $k = k + 1$
$c = 1$

Nous avons besoin d'utiliser la construction de Branicky pour incrémenter k à chaque fois que la variable représentant l'état de la machine de Turing atteint l'état final. Et de la réutiliser pour incrémenter c quand $\theta_j(1 - \theta_j(x - 2^{m(n)}k))$ atteint 1.

Notons y_q la fonction qui simule l'état de la machine de Turing, d l'état final de la machine de Turing et supposons qu'il s'agit du dernier état de cette machine, de cette façon, si la simulation de la machine de Turing approche d, nous sommes surs que le calcul est terminé. La simulation du calcul va être représentée par un système d'équations différentielles, parmi lesquelles y_q est une composante. Nous allons calculer k avec les fonctions y_1 et y_2 :

$$\begin{aligned} y_1' &= \lambda(1-y_1)^3\theta_j(\sin(2\pi t))\theta_j(y_q - d)(y_2 - y_1) \\ y_2' &= \lambda(y_1 - y_2)^3\theta_j(-\sin(2\pi t)) \\ y_1(0) &= 0 \\ y_2(0) &= 0 \end{aligned}$$

Nous allons simuler c par les fonction y_3 et y_4 suivantes :

$$\begin{aligned} y_3' &= \lambda(1-y_3)^3\theta_j(\sin(2\pi t))\theta_j(1 - \theta_j(x - 2^{m(n)}k))(y_4 - y_3) \\ y_4' &= \lambda(y_3 - y_4)^3\theta_j(-\sin(2\pi t)) \\ y_3(0) &= 0 \\ y_4(0) &= 0 \end{aligned}$$

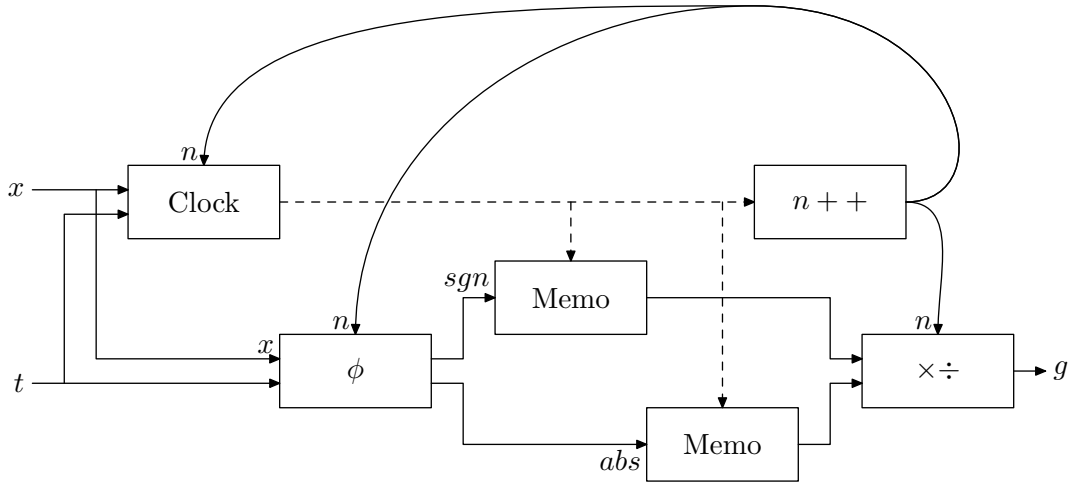

FIG. 5.6: Schéma d'un GPAC calculant une fonction récursivement calculable

La fonction y_3 représente donc une horloge signalant quand un calcul se termine. □

5.4.4 Résultat

Proposition 5.12 *Une fonction est θ_j-GPAC-calculable si elle est récursivement calculable sur un fermé.*

Démonstration : Il ne reste pour prouver cette proposition qu'à regrouper les trois lemmes précédents et mettre les composants construits dans une boucle où nous incrémenterons la précision à chaque pas comme il est suggéré dans la figure 5.2.

La preuve du lemme 5.11 montre déjà comment réaliser l'incrémentation d'une variable quand l'horloge indique la fin d'un calcul, de la même façon, on peut remettre à zéro les éléments de calcul et d'horloge avec les nouvelles entrées.

La connexion va se faire comme indiqué sur la figure 5.6. Le composant $n++$ prend en entrée le signal de l'horloge et donne en sortie n. À l'origine, n vaut 1, puis à chaque fois que l'entrée atteint 1, n est incrémenté d'une unité. Ce fonctionnement est réalisé par un système d'équations de la forme

$$y_1' = \lambda(1)^3\theta_j(\sin(2\pi t))\theta_j(y_4 - 1)$$
$$y_2' = \lambda(y_1 - y_2)^3\theta_j(-\sin(2\pi t))$$

où y_4 représente la valeur de la sortie de l'élément horloge.

La remise à zéro de l'horloge se fait par un système d'équations différentielles de la forme

$$y_3' = \lambda(-y_3)^3\theta_j(\sin(2\pi t))\theta_j(y_4 - 1)$$
$$y_4' = \lambda(y_3 - y_4)^3\theta_j(-\sin(2\pi t)).$$

Nous avons donc tous les ingrédients pour réaliser le circuit représenté en figure 5.6 dont la sortie g approche à 2^{-n} la valeur de $f(x)$. La fonction $\varepsilon(t)$ qui vaut 2^{-n} est donc une borne de l'erreur commise en approchant f par g. □

5.5 Les fonctions récursivement calculables sont GPAC-calculables

La section précédente montrait qu'étant donnée une fonction récursivement calculable f, on pouvait construire une équation différentielle avec θ_j dont une des composantes approche f et une autre borne l'erreur et tend vers 0. Nous voulons maintenant montrer que c'est possible sans l'utilisation de θ_j.

Pour ce faire, nous devons montrer que

1. θ_j peut être simulé par une fonction générable par GPAC, et donc analytique,
2. la construction de Branicky peut être adaptée pour ne pas utiliser θ_j. La précision en pâtira mais restera dans des limites raisonnables.

Une fois cela fait, il ne restera plus qu'à montrer que l'erreur faite avec cette nouvelle construction peut être bornée par une fonction générée par GPAC. L'idée principale est que comme la plupart des valeurs transmises dans les fils des circuits représentent des entiers, une erreur bornée permet de quand même avoir une bonne idée de la valeur cible.

5.5.1 Résultats préliminaires

Nous allons pour réécrire nos équations sans θ_j utiliser des fonctions de contraction d'erreur σ et l_2 qui sont trivialement pCp.

Définition 5.13 (σ)

$$\sigma : x \mapsto x - \frac{1}{5}\sin(2\pi x)$$

Cette fonction stabilise l'erreur auprès des entiers de façon statique contrairement à la fonction l_2 suivante qui permet de contrôler dynamiquement l'erreur.

Définition 5.14 (l_2)

$$l_2 : (x, y) \mapsto \frac{1}{\pi}\arctan\left(4y\left(x - \frac{1}{2}\right)\right) + \frac{1}{2}$$

l_2 est une fonction qui contracte l'erreur autour de 0 et 1 à une valeur bornée par $\frac{1}{y}$. Plus précisément, l_2 vérifie l'assertion suivante : pour $\tilde{a} \in \left]-\infty; \frac{1}{4}\right[\cup \left]\frac{3}{4}; +\infty\right[$, on a pour tout $y > 0$, $|l_2(\tilde{a}, y) - a| < \frac{1}{y}$.

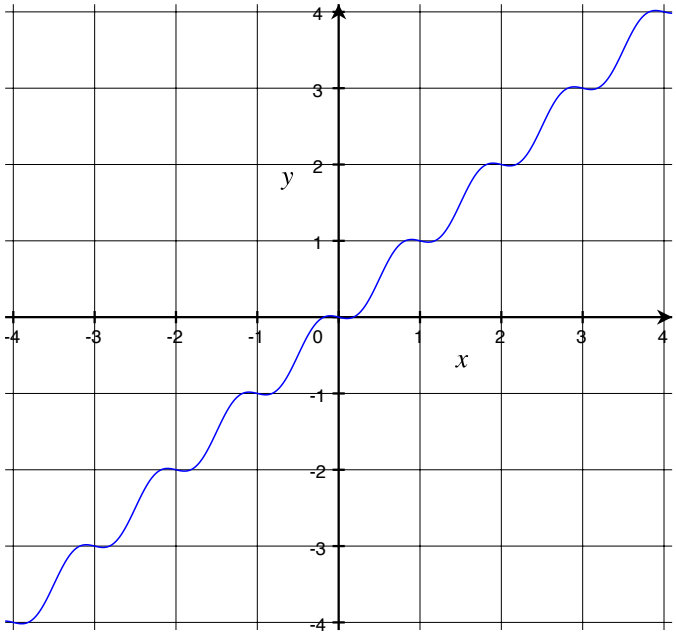

FIG. 5.7: Représentation graphique de σ

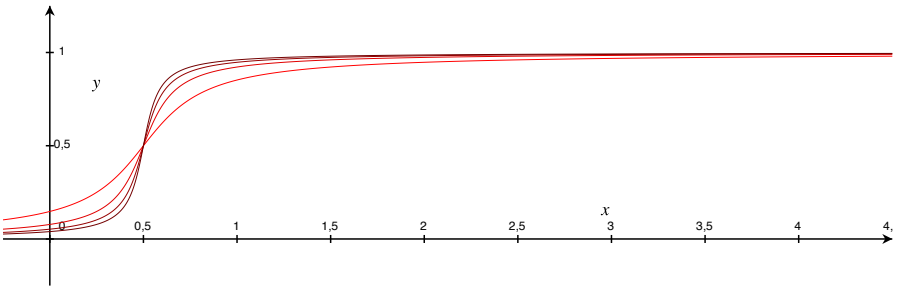

FIG. 5.8: Représentation graphique de $l_2(x, y)$ pour $y \in \{1, 2, 3, 4\}$

5.5.2 Composant de mémorisation

Le composant de mémorisation défini dans la section précédente est utilisé pour stocker les valeurs de *sgn* et *abs* obtenues à l'aide des simulations de machines de Turing. Nous devons nous passer des θ_j et des σ_ε et $\sigma_{1-\varepsilon}$ pour écrire des équations sans θ_j. Pour cela, nous allons devoir créer une fonction $T : \mathbb{R} \to \mathbb{R}$ vérifiant

$$T(c) \begin{cases} \geq 3/4 & \text{si } c > 1-\varepsilon \\ \leq 1/4 & \text{si } c < \varepsilon. \end{cases}$$

On peut considérer que le ε utilisé dans la définition de notre composant de mémorisation est supérieur à $1/4$. La fonction $x \mapsto l_2(x, 1)$ vérifie alors bien nos conditions. Il est maintenant possible de remplacer es équations (5.1) et (5.2) définissant la sortie du composant de mémoire par

$$y_1' = \lambda(\sigma(f) - y_1)^3 l_2(\sin(2\pi t), m) l_2(T(c), n) + \lambda(y_2 - y_1)^3 l_2(\sin(2\pi t), m) l_2(1 - T(c), n)$$

$$y_2' = \lambda(y_1 - y_2)^3 l_2(\sin(2\pi t), m)$$

avec des m et n choisis pour obtenir une précision suffisante.

Ces équations caractérisent manifestement des équations pCp. De plus, on sait que l'erreur commise par $l_2(., n)$ est inférieure à $1/n$, donc pour obtenir sur y_1 une erreur inférieure à $1/4$, on doit poser des n et m suffisamment grands pour que $\frac{\lambda(\sigma(f)-y_1)^3}{mn} < \frac{1}{8}$.

5.5.3 Réinitialisation

La remise à zéro pour commencer un nouveau calcul est présentée dans la preuve de la proposition 5.12. Il s'agit d'une part d'incrémenter n et d'autre part de mettre l'horloge à 0. Il suffit *a priori* de réaliser le même remplacement dans les équations que celui que nous avons réalisé ci-dessus à savoir utiliser $l_2(\sin(2\pi t), m)$ au lieu de $\theta_j(\sin(2\pi t))$. Cependant, l'erreur faite sur n doit être bornée de façon précise car on prendra ensuite $2^{m(n)}$ à partir de la valeur calculée de n dans le composant de calcul à précision donnée.

5.5.4 Calcul à précision donnée

Rappelons tout d'abord que la simulation de machine de Turing que nous utilisons est robuste à l'erreur, c'est à dire donne un résultat correct (à $1/4$ près) à condition que l'entrée soit donnée avec une précision de $1/4$.

La suppression des θ_j implique de se débarasser des r_i et ω_i. Mais du fait de la robustesse de la simulation de machine de Turing, nous pouvons conserver les résultats obtenus pour une entrée à $1/4$ près. Nous allons donc utiliser des translations de σ pour remplacer les r_i et des translations de

$$W : t, y \mapsto l_2\left(\frac{1}{2}\left(\sin^2(2\pi t) + \sin(2\pi t)\right), y\right)$$

avec un y représentant la précision voulue pour remplacer les ω_i.

Du fait de l'erreur commise, nous allons devoir soit agrandir les intervalles sur lesquels les translations de σ approchent avec une bonne précision la partie entière, soit utiliser plus de telles fonctions. Nous choisissons la deuxième solution. Nous allons utiliser 6 fonctions au lieu des 3 que nous avons choisies dans la partie avec θ_j.

Définissons pour $i \in \{0,1,2,3,4,5\}$ les fonctions suivantes :

$$\rho_i : x \mapsto \sigma\left(x - \frac{1}{4} - \frac{i}{6}\right)$$

$$W_i : x, y \mapsto l_2\left(\frac{1}{2}\left(\sin^2(2\pi(x - i/6)) + \sin(2\pi(x - i/6))\right), y\right).$$

Remarquons que pour $k \in \mathbb{N}, x \in [k + 1/2, k + 1]$, on a $|W_0(x, y)| < \frac{1}{y}$. On peut donc dire que y permet de contrôler l'erreur commise quand le ρ_i correspondant n'est pas une entrée adéquate. D'autre part, une étude rapide montre que pour $x \in [0,16; 0,34]$, on a $\frac{1}{2}\left(\sin^2(2\pi t) + \sin(2\pi t)\right) \in \left[\frac{3}{4}; 1\right]$. On a donc un intervalle de taille supérieure à $\frac{1}{6}$ sur lequel W_0 est clairement non nul. Ce qui montre que 6 fonctions suffisent à couvrir $[0; 1]$.

Si l'on appelle $y_{abs_i}(2^{m(n)}x, t)$ la sortie qui représente la valeur absolue de la simulation de machine de Turing au temps t sur l'entrée $\rho_i(2^{m(n)}x)$, le résultat pour les 6 machines en parallèle va être

$$y_{abs}(x, n, t) = \frac{\sum_{i=0}^{5} W_i(2^{m(n)}x, \gamma) \times y_{abs_i}(2^{m(n)}x, t)}{\sum_{i=0}^{5} W_i(2^{m(n)}x, \gamma)}.$$

Le cas de y_sgn est identique. Le choix de γ doit être fait de façon à avoir une précision suffisante sur y_{abs}. On obtient avec γ suffisamment grand un résultat approchant abs à $\varepsilon > 0$ près pour un ε fixé. Le principe est le même que pour le choix de m et n dans le paragraphe 5.5.2 à savoir avoir un γ dynamique dépendant des erreurs produites par les autres composants pour contrôler cette erreur.

5.5.5 Bornes sur l'erreur

L'erreur commise lors de ce calcul provient de deux facteurs : d'une part, l'erreur qui existe déjà dans le calcul avec θ_j à savoir,

$$\left|(1 - 2sgn(k, n))\frac{abs(k, n)}{2^n} - f(x)\right| < \frac{1}{2^n};$$

d'autre part l'erreur commise sur n, y_{abs}, y_{sgn}.

Nous avons obtenu des approximations de $2^{m(n)}$, abs et sgn à ε près. Cela signifie que l'erreur totale commise pour approcher $f(x)$ est majorée par

$$2^{-n} + \left(\frac{abs(k, n) - \varepsilon}{2^n + \varepsilon} - \frac{abs(k, n) + \varepsilon}{2^n - \varepsilon}\right).$$

Cette erreur est donc majorée par

$$2^{-n} + \frac{2\varepsilon}{2^n - \varepsilon}.$$

En considérant $n \geq 1$ et en prenant ε petit, on obtient donc une erreur de l'ordre de 2^{1-n}.

5.5.6 Résultat

Les paragraphes précédents montrent qu'il est possible de se débarrasser des θ_j dans la preuve de la proposition 5.12. On peut donc simuler toute fonction récursivement calculable par une fonction GPAC-calculable.

Proposition 5.15 *Si une fonction est récursivement calculable sur un fermé, alors elle est* GPAC-*calculable.*

Démonstration : Reprenons la preuve de la proposition 5.12. Nous avons montré qu'une fonction récursivement calculable f était θ_j-GPAC-calculable. Les paragraphes de cette section ont montré qu'il était possible de supprimer les θ_j qui apparaissent dans cette preuve et de toujours avoir un GPAC qui approchait $f(x)$ en $O(2^{-n})$. □

Nous avons obtenu par les propositions 5.6 et 5.15 les deux sens du théorème 5.4 qui incarne le méta-théorème 4 pour des fonctions définies sur des fermés. Nous avons donc prouvés que pour les fonctions définies sur des fermés, les notions de GPAC-calculabilité et de calculabilité au sens de l'analyse récursive se recouvrent.

Quatrième partie

Épilogue

Conclusion

Dans la partie I, nous avons rappelé les définitions de quelques modèles de calcul sur les entiers, nous avons ensuite montré des modèles de calcul sur les réels en distinguant les modèles dits hybrides et les modèles continus.

Dans la partie II, nous avons montré tout d'abord que l'on pouvait définir une classe de fonctions sur les réels par clôture dont la partie discrète caractérise les fonctions récursives discrètes. Nous avons ensuite montré qu'à l'aide d'un opérateur de limites, nous pouvions étendre des résultats sur la partie discrètes de fonctions en des résultats sur les réels. Nous avons ainsi caractérisé par des classes de fonctions définies par clôture les classes des fonctions récursivement calculables, des fonctions élémentairement calculables et des fonctions calculables dans la hiérarchie de Grzegorczyk.

Dans la partie III, nous avons exhibé un lien entre les fonctions GPAC-calculables et les fonctions récursivement calculables. Ce résultat utilise une définition particulière de la GPAC-calculabilité qui repose sur un calcul convergeant.

En résumé, pour $n > 3$,

$$
\begin{array}{ccccccc}
\mathscr{E}(\mathbb{N}) & & \mathscr{E}_n(\mathbb{N}) & & \mathcal{PR}(\mathbb{N}) & & \mathcal{R}ec(\mathbb{N}) \\
\wr\!\!\wr^1 & & \wr\!\!\wr^1 & & \wr\!\!\wr^1 & & \wr\!\!\wr^1 \\
\mathscr{L} & \subsetneq & \mathscr{L}_n & \subseteq & \bar{\mathscr{D}} & \subsetneq & \mathscr{L}+!\mu \\
\cap\!\!| & & \cap\!\!| & & \cap\!\!| & & \cap\!\!| \\
\mathscr{L}^\star & \subsetneq & \mathscr{L}_n^\star & \subsetneq & (\bar{\mathscr{D}}+\theta_3)^\star & \subsetneq & \mathscr{L}_{!\mu}^\star \\
\| & & \| & & \cup\!\!| & & \| \\
\mathscr{E}(\mathbb{N})\cap\mathscr{C}^2(\mathbb{F}) & \subsetneq & \mathscr{E}_n(\mathbb{R})\cap\mathscr{C}^2(\mathbb{F}) & \subsetneq & \mathcal{PR}(\mathbb{R})\cap\mathscr{C}^2(\mathbb{F}) & \subsetneq & \mathcal{R}ec(\mathbb{R})\cap\mathscr{C}^2(\mathbb{K}_\mathbb{Q}) \\
 & & & & & & \| \\
 & & & & & & \text{GPAC-calculable}
\end{array}
$$

où $\mathscr{C}^2(\mathbb{F})$ représente l'ensemble des fonctions de classe $\mathscr{C}^2$ sur un fermé de $\mathbb{R}^d$, et $\mathscr{C}^2(\mathbb{K}_\mathbb{Q})$ l'ensemble des fonctions $\mathscr{C}^2$ sur un compact à bornes rationnelles.

Ces résultats montrent en quelque sorte que les modèles des fonctions $\mathbb{R}$-récursives, et GPAC-calculables, à quelques adaptations près, sont équivalents à la classe des fonctions récursivement calculables. En d'autres termes, nous avons établi un premier pas vers une thèse à la Church-Turing pour les fonctions sur les réels : toutes les fonctions raisonnables sont calculées par GPAC avec limite, que ce soient les fonctions $\mathbb{R}$-récursives à opérateurs physiquement raisonnables ou les fonctions générées par l'analyse récursive.

Cela constitue cependant un objectif à long terme, qui demanderait de s'assurer que les fonctions de l'analyse récursive sont exactement les fonctions raisonnables sur les réels.

[1] Le signe $\approx$ signifie ici « modulo DP ». Formellement, $\mathscr{E}(\mathbb{N}) \approx \mathscr{L}$ signifie que $\mathscr{E}(\mathbb{N}) = DP(\mathscr{L})$.

En effet il n'est pas certain que l'analyse récursive soit *le bon modèle*. Les tenants de l'analyse récursive affirment que seules les fonctions continues sont raisonnables, dénonçant ainsi l'impossibilité physique de la comparaison de deux nombres réels, ce qui est par exemple réalisable dans le modèle BSS, mais dans ce cas, les fonctions non-dérivables sont-elles raisonnables ? Les fonctions générées par GPAC sont assurément raisonnables, mais cette classe est faible, et il semblait naturel de l'étendre par un processus de limite comme nous l'avons fait. Il n'est en effet pas intuitivement stupide de programmer la machine et devoir attendre un certain temps pour obtenir un résultat à une précision satisfaisante. Peut-être obtiendrait-on un meilleur modèle en utilisant un opérateur de limite plus restreint. On se heurte en fait au problème soulevé dans [Cop02] : la thèse de Church-Turing peut être interprétée de différentes façons, et le problème que nous aurions voulu résoudre n'est-il pas « quelle est la classe de fonctions qui peuvent et pourront être calculées par un dispositif quelconque ?» Question qui réclame de connaître précisément et parfaitement le monde physique.

Nos résultats sont cependant intéressants puisqu'ils donnent une équivalence dans le modèle de l'analyse récursive qui est assez reconnu. Nous avons aussi exhibé des classes de fonctions qui caractérisent les fonctions élémentaires et les niveaux ≥ 3 de la hiérarchie de Grzegorczyk. Quelques points mériteraient d'être étudiés pour parfaire cette étude, par exemple, notre caractérisation des fonctions récursivement calculables par des fonctions $\mathbb{R}$-récursives n'est valable que pour les fonctions définies sur un compact à bornes rationnelles. Il serait intéressant de savoir ce qu'il en est des fonctions définies sur tout $\mathbb{R}$. Également, sur la classe $\mathscr{L}^{\star}_{!\mu}$, nous avons montré des résultats inspirés de la théorie de la calculabilité : fonction universelle, théorème de point fixe. D'autres résultats comme le théorème s-n-m pourraient être étudiés.

Une des propriétés utiles sur les modèles de calcul que nous n'avons pas étudiée est la robustesse. Ainsi, il est intéressant de savoir si une légère perturbation des conditions initiales de la machine aboutira à un résultat différent ou non. Cette question a été étudiée sur les modèles à dérivée constante par morceaux et quelques autres modèles dans [AB01]. Pour le cas de l'analyse récursive, la question de la robustesse est traitée par Pieter Collins et ses coauteurs dans [Col05a, Col05b, CL05, CvS04].

Un point qui pourrait être étudié également est la question de la complexité. Nous nous sommes dans cette thèse intéressé à des questions de calculabilité sur les réels. Nous avons cependant vu qu'il existe une notion de complexité en analyse récursive, et savoir comment cette complexité se transmet pour les fonctions $\mathbb{R}$-récursives serait intéressant. L'article [CO06] pourrait être un pas pour l'étude de cette question : Manuel Campagnolo et Kerry Ojakian y ont en effet montré ce qu'on pourrait qualifier de théorème de transfert permettant d'étendre des résultats sur des classes de fonctions discrètes à des classes de fonctions sur les réels.

Bibliographie

[AB01] Eugene Asarin et Ahmed Bouajjani – « Perturbed Turing machines and hybrid systems », *16th Annual IEEE Symposium on Logic in Computer Science*, 2001, p. 269.

[AM98] Eugene Asarin et Oded Maler – « Achilles and the tortoise climbing up the arithmetical hierarchy », *Journal of Computer and System Sciences* **57** (1998), no. 3, p. 389–398.

[BCGH06] Olivier Bournez, Manuel L. Campagnolo, Daniel S. Graça et Emmanuel Hainry – « The general purpose analog computer and computable analysis are two equivalent paradigms of analog computation », *Theory and Applications of Models of Computation, TAMC 2006* (J.-Y. Cai, S. B. Cooper et A. Li, éds.), Lecture Notes in Computer Science, vol. 3959, Springer, 2006, p. 631 – 643.

[BCSS98] Lenore Blum, Felipe Cucker, Mike Shub et Steve Smale – *Complexity and real computation*, Springer, 1998.

[BH98] Vasco Brattka et Peter Hertling – « Feasible real random access machines », *Journal of Complexity* **14** (1998), no. 4, p. 490–526.

[BH04] Olivier Bournez et Emmanuel Hainry – « An analog characterization of elementary computable functions over the real numbers », *International Colloquium on Automata, Languages and Programming (ICALP 2004)* (J. Díaz, J. Karhumäki, A. Lepistö et D. Sannella, éds.), Lecture Notes in Computer Science, vol. 3142, 2004, p. 269–280.

[BH05a] — , « Elementary computable functions over the real numbers and R-sub-recursive functions », *Theoretical Computer Science* **348** (2005), no. 2-3, p. 130–147.

[BH05b] — , « Real recursive functions and real extentions of recursive functions », *Machines, Computations, and Universality, MCU 2004* (M. Margenstern, éd.), Lecture Notes in Computer Science, vol. 3354, Springer-Verlag, 2005, p. 116–127.

[BH06] — , « Recursive analysis characterized as a class of real recursive functions », *Fundamenta Informaticae* **74** (2006), no. 4, p. 409–433.

[Bou99] Olivier Bournez – « Complexité algorithmique des systèmes dynamiques continus et hybrides », Thèse de doctorat, École Normale Supérieure de Lyon, janvier 1999.

[Bra95] Michael S. Branicky – « Universal computation and other capabilities of hybrid and continuous dynamical systems », *Theoretical Computer Science* **138** (1995), no. 1, p. 67–100.

[Bra03] Vasco Brattka – « Computability over topological structures », Computability and Models (S. B. Cooper et S. S. Goncharov, éds.), Kluwer Academic Publishers, New York, 2003, p. 93–136.

[BSS89] Lenore Blum, Mike Shub et Steve Smale – « On a theory of computation and complexity over the real numbers : NP-completeness, recursive functions and universal machines », *Bulletin of the American Mathematical Society* **21** (1989), no. 1, p. 1–46.

[Bus31] Vannevar Bush – « The differential analyzer. A new machine for solving differential equations », *Journal of the Franklin Institute* **212** (1931), p. 447–488.

[Cam01] Manuel L. Campagnolo – « Computational complexity of real valued recursive functions and analog circuits », Thèse de doctorat, IST, Universidade Técnica de Lisboa, 2001.

[CL05] Pieter Collins et John Lygeros – « Computability of finite-time reachable sets for hybrid systems », *44th IEEE Conference on Decision and Control and the European Control Conference*, 2005.

[Clo98] Peter Clote – « Computational models and function algebras », Handbook of Computability Theory (E. R. Griffor, éd.), North-Holland, Amsterdam, 1998, p. 589–681.

[CM01] Manuel L. Campagnolo et Cristopher Moore – « Upper and lower bounds on continuous-time computation », *2nd International Conference on Unconventional Models of Computation - UMC'2K* (I. Antoniou, C. Calude et M. Dinneen, éds.), Springer, 2001, p. 135–153.

[CM05] José Félix Costa et Jerzy Mycka – « What lies beyond the mountains ? », *EATCS Bulletin* **85** (2005), p. 179–189.

[CMC00a] Manuel L. Campagnolo, Cristopher Moore et José Félix Costa – « An analog characterization of the subrecursive functions », *4th Conference on Real Numbers and Computers* (P. Kornerup, éd.), Odense University Press, 2000, p. 91–109.

[CMC00b] — , « Iteration, inequalities, and differentiability in analog computers », *Journal of Complexity* **16** (2000), no. 4, p. 642–660.

[CMC02] — , « An analog characterization of the Grzegorczyk hierarchy », *Journal of Complexity* **18** (2002), no. 4, p. 977–1000.

[CO06] Manuel L. Campagnolo et Kerry Ojakian – « The methods of approximation and lifting in real computation », *Third International Conference on Computability and Complexity in Analysis (CCA 2006)*, 2006.

[Col05a] Pieter Collins – « Hybrid trajectory spaces », 2005.

[Col05b] — , « On the computability of reachable and invariant sets », *44th IEEE Conference on Decision and Control and the European Control Conference*, 2005.

[Cop98] B. Jack Copeland – « Even Turing machines can compute uncomputable functions », *Unconventional Models of Computation (UMC'98)* (J. Casti, C. Calude et M. Dinneen, éds.), 1998, p. 150–164.

[Cop02] — , « The Church-Turing thesis », The Stanford Encyclopedia of Philosophy (E. N. Zalta, éd.), `http://plato.stanford.edu/entries/church-turing/`, 2002.

[Cut80] Nigel Cutland – *Computability : An introduction to recursive function theory*, Cambridge University Press, 1980.

[CvS04] Pieter Collins et Jan H. van Schuppen – « Observability of piecewise-affine hybrid systems », *Hybrid Systems : Computation and Control : 7th International Workshop, HSCC 2004* (R. Alur et G. J. Pappas, éds.), Lecture Notes in Computer Science, vol. 2993, Springer, 2004, p. 265–279.

[DL03] Jérôme Durand-Lose – « Calculer géométriquement sur le plan — machines à signaux — », Habilitation à diriger des recherches, Université de Nice-Sophia Antipolis, décembre 2003.

[DL05] — , « Abstract geometrical computation : Turing-computing ability and undecidability », *CiE* (S. B. Cooper, B. Löwe et L. Torenvliet, éds.), Lecture Notes in Computer Science, vol. 3526, 2005, p. 106–116.

[EN02] Gábor Etesi et István Németi – « Non-Turing computations via Malament-Hogarth space-times », *International Journal Theoretical Physics* **41** (2002), p. 341–370.

[GC03] Daniel S. Graça et José Félix Costa – « Analog computers and recursive functions over the reals », *Journal of Complexity* **19** (2003), no. 5, p. 644–664.

[GCB05] Daniel S. Graça, Manuel L. Campagnolo et Jorge Buescu – « Robust simulations of Turing machines with analytic maps and flows », *CiE 2005 : New Computational Paradigms* (S. B. Cooper, B. Löwe et L. Torenvliet, éds.), Lecture Notes in Computer Science, vol. 3526, Springer, 2005, p. 169–179.

[Gra04] Daniel S. Graça – « Some recent developments on Shannon's General Purpose Analog Computer », *Mathematical Logic Quarterly* **50** (2004), no. 4-5, p. 473–485.

[Grz57] Andrzej Grzegorczyk – « On the definitions of computable real continuous functions », *Fundamenta Mathematicae* **44** (1957), p. 61–71.

[GZB06] Daniel S. Graça, Ning Zhong et Jorge Buescu – « The ordinary differential equation defined by a computable function whose maximal interval of existence is non-computable », *7th Conference on Real Numbers and Computers (RNC 7)*, 2006.

[Hog92] Mark L. Hogarth – « Does general relativity allow an observer to view an eternity in a finite time ? », *Foundations of Physics Letters* **5** (1992), p. 173–181.

[Kaw05] Akitoshi Kawamura – « Type-2 computability and Moore's recursive functions », *6th Workshop on Computability and Complexity in Analysis (CCA 2004)* (V. Brattka, L. Staiger et K. Weihrauch, éds.), Electronic Notes in Theoretical Computer Science, vol. 120, Elsevier, 2005, p. 83–95.

[KM99] Pascal Koiran et Cristopher Moore – « Closed-form analytic maps in one and two dimensions can simulate universal Turing machines », *Theoretical Computer Science* **210** (1999), no. 1, p. 217–223.

[Ko91] Ker-I Ko – *Computational complexity of real functions*, Birkhäuser, 1991.

[Lac55] Daniel Lacombe – « Extension de la notion de fonction récursive aux fonctions d'une ou plusieurs variables réelles III », *Comptes Rendus de l'Académie des Sciences Paris* **241** (1955), p. 151–153.

[LFA74] Jacqueline Lelong-Ferrand et Jean-Marie Arnaudiès – *Cours de mathématiques, tome 2 : analyse*, Dunod, 1974.

[LR87] Leonard Lipshitz et Lee A. Rubel – « A differentially algebraic replacement theorem, and analog computability », *Proceedings of the American Mathematical Society* **99** (1987), no. 2, p. 367–372.

[MC04] Jerzy Mycka et José Félix Costa – « Real recursive functions and their hierarchy », *Journal of Complexity* **20** (2004), no. 6, p. 835–857.

[MC05] — , « The computational power of continuous dynamic systems », *Machines, Computations, and Universality, MCU 2004* (M. Margenstern, éd.), Lecture Notes in Computer Science, vol. 3354, Springer, 2005, p. 164–175.

[MC06] — , « Undecidability over continuous time », *Logic Journal of the IGPL* **14** (2006), no. 5, p. 649–658, Oxfrod University Press.

[Mil95] Jonathan Mills – « Programmable VLSI extended analog computer for cyclotron beam control », Tech. Report 441, Indiana University Computer Science, September 1995.

[Min67] Marvin Minsky – *Computation : Finite and infinite machines*, Prentice-Hall, 1967.

[Moo96] Cristopher Moore – « Recursion theory on the reals and continuous-time computation », *Theoretical Computer Science* **162** (1996), no. 1, p. 23–44.

[Mül86] Norbert Th. Müller – « Subpolynomial complexity classes of real functions and real numbers », *International Colloquium on Automata, Languages and Programming (ICALP 86)* (L. Kott, éd.), Lecture Notes in Computer Science, vol. 226, Springer, 1986, p. 284–293.

[Myc03a] Jerzy Mycka – « Infinite limits and R-recursive functions », *Acta Cybernetica* **16** (2003), no. 1, p. 83–91.

[Myc03b] — , « μ-recursion and infinite limits », *Theoretical Computer Science* **302** (2003), p. 123–133.

[Myh71] John Myhill – « A recursive function, defined on a compact interval and having a continuous derivative that is not recursive », *The Michigan Mathematical Journal* **18** (1971), p. 97–98.

[Odi92] Piergiorgio Odifreddi – *Classical recursion theory, volume I*, Studies in Logic and the Foundations of Mathematics, vol. 125, North-Holland, avril 1992.

[Orp97] Pekka Orponen – « A survey of continous-time computation theory », *Advances in Algorithms, Languages, and Complexity* (D.-Z. Du et K.-I. Ko, éds.), Kluwer Academic Publishers, 1997, p. 209–224.

[PE74] Marian B. Pour-El – « Abstract computability and its relations to the general purpose analog computer », *Transactions of the American Mathematical Society* **199** (1974), p. 1–28.

[PER89] Marian B. Pour-El et J. Ian Richards – *Computability in analysis and physics*, Springer, 1989.

[Ros84] Harvey E. Rose – *Subrecursion : Functions and hierarchies*, Clarendon Press, 1984.

[Rub93] Lee A. Rubel – « The extended analog computer », *Adv. Appl. Math.* **14** (1993), p. 39–50.

[Sha41] Claude E. Shannon – « Mathematical theory of the differential analyzer », *J. Math. Phys. MIT* **20** (1941), p. 337–354.

[Sie99] Hava T. Siegelmann – *Neural networks and analog computation : Beyond the turing limit*, Birkhäuser, 1999.

[Sip97] Michael Sipser – *Introduction to the theory of computation*, PWS Publishing Company, 1997.

[Smi06] Warren D. Smith – « Church's thesis meets the N-body problem », *Applied Mathematics and Computation* **178** (2006), no. 1, p. 154–183.

[SS94] Hava T. Siegelmann et Eduardo D. Sontag – « Analog computation via neural networks », *Theoretical Computer Science* **131** (1994), no. 2, p. 331–360.

[Sta01] Mike Stannet – « Hypercomputation is experimentally irrefutable », Tech. report, Dept. of Computer Science, Sheffield University, UK, 2001.

[TLK76] William Thomson (Lord Kelvin) – « On an instrument for calculating the integral of the product of two given functions », *Proceedings of the Royal Society of London*, vol. 24, 1876, p. 266–276.

[Tur36] Alan. M. Turing – « On computable numbers, with an application to the Entscheidungsproblem », *Proceedings of the London Mathematical Society* **2** (1936), no. 42, p. 230–265.

[Wei00] Klaus Weihrauch – *Computable analysis : an introduction*, Springer, 2000.

[WZ06] Klaus Weihrauch et Ning Zhong – « Beyond the first main theorem – when is the solution of a linear Cauchy problem computable ? », *Theory and*

Applications of Models of Computation, TAMC 2006 (J.-Y. Cai, S. B. Cooper et A. Li, éds.), Lecture Notes in Computer Science, vol. 3959, Springer, 2006, p. 783–792.

[Zho97] Qing Zhou – « Subclasses of computable real valued functions », *Computing and Combinatorics* (T. Jiang et D. T. Lee, éds.), Lecture Notes in Computer Science, vol. 1276, Springer, 1997, p. 156–165.

Table des figures

Liste des définitions

INSTITUT NATIONAL
POLYTECHNIQUE
DE LORRAINE

AUTORISATION DE SOUTENANCE DE THESE
DU DOCTORAT DE L'INSTITUT NATIONAL
POLYTECHNIQUE DE LORRAINE

o0o

VU LES RAPPORTS ETABLIS PAR :
Monsieur Giuseppe LONGO, Directeur de Recherche, Ecole Normale Supérieure, Paris
Monsieur Serge GRIGORIEFF, Professeur, LIAFA, Université Paris 7, Paris

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

Monsieur HAINRY Emmanuel

NANCY BRABOIS
2, AVENUE DE LA
FORET-DE-HAYE
BOITE POSTALE 3
F - 54501
VANDŒUVRE CEDEX

à soutenir devant un jury de l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE, une thèse intitulée :

"Modèles de calcul sur les réels, résultats de comparaison"

en vue de l'obtention du titre de :

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

Spécialité : « **Informatique** »

Fait à Vandoeuvre, le 20 novembre 2006
Le Président de l'I.N.P.L.
L. SCHUFFENECKER

TEL. 33/03.83.59.59.59
FAX. 33/03.83.59.59.55

Résumé

Il existe de nombreux modèles de calcul sur les réels. Ces différents modèles calculent diverses fonctions, certains sont plus puissants que d'autres, certains sont deux à deux incomparables. Le calcul sur les réels est donc de ce point de vue bien différent du calcul sur les entiers qui est unifié par la thèse de Church-Turing qui affirme que tous les modèles raisonnables calculent les mêmes fonctions.

Les résultats de cette thèse sont de deux sortes. Premièrement, nous montrons des équivalences entre les fonctions récursivement calculables et une certaine classe de fonctions $\mathbb{R}$-récursives et entre les fonctions GPAC-calculables et les fonctions récursivement calculables. Ces deux résultats ne sont cependant valables que si les fonctions présentent quelques caractéristiques : elles doivent être définies sur un compact et dans le premier cas être de classe $\mathscr{C}^2$. Deuxièmement, nous montrons également une hiérarchie de classes de fonctions $\mathbb{R}$-récursives qui caractérisent les fonctions élémentairement calculables, les fonctions $\mathscr{E}_n$-calculables pour $n \geq 3$ (où les $\mathcal{E}_n$ sont les fonctions de la hiérarchie de Grzegorczyk), et des fonctions récursivement calculables. Ce résultat utilise un opérateur de limite dont nous avons prouvé la généralité en montrant qu'il transfère une inclusion sur la partie discrète des fonctions en une inclusion sur les fonctions sur les réels elles-mêmes.

Ces résultats constituent donc une avancée vers une éventuelle unification des modèles de calcul sur les réels.

Mots clefs : Analyse récursive, calculabilité réelle, fonctions élémentaires, hiérarchie de Grzegorczyk, General Purpose Analog Computer

Abstract

Computation on the real numbers can be modelised in several different ways. There indeed exist a lot of different computation models on the reals. However, there exist few results for comparing those models, and most of these results are incomparability results. The case of computation over the real numbers hence is quite different from that of computation over integer numbers where Church-Turing's thesis claims that all reasonable models compute exactly the same functions.

The results presented in this document are twofold. One, we show that recursively computable functions (in the sense of computable analysis) can be shown equivalent to some adequately defined subclass of $\mathbb{R}$-recursive functions, and also to GPAC-computable functions with GPAC-computable roughly meaning computable through a converging GPAC. Hence we get a machine independent characterization of recursively computable functions, and a bridge between type 2-machines and GPAC. Two, more than an analog characterization of recursively enumerable functions, we show that the limit operator we defined can be used to provide an analog characterization of elementarily computable functions and $\mathscr{E}_n$-computable functions for $n \geq 3$, where $\mathscr{E}_n$ represents the levels of Grzegorczyk's hierarchy.

Those results can be seen as a first step toward a unification of computable functions over the reals.

Keywords: Recursive analysis, analog computation, elementary functions, Grzegorczyk hierarchy, General Purpose Analog Computer