Approche de la calculabilité sur les réels

d'après un document de Klaus Weihrauch

Vincent Balat

Rapport de stage
de première année du magistère d'informatique de l'ENS Lyon,
effectué à l'Institut de Mathématiques de Luminy, Marseille
sous la direction de M. François Blanchard

juillet 1996

Remerciements

Je tiens à remercier particulièrement mon directeur de stage, François Blanchard, pour ses précieux conseils, et pour la liberté de travail qu'il m'a laissée.

Je voudrais également exprimer ma reconnaissance à tous les membres du laboratoire qui m'ont aidé directement ou indirectement dans mon travail, et à ceux qui ont contribué à rendre mon séjour à Marseille agréable, en particulier à Laurent Vuillon, Pascal Hubert, Ali Messaoudi et Pierre Barthélémy.

Table des matières

Introduction

Le développement des ordinateurs a entraîné la création d'une théorie pour essayer de définir précisément et d'étudier les opérations réalisables par des machines. Intuitivement, un algorithme est la description detaillée d'une suite d'opérations permettant d'obtenir un résultat à partir de données. Cette définition est néanmoins trop vague pour permettre l'étude précise de la notion de calculabilité.

Une théorie de la calculabilité a été développée en ce qui concerne les ensembles discrets ; elle semble aujourd'hui acceptée par tous. La notion d'algorithme a été définie de plusieurs façons différentes : fonctions récursives, machines de Turing, algorithmes de Markov, etc. Pour chacun de ces modèles, tout le monde semble d'accord pour dire qu'ils reflètent bien la notion intuitive de calcul automatique. C'est ce qu'expriment la *thèse de Church* (pour les fonctions récursives), la *thèse de Turing* (pour les machines de Turing), et le *principe de normalisation* (pour les algorithmes de Markov). Heureusement, il est facile de montrer l'équivalence de ces différents modèles.

En ce qui concerne les ensembles non dénombrables, et en particulier l'ensemble $\mathbb{R}$ des nombres réels, plusieurs théories ont été proposées depuis la première définition des réels *calculables* par Turing en 1936 (voir [Tur 36] et [Tur 37]), mais aucune n'a encore été acceptée comme la théorie correspondant réellement à la notion intuitive de calculabilité en analyse. Chacune d'elles permet d'avoir une vision constructive de l'analyse : de nombreux théorèmes classiques de l'analyse ont une version constructive.

La difficulté pour trouver un bon modèle semble venir du fait qu'il n'y a pas qu'une seule notion d'algorithme. Il est important de percevoir la différence entre les notions de problème *calculable algorithmiquement* et *calculable par une machine*. S'il est généralement admis que tout ce qui est calculable par une machine est calculable algorithmiquement, la réciproque dépend beaucoup de la définition de la machine. Le terme *machine*, comme le terme *algorithme*, fait appel à une notion intuitive difficilement formalisable. On doit donc essayer de trouver une généralisation à la thèse de Church pour fixer les rapports entre ces deux notions.

Si un travail peut être effectué algorithmiquement, cela ne veut pas dire qu'il existe une machine « physique » capable d'effectuer ce travail. Par exemple, nos ordinateurs actuels travaillent sur des données discrètes, mais on peut considérer d'autres modèles de machines, opérant sur des objets différents, comme des nombres réels.

Mon but est de présenter un document de Klaus Weihrauch intitulé *A simple Introduction to computable analysis* [Wei 95], qui propose une théorie de la calculabilité sur les réels. J'essaierai ensuite de donner un aperçu des autres approches possibles en présentant deux autres théories, et en les comparant à celle de Weihrauch. J'ai choisi de m'attarder surtout sur les problèmes liés au choix du modèle, notamment le choix des représentations des réels en mémoire, plutôt que sur les propriétés engendrées par ces choix – certaines seront néanmoins abordées car elles permettent de valider les modèles.

La « théorie de l'effectivité de type 2 »[1], théorie décrite par Weihrauch dans l'article présenté ici, ne suppose pas les machines capables de travailler directement sur des nombres réels, ce qui correspond à la réalité physique. Elle étend la théorie classique de la calculabilité par l'intermédiaire de *représentations*. La notion de fonction réelle calculable qui en découle correspond à une définition donnée par le logicien polonais A. Grzegorczyk en 1955 ([Grz 55] et [Grz 57]). Cette définition est inspirée de la définition des fonctions récursives de Church-Rosser. Grzegorczyck commence par définir la notion de *fonctionnelle calculable*, comme étant le plus petit ensemble contenant l'identité, les fonctionnelles constantes, et clos par des opérations de substitution, d'identification, et de « minimum effectif ». Il définit les fonctions calculables à partir de cette notion, et en donne plusieurs autres définitions équivalentes dans [Grz 57].

Le chapitre 1 présente les principales définitions données par Weihrauch pour pouvoir parler de calcul sur des réels. Le chapitre 2 montre comment on peut faire le lien entre calculabilité et topologie. Un modèle sera correct si ce lien est naturel. Cela dépend beaucoup du choix de la repésentation des réels, dont on discutera dans le chapitre 3.

La théorie basée sur la définition de Grzegorczyk est appelée l'*approche polonaise*. L'*approche russe*, introduite par Ceitin en 1959, est une autre façon de généraliser la calculabilité classique à l'analyse. Elle considère uniquement les nombres réels calculables (voir chapitre 4). Ce chapitre sera également l'occasion de montrer certaines propriétés de l'ensemble des réels calculables.

A l'opposé de ces deux approches, le modèle « real-RAM » (*real random access machine*, ou machine réelle à accès aléatoire) ne se préoccupe pas de la représentation des réels. Les machines qu'il considère savent mémoriser et effectuer des opérations simples sur les réels. Bien que très éloignée des mo-

1. Type 2 Theory of Effectivity

dèles réalisables physiquement, l'approche est légitime, car les approximations de réels utilisées par les ordinateurs, comme la virgule flottante, permettent de s'en approcher dans un bon nombre de cas. Elle permet par exemple de définir une notion de sous-ensembles de $\mathbb{R}$ *récursivement énumérables* proche de celle de la calculabilité classique. Par exemple, le complémentaire de l'ensemble de Cantor est récursivement énumérable car il existe un algorithme qui s'arrête sur ses points et uniquement sur ceux-là (voir chapitre 5).

Notations

On notera Σ un alphabet fini comportant tous les symboles dont on a besoin. Ses éléments seront appelés *lettres* ou *caractères*. Σ^* est l'ensemble des mots (suites finies de lettres) sur Σ. ε représente le mot vide, $\Sigma^+ = \Sigma^* \backslash \{\varepsilon\}$. $\Sigma^{\mathbb{N}}$ est l'ensemble des suites infinies de lettres. Les mots et les suites de lettres seront notés de manière intuitive ; par exemple, si $w \in \Sigma^*$, $01^3w(01)^{\mathbb{N}}$ représentera la suite constituée des lettres 0, 1, 1, 1, puis des lettres de w, puis du mot 01 répété indéfiniment. La notation $f :\subseteq A \longrightarrow B$ sera utilisée pour une fonction de domaine inclus dans A à valeurs dans B.

Chapitre 1

La notion de calculabilité en analyse

Nous allons devoir représenter des nombres réels grâce aux lettres d'un alphabet Σ. L'ensemble Σ^* étant dénombrable, on ne peut l'utiliser pour une notation de $\mathbb{R}$. On va donc être obligés de travailler sur des suites infinies. Cela nécessite une modification de la définition des machines de Turing. On verra ensuite comment on peut passer de Σ^* ou $\Sigma^{\mathbb{N}}$ à d'autres ensembles comme $\mathbb{R}$, en associant à chaque élément un nom. Enfin on généralisera les notions d'ensembles *récursifs* et *récursivement énumérables.*

1.1 Extension de la calculabilité aux suites infinies

Pour pouvoir utiliser des suites de lettres infinies, on définit un nouveau type de machines de Turing de la manière suivante :

Définition 1.1.1 *Une* ***machine de Turing de type 2*** *est une machine déterministe M avec k rubans d'entrée ($k \in \mathbb{N}$), numérotés de 1 à k, un ruban de sortie, de numéro 0, et un nombre fini de rubans supplémentaires, dits rubans de travail, telle que :*

- *Les têtes de lecture ne peuvent avancer que dans un sens sur les rubans d'entrée et de sortie,*
- *On a une spécification de type de la forme $(Y_1, ..., Y_k, Y_0)$, où $Y_1, ..., Y_k, Y_0 \in \{\Sigma^*, \Sigma^{\mathbb{N}}\}$. Si $Y_i = \Sigma^*$, le ruban i servira aux inscriptions finies, si $Y_i = \Sigma^{\mathbb{N}}$, le ruban i servira pour des inscriptions infinies.*

Remarque 1.1.2 *Dans toute la suite,* $\{Y, Y', Y_0, ..., Y_k\} \subseteq \{\Sigma^*, \Sigma^{\mathbb{N}}\}$*, et* M*,* M'*,* $M_0, ..., M_k$ *sont des ensembles quelconques.*

La machine écrit, lettre par lettre, des données sur le ruban de sortie, à partir des données lues en entrée. Elle peut évidemment ne pas lire les données en entrée entièrement. Bien sûr, si $Y_0 = \Sigma^{\mathbb{N}}$, la machine ne s'arrête jamais, mais la tête de lecture du ruban de sortie ne peut avancer que dans un sens, ce qui permet de pouvoir commencer à traiter le début du résultat même si la machine ne s'arrête pas. Pour les rubans d'entrée – et le ruban de sortie dans le cas de Σ^* – on aurait pu autoriser le retour en arrière des têtes, ce qui aurait donné un modèle exactement équivalent.

On peut considérer qu'une machine de type 2 calcule une fonction de la manière suivante :

Définition 1.1.3 *La fonction* f_M *calculée par la machine* M *est la fonction d'un sous-ensemble de* $Y_1 \times ... \times Y_k$ *dans* Y_0 *définie par :*

- *si* $Y_0 = \Sigma^*$*,* $f_M(y_1, ..., y_k) = w \Longleftrightarrow M$*, sur l'entrée* $(y_1, ..., y_k)$*, s'arrête, avec le mot* $w \in \Sigma^*$ *sur le ruban de sortie,*
- *si* $Y_0 = \Sigma^{\mathbb{N}}$*,* $f_M(y_1, ..., y_k) = p \Longleftrightarrow M$*, sur l'entrée* $(y_1, ..., y_k)$*, écrit la suite* $p \in \Sigma^{\mathbb{N}}$ *sur le ruban de sortie, chaque lettre de* p *étant atteinte au bout d'un temps fini.*

Définition 1.1.4 *Une fonction* $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ *est dite* ***calculable*** *s'il existe une machine* M *de type 2 qui calcule cette fonction.*

Une suite p *de* $\Sigma^{\mathbb{N}}$ *est un élément* ***calculable*** de Y_0 *si et seulement si elle l'est quand on la considère comme une fonction 0-aire.*

On peut énoncer à partir de ces définitions une généralisation de la thèse de Church/Turing, pour affirmer que ce modèle correspond bien à l'idée de calcul par une machine :

Thèse 1.1.5 (Church/Turing) *Une fonction* $f_M :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ *est calculable physiquement par une machine, si et seulement si elle peut être calculée par une machine de type 2.*

Bien sûr, cette affirmation n'est pas démontrable. Cependant c'est sur elle que repose toute la théorie. Elle semble néanmoins correspondre très bien à la notion de calcul induite par nos ordinateurs actuels ; Weihrauch estime même qu'il est évident que c'est vraiment la seule notion intuitive de calcul.

1.2 Calculabilité sur d'autres ensembles

1.2.1 Systèmes de noms

La théorie de l'effectivité de type 2 fait donc l'hypothèse que les machines ne savent travailler que sur des suites de symboles. Pour transférer la notion de calculabilité à d'autres ensembles – comme $\mathbb{R}$ – il va être nécessaire de passer par l'intermédiaire de *systèmes de noms* :

Définition 1.2.1 *Une **notation** d'un ensemble M est une application surjective $\nu :\subseteq \Sigma^* \longrightarrow M$.*

*Une **représentation** d'un ensemble M est une application surjective $\delta :\subseteq \Sigma^{\mathbb{N}} \longrightarrow M$.*

*Un **système de noms** est une notation ou une représentation.*

Remarque 1.2.2 *Dans toute la suite, γ_i (pour $i \in \{0, ..., k\}$), γ et γ', sont des systèmes de noms. $\gamma_i :\subseteq Y_i \longrightarrow M_i$, $\gamma :\subseteq Y \longrightarrow M$, $\gamma' :\subseteq Y' \longrightarrow M'$.*

L'alphabet Σ va donc nous servir à donner un « nom » aux éléments de l'ensemble sur lequel on veut travailler. La représentation des réels la plus courante est la représentation décimale, que l'on notera $\delta_{dec} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \mathbb{R}$.

On utilisera souvent pour $\mathbb{N}$ la notation binaire $\nu_{bin} :\subseteq \Sigma^* \longrightarrow \mathbb{N}$ (par exemple ν_{bin}(« 10110 »)$= 22$), et pour $\mathbb{Q}$, la notation $\nu_{\mathbb{Q}} :\subseteq \Sigma^* \longrightarrow \mathbb{Q}$ (par exemple : $\nu_{\mathbb{Q}}$(« -110/101 »)$= -6/5$). (On peut les définir précisément très facilement de plusieurs manières équivalentes.)

Remarque 1.2.3 *Pour alléger les notations, on notera souvent $\overline{u}$ au lieu de $\nu_{\mathbb{Q}}(u)$.*

On peut définir maintenant la notion de calculabilité sur d'autres ensembles que Σ^* ou $\Sigma^{\mathbb{N}}$, par l'intermédiaire de systèmes de noms.

Définition 1.2.4 *Un élément x de M est dit γ-calculable s'il existe un élément calculable y de Y tel que $\gamma(y) = x$.*

Une fonction $F :\subseteq M_1 \times ... \times M_k \longrightarrow M_0$ est dite $(\gamma_1, ..., \gamma_k, \gamma_0)$-calculable s'il existe une fonction calculable $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ telle que $F(\gamma_1(y_1), ..., \gamma_k(y_k)) = \gamma_0(f(y_1, ..., y_k))$, partout où $F(\gamma_1(y_1), ..., \gamma_k(y_k))$ existe.

On dispose maintenant des outils nécessaire pour pouvoir dire qu'un réel ou une fonction réelle sont calculables par une machine.

En calculabilité classique, le choix de la notation de $\mathbb{N}$ est souvent implicite (il s'agit par exemple de la notation binaire ou unaire), et n'a pas trop d'importance. Ici, on va voir que toutes les représentations ne sont pas équivalentes. Il va donc falloir faire attention à ce choix. La représentation devra contenir assez d'informations sur les nombres pour être facilement utilisable par la machine, mais ne doit pas demander d'informations que la machine ne peut pas calculer, ce qui interdirait des calculs simples.

1.2.2 Choix d'une représentation

Il faut choisir une représentation adéquate pour avoir une notion « correcte » de calculabilité sur $\mathbb{R}$. Malheureusement, δ_{dec} ne peut pas être utilisée, car des fonctions très simple sur $\mathbb{R}$ ne sont pas δ_{dec}-calculables. L'exemple donné par Weihrauch est le suivant : soit f la fonction définie sur $\mathbb{R}$ par $f(x) = 3x$. f n'est pas δ_{dec}-calculable. Pour le prouver, il suffit de considérer le nombre $x = 1/3$, représenté par la suite $p = 0,3^{\mathbb{N}}$ $(= 0,3333...)$. Supposons la fonction f δ_{dec}-calculable. Il existe un machine M qui la calcule. On a donc $f_M(p) = 0,9^{\mathbb{N}}$ ou $f_M(p) = 1,0^{\mathbb{N}}$. Considérons d'abord le premier cas. Après un certain nombre n de pas de calcul, M a écrit la chaîne « 0, » sur le ruban de sortie. Pour cela, elle a lu au plus les n premiers caractères de l'entrée. Considérons maintenant en entrée la chaîne $q = 0,3^n40^{\mathbb{N}}$. Pendant les n premiers pas de calcul, M va écrire « 0, » sur le ruban de sortie, alors que le résultat doit commencer par « 1, ». Le deuxième cas est similaire.

En fait, la représentation décimale est trop rigide, car elle contient « trop d'information » ; par exemple, le fait de lire « 1, » sur le ruban de sortie nous indique déjà que le résultat sera dans l'intervalle $[1,2]$, information que la machine ne peut connaître *a priori* dans le cas de la multiplication de $0,3^{\mathbb{N}}$ par 3. D'où l'idée de proposer un classement des systèmes de noms selon la « quantité d'information » qu'ils comportent. Pour deux systèmes de noms γ et γ', on dira que les γ-noms contiennent plus d'information que les γ'-noms s'il existe une machine capable de transformer les γ-noms en γ'-noms, mais pas le contraire. En terme de réduction (définie ci-dessous), $\gamma \leq \gamma'$ mais $\gamma' \not\leq \gamma$.

Définition 1.2.5 *On dit que γ est **réductible** à γ' ($\gamma \leq \gamma'$), si et seulement si il existe une fonction f calculable telle que :*

$$\forall y \in dom(\gamma)\ \gamma(y) = \gamma'(f(y))$$

*Si $\gamma \leq \gamma'$ et $\gamma' \leq \gamma$, on dira que γ et γ' sont **équivalents** ($\gamma \equiv \gamma'$).*

$\leq$ est une relation de préordre partielle. $\equiv$ est une relation d'équivalence.

Il faut maintenant trouver un système de noms correct pour $\mathbb{R}$.

Définition 1.2.6 *La **représentation de Cauchy** est l'application $\rho_C :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \mathbb{R}$ définie par $\rho_C(p) = x$ si et seulement si $p = u_0\#u_1\#...$ (où $\# \in \Sigma$), avec $\forall i\ u_i \in dom(\nu_{\mathbb{Q}})$, $\forall k, k < i \Longrightarrow |\overline{u_i} - \overline{u_k}| < 2^{-k}$ et $x = \lim \overline{u_k}$.*

Proposition 1.2.7 *On a la relation : $\delta_{dec} \leq \rho_C$. (démonstration p 20 de [Wei 95])*

Les rationnels sont évidemment ρ_C-calculables. Weihrauch montre aussi que par exemple $\sqrt{2}$ et $\log_3(5)$ sont ρ_C-calculables, et que les fonctions réelles

$(x, y) \mapsto x + y$, $(x, y) \mapsto x.y$, $(x, y) \mapsto \max(x, y)$, et $x \mapsto 1/x$ sont (ρ_C, ρ_C, ρ_C)-calculables (ou (ρ_C, ρ_C)-calculables). Un résultat important affirme qu'une série entière associée à une suite réelle calculable est calculable sur tout disque de centre O strictement inclus dans son disque de convergence (voir p25 de [Wei 95]). Enfin, l'ensemble des fonctions réelles calculables est clos par composition. De tout ceci, on peut déduire la calculabilité de nombreuses fonctions (e^x, $\sin x$,...) ainsi que de nombreux réels (π, e,...).

ρ_C semble donc plus appropriée que δ_{dec} pour étendre la calculabilité aux réels. Ce choix ne semble pas mauvais car on verra qu'il entraîne des propriétés topologiques naturelles. Remarquons que l'on impose une convergence rapide de la suite $(\overline{u_i})$ dans la définition de ρ_C. Une tentative de justification de ce choix sera présentée au chapitre 3.

1.3 Notions calculabilistes sur les ensembles de réels

Comme en calculabilité classique, on peut définir des notions d'ensembles *récursifs* et *récursivement énumérables* :

Définition 1.3.1 *Soit $X \subseteq W \subseteq Y_1 \times ... \times Y_k$. X est dit **récursivement énumérable (r.e.) dans** W s'il existe une fonction $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow \Sigma^*$ calculable telle que $X = W \cap dom(f)$.*

*X est dit **récursif** dans W s'il existe une fonction $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow \Sigma^*$ calculable telle que $f(x) = 0$ si $x \in X$ et $f(x) = 1$ si $x \in W \backslash X$.*

*Si $X = W$, X est dit **récursivement énumérable (r.e.) ou récursif**.*

Ces définitions ressemblent beaucoup à celles de la calculabilité classique. Remarquons que les fonctions qu'elles utilisent sont à valeurs dans Σ^*, donc la notion d'arrêt de la machine peut être utilisée. Cette définition d'ensemble récursif est équivalente à celle donnée dans [Wei 95] : X est dit **récursif** dans W si X et $W \backslash X$ sont r.e. (Pour l'une des implications, il suffit de simuler parallèlement les deux machines en effectuant alternativement la recopie d'un caractère de chaque ruban d'entrée sur des rubans auxiliaires, avec replacement des têtes, et un pas de chacune des machines sur ces rubans auxiliaires. On s'arrête dès que l'une des deux s'arrête, après avoir répondu 0 ou 1. L'autre implication est évidente.)

On ne peut pas, comme en calculabilité classique, caractériser les ensembles r.e. comme les images de fonctions récursives totales. Les parties r.e. de $\Sigma^{\mathbb{N}}$ ne sont pas *énumérables* dans le sens intuitif (de toute façon, elles ne sont pas toujours dénombrables). Même en ne considérant que les ensembles d'éléments calculables de $\Sigma^{\mathbb{N}}$, l'équivalence semble être fausse : on peut énumérer effectivement un ensemble de réels calculables défini comme le domaine d'une fonction calculable, mais la réciproque a peu de chances d'être vraie.

Encore une fois, ces définitions pour les suites de lettres peuvent être étendues de $X \subseteq Y_1 \times ... \times Y_k$ à un sous-ensemble de $M_1 \times ... \times M_k$:

Définition 1.3.2 *$X \subseteq M_1 \times ... \times M_k$ est dit $(\gamma_1, ..., \gamma_k)$-r.e. (récursif) si $\{(y_1, ..., y_k) \in Y_1 \times ... \times Y_k / (\gamma_1(y_1), ..., \gamma_k(y_k)) \in X\}$ est r.e. (récursif) dans $dom(\gamma_1) \times ... \times dom(\gamma_k)$.*

Il est possible de caractériser ces ensembles grâce à la proposition suivante :

Proposition 1.3.3 *Soit $X \subseteq \mathbb{R}$.*
X est ρ_C-r.e. $\Longleftrightarrow \exists Y \subseteq dom(\nu_{\mathbb{Q}})^2$ (Y r.e. et $X = \bigcup \{]\overline{u}, \overline{v}[, (u,v) \in Y\}$
X est ρ_C-récursif $\Longleftrightarrow X = \emptyset$ ou $X = \mathbb{R}$.

Il n'existe donc aucun sous-ensemble ρ_C-récursif de $\mathbb{R}$ non trivial ! Des exemples d'ensembles ρ_C-récursivement énumérables sont $\{x/x > a\}$, $\{(x,y)/x < y\}$,... avec a ρ_C-calculable. Leurs complémentaires ne le sont pas, sinon ils seraient ρ_C-récursifs.

On verra par la suite que ces notions sont très liées à certaines notions topologiques, via des représentations correctes.

Chapitre 2

Notions topologiques

Les notions introduites au chapitre précédent sont très liées à des notions topologiques sur $\mathbb{R}$. On verra notamment que les propriétés topologiques sont souvent conservées en passant aux représentations par l'intermédiaire de ρ_C. Le théorème le plus important de la théorie est sans-doute celui qui affirme que toute fonction réelle calculable est continue. A la fin du chapitre, on verra comment on peut faire de l'analyse constructive avec des objets plus complexes que les réels : ouverts, fermés, compacts, fonctions,...

2.1 Continuité

Pour permettre le travail sur des suites infinies, on a été obligé de limiter le déplacement de la tête de lecture/écriture du ruban de sortie à un seul sens. De cette caractéristique découle une propriété très importante des fonctions calculables sur des ensembles comme Σ^* et $\Sigma^{\mathbb{N}}$:

Proposition 2.1.1 (Propriété de finitude) *Soit $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ calculable. Si $f(x) = y$, toute portion finie de y ne dépend que d'un préfixe fini de x.*

En fait, il s'agit juste de l'expression de la continuité pour la topologie discrète $\tau_d = \{A/A \subseteq \Sigma^*\}$ (sur Σ^*) et la topologie de Cauchy $\tau_C = \{A\Sigma^{\mathbb{N}}/A \subseteq \Sigma^*\}$ (sur $\Sigma^{\mathbb{N}}$). Ces topologies sont métrisables. On utilisera sur $\Sigma^{\mathbb{N}}$ la distance définie par :

$$\forall p, q \in \Sigma^{\mathbb{N}} \; d(p,q) = \begin{cases} 0 & \text{si } p = q \\ 2^{-n} & \text{si } p \neq q \text{, où } n \text{ est la longueur du plus} \\ & \text{long préfixe commun à } p \text{ et } q. \end{cases}$$

En terme de continuité, la propriété de finitude devient :

Théorème 2.1.2 *Toute fonction $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ calculable est continue.*

Voyons maintenant comment cette propriété se traduit pour d'autres ensembles que ceux des suites de lettres, en particulier pour $\mathbb{R}$. Il faut relier les notions de continuité entre elles.

Définition 2.1.3 *Une fonction* $F :\subseteq M_1\times...\times M_k \longrightarrow M_0$ *est dite* $(\gamma_1, ..., \gamma_k, \gamma_0)$*-continue s'il existe une fonction continue* $f :\subseteq Y_1 \times ... \times Y_k \longrightarrow Y_0$ *telle que* $F(\gamma_1(y_1), ..., \gamma_k(y_k)) = \gamma_0(f(y_1, ..., y_k))$*, partout où* $F(\gamma_1(y_1), ..., \gamma_k(y_k))$ *existe.*

Le principal théorème sur la continuité est le suivant :

Théorème 2.1.4 *Soit* $f :\subseteq \mathbb{R}^k \longrightarrow \mathbb{R}$*.* f *est continue si et seulement si elle est* $(\rho_C, ..., \rho_C)$*-continue.*

Le théorème 2.1.2 reste donc valable quand on étend la calculabilité aux fonctions réelles par l'intermédiaire de ρ_C :

Théorème 2.1.5 *Toute fonction réelle* (ρ_C, ρ_C)*-calculable est continue.*

Cette propriété limite considérablement le pouvoir de calcul des machines. Elle peut paraître surprenante à première vue, mais elle correspond cependant bien à la réalité puisque jusqu'à présent, aucune fonction réelle discontinue définie sur un intervalle n'a été implantée correctement, d'après Weihrauch, sur une machine physique. Cela montre bien qu'une fonction peut être « facilement définissable », mais pas calculable. On verra une généralisation du théorème précédent page 20.

2.2 Autres définitions

La continuité est la notion topologique qui correspond à celle de calculabilité. De même, on définit les **ensembles** $(\gamma_1, ..., \gamma_k)$**-ouverts** en remplaçant *r.e.* par *ouvert* dans la définition 1.3.2. On a les résultats suivants :

Théorème 2.2.1

1. *Pour* $X \subseteq \mathbb{R}$*, on a l'équivalence : X est* ρ_C*-ouvert* $\Longleftrightarrow$ *X est ouvert.*
2. *Tout ensemble* ρ_C*-r.e. est ouvert.*

Le premier montre que ρ_C transporte bien la notion d'ouvert. Le deuxième peut servir à prouver que des ensembles ne sont pas r.e.

On définit également la notion de **réductibilité (équivalence) topologique** (ou *t*-**équivalence**) en remplaçant le mot *calculable* par *continue* dans la définition 1.2.5. On notera $\gamma \leq_t \gamma'$ et $\gamma \equiv_t \gamma'$. Puisqu'une fonction calculable est continue, on a :

$$\gamma \leq \gamma' \Longrightarrow \gamma \leq_t \gamma'$$

On peut montrer que :

$$\rho_C \not\leq_t \delta_{dec}$$

(donc $\rho_C \not\leq \delta_{dec}$)

2.3 Représentation d'autres objets topologiques

Beaucoup de définitions données jusqu'à présent concernent non seulement $\mathbb{R}$, mais aussi n'importe quel ensemble. Cela va nous permettre de travailler avec des machines sur des objets variés. Les systèmes de noms permettent de travailler sur des ensembles dont le cardinal est inférieur ou égal à celui de $\Sigma^{\mathbb{N}}$. Il n'est donc pas possible de donner une représentation pour l'ensemble des sous-ensembles de $\mathbb{R}$. Cependant, l'ensemble des ouverts, l'ensemble des fermés et l'ensemble des compacts de $\mathbb{R}$ admettent des représentations. On se contentera dans cette section d'énoncer les définitions de quelques représentations, ainsi que quelques théorèmes permettant de relier les notions entre elles, afin de montrer l'utilité de ces représentations et l'opportunité de leur choix. On supposera que l'alphabet Σ contient les caractères $\#$ et $\diamond$, qui seront utilisés comme séparateurs.

Définition 2.3.1 (Représentation des ouverts) *Soit $\tau_{\mathbb{R}}$ la topologie usuelle sur $\mathbb{R}$, et $\delta_{ouv} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \tau_{\mathbb{R}}$ définie par :*
$\delta_{ouv}(p) = \bigcup\{]\overline{u}, \overline{v}[\ /\ u, v \in dom(\nu_{\mathbb{Q}}) \text{ et } \#u \diamond v\# \text{ sous-mot de } p\}$

Définition 2.3.2 (Représentation des fermés) *Soit $\delta_{fer} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \{$ fermés de $\mathbb{R}$ $\}$ définie par :*
$\delta_{fer}(p) = \mathbb{R} \backslash \delta_{ouv}(p)$

Définition 2.3.3 (Représentations des compacts) *Soit K l'ensemble des compacts de $\mathbb{R}$. Définissons d'abord une notation des ensembles finis d'intervalles ouverts à bornes rationnelles par :*

$$\begin{array}{ll} \iota(w) = \Omega & \text{si et seulement si } w = u_1\#v_1\#...u_k\#v_k, \\ & \text{avec } u_1, ..., u_k, v_1, ...v_k \in dom(\nu_{\mathbb{Q}}), \\ & \text{et } \Omega = \{]\overline{u_1}, \overline{v_1}[, ...,]\overline{u_k}, \overline{v_k}[\}. \end{array}$$

On a plusieurs représentations possibles de K, correspondant à plusieurs définitions possibles des compacts de $\mathbb{R}$. En voici deux ; Weihrauch en donne plusieurs autres dans [Wei 96] :

Fermés bornés : *Soit $\delta_{fb} \subseteq \Sigma^{\mathbb{N}} \longrightarrow K$ définie par :*

$$\begin{array}{ll} \delta_{fb}(p) = X & \text{si et seulement si } p = u\#q, \\ & \text{avec } u \in dom(\nu_{\mathbb{Q}}),\ q \in dom(\delta_{fer}),\ X = \delta_{fer}(q), \\ & \text{et } X \subseteq [-\overline{u}, \overline{u}] \end{array}$$

Borel - Lebesgue : *Soit $\delta_{bl} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow K$ définie par :*

$$\begin{array}{ll} \delta_{bl}(p) = X & \text{si et seulement si } p = w_0 \diamond w_1 \diamond ..., \text{ avec } w_0, w_1, ... \in dom(\iota), \\ & \text{et } \forall w \in dom(\iota)\ X \subseteq \bigcup \iota(w) \Longleftrightarrow \exists i\ w = w_i \end{array}$$

Soit $C(X)$ l'ensemble des fonctions continues de domaine $X \subseteq \mathbb{R}$ et à valeurs dans $\mathbb{R}$. On peut donner les représentations suivantes de $C(\mathbb{R})$ et $C([0,1])$:

Définition 2.3.4 (Représentation de $C(\mathbb{R})$) *Soit* $\delta_{C(\mathbb{R})} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow C(\mathbb{R})$ *définie par :*

$\delta_{C(\mathbb{R})}(p) = f$ *si et seulement si* $p = u_0\#v_0\#x_0\#y_0 \diamond u_1\#v_1\#x_1\#y_1 \diamond ...$
avec $\forall i \in \mathbb{N}\ u_i, v_i, x_i, y_i \in dom(\nu_{\mathbb{Q}})$, *et* $f[a,b] \subseteq]c,d[\Longleftrightarrow$
$\exists i\ (a = \overline{u_i}, b = \overline{v_i}, c = \overline{x_i}, d = \overline{y_i})$

Définition 2.3.5 (Représentation de Cauchy de $C([0,1])$)

- *On commence par définir une notation* $\alpha :\subseteq \Sigma^* \longrightarrow A$ *de l'ensemble* A *des fonctions définies sur* $[0,1]$ *continues et affines par morceaux « à sommets rationnels » par :*
  $\alpha(w) = f$ *si et seulement si* $w = u_0\#v_0\#...u_k\#v_k$,
  avec $\forall i \in \{0,...,k\}\ u_i, v_i, x_i, y_i \in dom(\nu_{\mathbb{Q}})$,
  $0 = \overline{u_0} < ... < \overline{u_k} = 1$, *et* f *est la fonction affine sur chacun des intervalles* $[\overline{u_i}, \overline{u_{i+1}}]$ *telle que*
  $\forall i \in \{0,...,k\} f(\overline{u_i}) = \overline{v_i}$

- *Soit* $\delta_{C([0,1])} :\subseteq \Sigma^{\mathbb{N}} \longrightarrow C([0,1])$ *définie par :*
  $\delta_{C([0,1])}(p) = f$ *si et seulement si* $p = w_0 \diamond w_1 \diamond ...$ *avec* $\forall i \in \mathbb{N}\ w_i \in dom(\alpha)$,
  et $\forall k \forall i > k\ d_\infty(\alpha(w_i), \alpha(w_k)) < 2^{-k}$
  et $\forall k\ d_\infty(f, \alpha(w_k)) \leq 2^{-k}$,
  d_∞ *étant la distance sur* $C([0,1])$ *définie par :*
  $\forall f, g \in C([0,1])\ d_\infty(f,g) = \sup_{x \in [0,1]} |f(x) - g(x)|$.

Les exemples de représentations donnés ci-dessus permettent de donner une version effective d'un certain nombre de théorèmes de l'analyse. On peut aussi énoncer des théorèmes reliant les différentes notions définies entre elles. (voir [Wei 95]) Nous nous contenterons ici d'en mentionner quelques-uns.

Théorème 2.3.6 (Quelques propriétés des représentations définies ci-dessus)

- X *est* δ_{ouv}*-calculable* $\Longleftrightarrow X$ *est* ρ_C*-r.e.*
- $f :\subseteq \mathbb{R} \longrightarrow \mathbb{R}$ *est* (ρ_C, ρ_C)*-calculable* $\Longleftrightarrow f$ *est* $\delta_{C(\mathbb{R})}$*-calculable.*
- *L'union et l'intersection sont* $(\delta_{ouv}, \delta_{ouv}, \delta_{ouv})$*-calculables.*
- *L'union et l'intersection sont* $(\delta_{bl}, \delta_{bl}, \delta_{bl})$*-calculables.*
- *Les représentations* δ_{fb} *et* δ_{bl} *sont équivalentes.* (version effective du théorème de Heine-Borel)
- *La fonction*

$$\begin{array}{rcl} Max : C([0,1]) & \longrightarrow & \mathbb{R} \\ f & \longrightarrow & \max_{x \in [0,1]} f(x) \end{array}$$

  est $(\delta_{C(\mathbb{R})}, \rho_C)$*-calculable.*

Weihrauch montre également que la dérivation n'est pas effectivement réalisable alors que l'intégration l'est. Il s'intéresse enfin à la détermination des zéros d'une fonction réelle continue, et montre qu'il n'est pas possible dans le cas général de trouver les zéros d'une fonction représentée grâce à $\delta_{C(\mathbb{R})}$. Cependant, la recherche des zéros est possible si l'on se place sur l'ensemble des fonctions de $C([0,1])$ qui ont un zéro et un seul. Il est également possible, étant donnés une fonction f de $C([0,1])$ et un entier n, de trouver une valeur de x – si elle existe – telle que $f(x) < 2^{-n}$.

Toutes ces propriétés montrent que le choix fait pour les représentations n'est pas mauvais. Nous allons essayer d'expliquer ces choix dans le chapitre suivant.

Chapitre 3

Choix des représentations

La principale difficulté pour généraliser la calculabilité sur $\Sigma^{\mathbb{N}}$ à d'autres ensembles est le choix de la représentation. Les propriétés que l'on va pouvoir déduire dépendent beaucoup de ce choix. Il faut trouver une représentation qui contienne les informations dont on a besoin en entrée pour faire un calcul, et qui puisse contenir en sortie les informations produites par la machine. En fait, ces informations dépendent évidemment de ce que l'on veut faire.

3.1 Représentation standard

Il est possible de formaliser cette notion en définissant des représentations à partir d'une *structure d'information*. Il n'y a pas de « bonne » représentation de $\mathbb{R}$, mais pour une structure d'information donnée, on peut trouver la bonne représentation – à équivalence près.

Définition 3.1.1 *Une* ***structure d'information*** *sur un ensemble M est un triplet (M, σ, ν), où :*

- *σ est un ensemble de sous-ensembles de M, qui sépare les points, c'est-à-dire tel que $\cup\sigma = M$ et $\forall x, y \in M\ \{A \in \sigma / x \in A\} = \{A \in \sigma / y \in A\} \Longrightarrow x = y$*
- *ν est une notation de σ (avec $dom(\nu) \subseteq (\Sigma \backslash \{\#\})^+$).*

Par exemple, pour $M = \mathbb{R}$, on peut considérer la structure d'information définie par $x \in \nu(w) \Leftrightarrow x > \overline{w}$. Un réel x sera défini de manière unique par la donnée de tous les ensembles de type $]u, +\infty[$ (avec u rationnel) qui le contiennent ; c'est la borne inférieure de leur intersection. Cette structure contient donc bien toutes les informations nécessaires pour définir x.

Une fois la structure d'information de M choisie, on peut définir la *représentation standard* de M qui lui correspond : un nom de $x \in M$ dans la représentation standard contient l'énumération de toutes ses *propriétés élémentaires*, c'est-à-dire de tous les éléments de σ qui contiennent x :

Définition 3.1.2 *La **représentation standard** pour une structure d'information (M, σ, ν) est l'application $\delta_\nu :\subseteq \Sigma^{\mathbb{N}} \longrightarrow M$ définie par :*
$\delta_\nu(p) = x \iff \{w/x \in \nu(w)\} = \{w/\#w\#$ *est un sous-mot de* $p\}$.

Si l'on connaît la structure d'information que l'on veut utiliser, les meilleures représentations seront la représentation standard et les représentations qui lui sont équivalentes.

3.2 Représentations admissibles

A chaque structure d'information correspond en fait une topologie sur M, dont σ est une base. Par exemple, la topologie associée à l'exemple de structure d'information donné ci-dessus est $\{]x, +\infty[, x \in \overline{\mathbb{R}}\}$.

La représentation standard associée à la topologie classique $\tau_{\mathbb{R}}$ sur $\mathbb{R}$ correspond bien à l'idée courante d'approximation d'un nombre. On peut définir la structure d'information $(\mathbb{R}, \sigma, \nu)$ qui lui correspond par $x \in \sigma(w) \Leftrightarrow (w = u \diamond v$ et $\overline{u} < x < \overline{v})$. Il est donc normal de chercher à utiliser une représentation qui lui est équivalente.

Théorème 3.2.1 *La représentation standard δ_ν associée à la topologie $\tau_{\mathbb{R}}$ est équivalente à ρ_C.*

démonstration : $\star$ Soit M la machine définie par l'algorithme suivant. M reçoit en entrée une suite $p \in dom(\delta_\nu)$. A l'étape i, M lit la suite p jusqu'à y trouver la notation d'un intervalle de largeur inférieure à 2^{-i}. M écrit ensuite sur le ruban de sortie le mot u_i tel que $\overline{u_i}$ soit le milieu de cet intervalle, suivi d'un $\#$.
Soient $j, k \in \mathbb{N}$; on suppose $k > j$.

$$\begin{aligned} |\overline{u_j} - \overline{u_k}| &\leq |\overline{u_j} - x| + |x - \overline{u_k}| \\ &\leq 2^{-j-1} + 2^{-k-1} \leq 2^{-j} \end{aligned}$$

donc on a bien $\delta_\nu = \rho_C \circ f_M$.
$\star$ Soit I un intervalle ouvert à bornes rationnelles. On sait énumérer tous les mots de la forme $u \diamond v$ avec $u, v \in dom(\nu_{\mathbb{Q}})$ tels que $I \subseteq]\overline{u}, \overline{v}[$ (démonstration triviale).
Considérons donc la machine M suivante. M reçoit en entrée une suite $p \in dom(\rho_C)$ de la forme $u_0\#u_1\#....$ A l'étape 0, M lit $u_0\#$, le recopie sur un ruban auxiliaire, et écrit le premier élément de l'énumération des intervalles qui contiennent $]\overline{u_0} - 2^0, \overline{u_0} + 2^0[$ (voir ci-dessus). A l'étape i, M recopie $u_i\#$ à la

suite des autres sur le ruban auxiliaire, puis lit ce ruban depuis le début ; pour chaque u_j, elle écrit le $(i-j+1)$-ème élément de l'énumération qui correspond à l'intervalle $I =]\overline{u_j} - 2^j, \overline{u_j} + 2^j[$.
Soit $w = u \diamond v \in dom(\nu)$. Si $x \in \nu(w)$ alors il existe i tel que $2^{-i} < min(|\overline{u} - x|, |\overline{v} - x|)$. Pour cette valeur de i, $]\overline{u_i} - 2^{-i}, \overline{u_i} + 2^{-i}[\subseteq]\overline{u}, \overline{v}[$, donc $u \diamond v$ est dans la suite construite par M. Réciproquement, soit $\#w\#$ un sous-mot de la suite construite par M. $w = u \diamond v$. Il existe i tel que $x \in]\overline{u_i} - 2^{-i}, \overline{u_i} + 2^{-i}[\subseteq]\overline{u}, \overline{v}[$, donc $x \in \nu(w)$. □

Définition 3.2.2 *Les représentations t-équivalentes à la représentation standard pour une topologie τ sont dites* ***admissibles*** *pour cette topologie.*

Sur $\mathbb{R}$, ρ_C est admissible avec la topologie usuelle $\tau_{\mathbb{R}}$, ce qui justifie le choix de ρ_C comme représentation pour $\mathbb{R}$.

De même, les représentations données au chapitre 2 des ouverts fermés, compacts et fonctions continues sont admissibles pour les structures d'informations définies ci-dessous :

ouverts : $M = \tau_{\mathbb{R}}, U \in \nu(w) \Leftrightarrow (w = u \diamond v$ et $[\overline{u}, \overline{v}] \subseteq U)$

compacts : $M = K, X \in \nu(w) \Leftrightarrow (w = u_1 \diamond v_1 \diamond ... u_k \diamond v_k$
et $X \subseteq]\overline{u_1}, \overline{v_1}[\cup ... \cup]\overline{u_k}, \overline{v_k}[)$

C($\mathbb{R}$) : $M = C(\mathbb{R}), f \in \nu(w) \Leftrightarrow (w = u \diamond v \diamond x \diamond y$ et $f[\overline{u}, \overline{v}] \subseteq]\overline{x}, \overline{y}[)$

Avec les représentations admissibles, les éléments sont décrits grâce à des propriétés topologiques (appartenance à des ouverts). C'est pour cette raison qu'on les choisit. Ce sont elles qui permettent d'énoncer des résultats comme le théorème 2.1.5, dont voici une généralisation :

Théorème 3.2.3 *Soient, pour $i \in \{0, ..., k\}$, des représentations $\delta_i :\subseteq \Sigma^{\mathbb{N}} \longrightarrow M_i$, admissibles pour les topologies τ_i. Une fonction $f :\subseteq M_1 \times ... \times M_k \longrightarrow M_0$ est continue si et seulement si elle est $(\delta_1, ..., \delta_k, \delta_0)$-continue.*

Ce théorème est un des points fondamentaux de la théorie, ce qui explique le choix des représentations admissibles pour étendre la calculabilité. Cela n'empêche pas que d'autres représentations soient plus appropriées dans certains cas très particuliers. Il n'y a donc pas une seule notion de calculabilité, même à l'intérieure de ce modèle.

Chapitre 4

L'approche russe

Ce chapitre présente une autre approche de la calculabilité en analyse, très proche de celle de Weihrauch. Tout comme celle-ci, elle est réalisable physiquement. La principale différence vient du fait qu'elle ne considère que les nombres réels calculables. Ce chapitre reprend des notions tirées d'un livre de B. A. Kushner [Kus 85].

4.1 Réels calculables

Le modèle de Weihrauch permet de travailler sur tous les réels. S'il correspond bien à ce que les machines peuvent faire, il permet même d'effectuer des opérations qui ne sont pas physiquement réalisables. Pourquoi en effet autoriser le travail sur des réels non-calculables – ceux-ci ne pouvant par définition pas être introduits dans la machine ?

L'idée est donc de représenter les réels non plus par une suite infinie de lettres, mais par le programme qui permet de les calculer. On peut donc travailler avec des machines de Turing classiques (Kushner travaille avec des algorithmes de Markov [1]).

La théorie de Kushner correspond à une vision résolument constructiviste de l'analyse. Les objets classiques de l'analyse (réels, suites,...) sont redéfinis en les limitant à ceux qui sont calculables. Nous les nommerons ici différemment pour éviter toute confusion.

1. Kushner a été élève de Markov.

4.1.1 Définitions préliminaires

Kushner n'utilise pas la notion de système de noms, mais assimile directement les nombres à leur nom. On peut néanmoins se ramener à la notion de nom en définissant des notations appropriées. On notera $\nu_{\mathbb{N}}$, $\nu_{\mathbb{Z}}$ et $\nu_{\mathbb{Q}}$ des notations de $\mathbb{N}$, $\mathbb{Z}$ et $\mathbb{Q}$ respectivement.

Pour coder les réels, on supposera que l'on dispose d'un codage des algorithmes par des mots – que l'on utilise les machines de Turing ou les algorithmes de Markov. On appellera ces mots des *programmes*. On pourra par exemple considérer une énumération $(\varphi_i)_{i\in\mathbb{N}}$ des fonctions partielles partiellement récursives. Un programme sera dans ce cas un $\nu_{\mathbb{N}}$-nom d'un des indices i tels que φ_i calcule la fonction voulue. La notation $\langle\alpha\rangle$ sera utilisée pour signifier le codage d'un algorithme quelconque de calcul d'un objet calculable α (nombre, suite,...). Si α est une suite, on écrira $\alpha(i) = \nu_{\mathbb{Q}}(\langle\alpha\rangle(i))$.

On a besoin de quelques définitions préliminaires [2] :

Définition 4.1.1 *Soit $(u_i)_{i\in\mathbb{N}}$ une suite de réels (ou rationnels). Un* ***module de convergence*** *de la suite u est une suite $(m_i)_{i\in\mathbb{N}}$ d'entiers naturels qui vérifie :*

$$\forall(i,j,n)\in\mathbb{N}^3\ (i,j\geq m_n \Longrightarrow |u_i-u_j|<2^{-n})$$

Le module de convergence permet d'avoir une idée de la manière dont la suite converge, et donc de la précision avec laquelle on approche sa limite. Un calcul sans indication sur la précision du résultat serait sans intérêt.

Définition 4.1.2 *Une suite u de rationnels est dite* ***fondamentale*** *si l'on peut construire un algorithme de calcul d'un module de convergence de u.*

Une suite u de rationnels est dite ***quasi-fondamentale*** *s'il existe un algorithme de calcul d'un module de convergence de u.*

Une suite u de rationnels est dite ***pseudo-fondamentale*** *si*

$$\forall n\exists m\forall(i,j)\ (i,j\geq m \Longrightarrow |u_i-u_j|<2^{-n})$$

(le « $\exists$ » n'étant pas nécessairement pris dans un sens constructif[3].)

Notons la différence entre *suite quasi-fondamentale* et *suite fondamentale* : l'existence d'un algorithme de calcul d'un module de convergence n'implique pas que l'on sache le construire.

2. Ces définitions diffèrent un peu de celle données par Kushner dans la mesure où l'on tient compte ici des suites non calculables.

3. Il s'agit en fait du « $\neg\neg\exists$ » de la logique constructiviste, pour laquelle « $\exists x$ » signifie que l'on dispose d'une méthode effective pour construire x ; « $\neg\neg\exists x$ » indique juste que l'on aboutit à une contradiction en supposant « $\neg\exists x$ ».

4.1.2 Classes de réels

Il faut maintenant définir ce que l'on entend par *nombre réel calculable*. Dans l'approche russe, un nombre réel est calculable – intuitivement – **si une machine peut nous en donner une approximation par un rationnel avec la précision que l'on veut**. L'idée est de donner un algorithme permettant de construire une suite de rationnels qui tend vers le nombre que l'on veut noter. Pour pouvoir donner un approximation avec une précision donnée, il est intéressant de donner en plus une information sur la vitesse de convergence, sous la forme par exemple d'un module de convergence. Les ensembles $\mathbb{R}_{FR}$, $\mathbb{R}_F$, $\mathbb{R}_q$ et $\mathbb{R}_p$ définis ci-dessous correspondent à plusieurs notions possibles de réels calculables. Remarquons que ces ensembles sont définis par leurs notations.

Définition 4.1.3 *Soit $\nu_{FR} :\subseteq \Sigma^* \longrightarrow \mathbb{R}$ l'application définie par : $\nu_{FR}(u) = x$ si et seulement si $\nu_{\mathbb{Q}}(u) = x$ ou $u = \langle\alpha\rangle \diamond \langle\beta\rangle$, avec α suite $(\nu_{\mathbb{N}}, \nu_{\mathbb{Q}})$-calculable de nombres rationnels convergeant vers x, et β module de convergence $(\nu_{\mathbb{N}}, \nu_{\mathbb{N}})$-calculable pour la suite α. On notera $\mathbb{R}_{FR}$ l'ensemble des réels dont ν_{FR} est la notation. Ses éléments seront appelés **FR-nombres** ou **nombres réels constructibles**.*

*Soit $\nu_F :\subseteq \Sigma^* \longrightarrow \mathbb{R}$ l'application définie par : $\nu_F(u) = x$ si et seulement si $\nu_{\mathbb{Q}}(u) = x$ ou $u = \langle\alpha\rangle\diamond$, avec α suite fondamentale $(\nu_{\mathbb{N}}, \nu_{\mathbb{Q}})$-calculable de nombres rationnels convergeant vers x. On notera $\mathbb{R}_F$ l'ensemble des réels dont ν_F est la notation. Ses éléments seront appelés **F-nombres**.*

*Soit $\nu_q :\subseteq \Sigma^* \longrightarrow \mathbb{R}$ l'application définie par : $\nu_q(u) = x$ si et seulement si $\nu_{\mathbb{Q}}(u) = x$ ou $u = \langle\alpha\rangle\diamond$, avec α suite quasi-fondamentale $(\nu_{\mathbb{N}}, \nu_{\mathbb{Q}})$-calculable de nombres rationnels convergeant vers x. On notera $\mathbb{R}_q$ l'ensemble des réels dont ν_q est la notation. Ses éléments seront appelés **quasi-nombres**.*

*Soit $\nu_p :\subseteq \Sigma^* \longrightarrow \mathbb{R}$ l'application définie par : $\nu_p(u) = x$ si et seulement si $\nu_{\mathbb{Q}}(u) = x$ ou $u = \langle\alpha\rangle\diamond$, avec α suite pseudo-fondamentale $(\nu_{\mathbb{N}}, \nu_{\mathbb{Q}})$-calculable de nombres rationnels convergeant vers x. On notera $\mathbb{R}_p$ l'ensemble des réels dont ν_p est la notation. Ses éléments seront appelés **pseudo-nombres**.*

Les notions de FR-nombres, F-nombres, quasi-nombres et pseudo-nombres ont été introduites par Shanin en 1962. On peut créer une théorie sur les réels calculables à partir de chacune d'entre elles. Celle de FR-nombre est celle qui correspond le mieux à la définition intuitive mentionnée ci-dessus, car c'est la seule à donner effectivement un moyen de connaître la précision de l'approximation. Kushner définit ces nombres par leur notation, et non comme ici par les réels auxquels ils sont associés, ce qui ne permet pas vraiment de comparer les ensembles. Essayons donc maintenant de voir les relations entre ces différents ensembles et l'ensemble des réels calculables au sens de Weihrauch.

Proposition 4.1.4

$$\mathbb{R}_{FR} \subseteq \mathbb{R}_F \subseteq \mathbb{R}_q \subseteq \mathbb{R}_p$$

démonstration : Il est clair que si x est un FR-nombre et peut s'écrire $\langle\alpha\rangle \diamond \langle\beta\rangle$, la suite α est fondamentale, donc $\nu_F(\langle\alpha\rangle\diamond) = x$. Enfin fondamentale $\Longrightarrow$ quasi-fondamentale $\Longrightarrow$ pseudo-fondamentale. □

On peut même affirmer que $\mathbb{R}_{FR} = \mathbb{R}_F$. ν_{FR} et ν_F sont deux notations du même ensemble. Pour les autres inclusions, le problème est beaucoup plus difficile à résoudre. Kushner montre qu'il existe des ν_p-noms qui ne sont pas des ν_q-noms. Il est impossible de construire un ν_q-nom qui ne soit pas un ν_F-nom. En revanche, il est possible de donner des exemples de mots de la forme $\langle\alpha\rangle\diamond$, dont on sait montrer que c'est un ν_q-nom, mais tel qu'on ne connaît pas actuellement de preuve que c'est un ν_F-nom[4].

L'ensemble $\mathbb{R}_{FR}$ semble donc le mieux correspondre à la notion de réel calculable, car sa notation donne une information sur la vitesse de convergence, ce qui est nécessaire pour pouvoir utiliser le résultat d'un calcul. Plaçons-nous donc dans cet ensemble, et considérons les notations induites par ν_F, ν_q, ν_p et ρ_C en restreignant leurs domaines pour qu'elles soient à valeurs dans $\mathbb{R}_{FR}$. On les notera abusivement de la même façon.

Proposition 4.1.5 *Dans $\mathbb{R}_{FR}$, on a $\nu_{FR} \leq \nu_F \leq \nu_q \leq \nu_p$.*

démonstration : $\star$ $\nu_{FR} = \nu_F \circ f_M$ avec f_M définie sur $dom(\nu_{FR})$ par $f_M(\langle\alpha\rangle \diamond \langle\beta\rangle) = \langle\alpha\rangle\diamond$.
$\star$ La restriction de ν_q à $dom(\nu_F)$ est égale à ν_F, et celle de ν_p à $dom(\nu_q)$ est égale à ν_q. □

La notation ν_{FR} est la plus riche en information. Etant donné qu'elle est la seule vraiment utilisable, Kushner la choisit pour définir la notion de réel calculable. Il en découle une notion équivalente à celle de Weihrauch :

Théorème 4.1.6 *Les FR-nombres sont les nombres ρ_C-calculable au sens de Weihrauch.*

démonstration : $\star$ Soit x un FR-nombre, et α, β deux suites calculables telles que $\nu_{FR}(\langle\alpha\rangle \diamond \langle\beta\rangle) = x$. Posons $v_i = \langle\alpha\rangle(\beta(i+2))$. La suite $q = v_0\#v_1\#...$ est calculable, et on a (pour $i > j$) :

$$\begin{array}{rcl} |\nu_{\mathbb{Q}}(v_i) - \nu_{\mathbb{Q}}(v_j)| & \leq & |\alpha(\beta(i+2)) - x| + |\alpha(\beta(j+2)) - x| \\ & \leq & 2^{-(i+2)} + 2^{-(j+2)} \\ & \leq & 2^{-j-1} < 2^{-j} \end{array}$$

enfin la suite $(\nu_{\mathbb{Q}}(v_i))_{i\in\mathbb{N}}$ converge vers x, car c'est une suite extraite de $(\alpha(i))_{i\in\mathbb{N}} = (\nu_{\mathbb{Q}}(\langle\alpha\rangle\ (i)))_{i\in\mathbb{N}}$.
$\star$ Réciproquement, soit x un nombre ρ_C-calculable. Soit M une machine qui

4. Pour plus de précision, on se reportera à la page 88 de [Kus 85].

calcule x pour la représentation ρ_C. On peut construire à partir de M une machine M' qui prend en entrée un entier i et qui écrit en sortie le mot compris entre le i-ème et le $(i+1)$-ème $\#$ de la sortie de M. Soit $\mathcal{M}$ son programme. Enfin soit β la suite définie par $\forall i \in \mathbb{N}\ \beta(i) = i$. β est un module de continuité pour la suite calculée par M, d'après la définition de la représentation de Cauchy. On a donc $\nu_{FR}(\mathcal{M} \diamond \langle\beta\rangle) = x$. □

Malheureusement, les systèmes de noms ρ_C et ν_{FR} ne sont pas équivalents. Il n'y a en effet aucun moyen de retrouver, à partir d'un ρ_C-nom, le programme qui l'a engendré :

Proposition 4.1.7 *Dans $\mathbb{R}_{FR}$, $\nu_{FR} \leq \rho_C$, et $\rho_C \not\leq_t \nu_{FR}$.*

démonstration : $\star$ La démonstration du théorème 4.1.6 nous donne une méthode effective pour passer de la notation ν_{FR} à ρ_C sur $\mathbb{R}_{FR}$, ce qui nous donne la première partie de la propriété.
$\star$ Pour montrer, l'autre partie, supposons que $\rho_C \leq_t \nu_{FR}$. Soit donc $f :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \Sigma^*$ une fonction continue telle que $\rho_C = \nu_{FR} \circ f$. Soit $p = (1\#)^{\mathbb{N}}$. On a $\rho_C(p) \in \mathbb{R}_{FR}$. Comme f est continue, il existe une boule centrée en p dont l'image est incluse dans l'ouvert $\{f(p)\}$ (avec la distance définie dans le chapitre 2). Soit $a\Sigma^{\mathbb{N}}$ une telle boule, avec $a = (1\#)^n$. Soit $b \in \Sigma^*$ tel que $\nu_{\mathbb{Q}}(b) = 1 + 2^{-n}$. Soit $q = (1\#)^n(b\#)^{\mathbb{N}}$. La suite q est calculable, car périodique à partir d'un certain rang. On a $f(p) = f(q)$, donc $\nu_{FR}(f(p)) = \nu_{FR}(f(q))$, alors que $\rho_C(p) \neq \rho_C(q)$, ce qui contredit $\rho_C = \nu_{FR} \circ f$. □

On devine deja que les notions de calculabilité engendrée par les deux modèles risquent de ne pas être tout-à-fait équivalentes, même si les notions de réels calculables coïncident.

4.1.3 L'ensemble des réels calculables

Les réels non-calculables existent bien. En effet, un nombre est dit *calculable* s'il existe une machine de Turing de type 2 capable de l'écrire sur le ruban de sortie. On peut proposer un codage de la machine par un mot (de longueur finie). L'ensemble des mots étant dénombrable, l'ensemble des réels calculables est au plus dénombrable.

En fait, on peut même donner un exemple de réel non calculable : Soit $A \subseteq \mathbb{N}$ un ensemble non-récursif, et $x = \Sigma_{i \in A} 2^{-i}$. Le nombre x n'est pas calculable. (voir [Wei 95, page 21])

Enonçons quelques propriétés de l'ensemble $\mathbb{R}_{FR}$ des réels calculables.

Théorème 4.1.8

- *$\mathbb{R}_{FR}$ est dense dans $\mathbb{R}$.*

– $\mathbb{R}\backslash\mathbb{R}_{FR}$ *est dense dans* $\mathbb{R}$ *.*

démonstration $\star$ La première affirmation est évidente puisque $\mathbb{Q} \subseteq \mathbb{R}_{FR}$.
$\star$ Pour la deuxième, considérons deux réels $x < y$. Notons $x = \Sigma_{i\in\mathbb{Z}} a_i.2^i$ et $y = \Sigma_{i\in\mathbb{Z}} b_i.2^i$, avec $\forall i\ a_i, b_i \in \{0,1\}$. On peut supposer que la suite $(a_{-i})_{i\in\mathbb{N}}$ n'est pas stationnaire égale à 1. Soit $n \in \mathbb{Z}$ tel que $a_n \neq b_n$, et $m < n$ tel que $a_m = 0$. Soit $A \subseteq \mathbb{N}$ un ensemble non récursif. Soit $z = \Sigma_{i>m} a_i.2^i + 2^m + \Sigma_{i\in A} 2^{-i+m-1}$. Il est facile de montrer que $x < z < y$ et que z n'est pas calculable. $\square$

Par ailleurs, $\mathbb{R}_{FR}$ est dénombrable. Mais peut-on l'énumérer effectivement ? C'est-à-dire existe-t-il un programme qui en donne la liste complète ? Nous allons voir que non. Weihrauch montre le résultat suivant :

Théorème 4.1.9 *Soit* $(u_i)_{i\in\mathbb{N}}$ *une suite* (ν_{bin}, ρ_C)*-calculable. Il existe un réel* x ρ_C*-calculable tel que* $\forall i \in \mathbb{N}\ x \neq u_i$*.*

Un résultat encore plus fort est montré dans [Kus 85] :

Théorème 4.1.10 *On peut construire une machine* M *qui calcule une fonction* f_M *telle que : pour toute suite* (ν_{bin}, ν_{FR})*-calculable* $(u_i)_{i\in\mathbb{N}}$ *de réels,* $f_M(\langle u\rangle) \in dom(\nu_{FR})$ *et* $\forall n\ \nu_{FR}(f_M(\langle u\rangle)) \neq u_n$*.*

Ce théorème, dans lequel on peut remplacer ν_{FR} par ρ_C, se démontre grâce à un argument « diagonal ». D'après la remarque de la page 11, cette propriété implique que $\mathbb{R}_{FR}$ n'est pas r.e. dans $\mathbb{R}$.

4.2 Les fonctions calculables

Définition 4.2.1 *Dans l'approche russe, une fonction* $f :\subseteq \mathbb{R}_{FR} \longrightarrow \mathbb{R}_{FR}$ *est dite **calculable** si elle est* (ν_{FR}, ν_{FR})*-calculable.*

On a vu que les réels calculables des deux approches étaient les mêmes. Puisque même dans l'approche polonaise il n'est pas possible de travailler physiquement sur des réels non-calculables, les deux approches seront équivalentes si les notions de fonctions calculables coïncident. En fait, la notion de fonction calculable de Grzegorczyk est définie pour des fonctions de $\mathbb{R}$ dans $\mathbb{R}$, et non de $\mathbb{R}_{FR}$ dans $\mathbb{R}_{FR}$. Grâce à la densité de $\mathbb{R}_{FR}$ dans $\mathbb{R}$, on pourra dire que les deux approches seront équivalentes si :
$f : \mathbb{R}_{FR} \longrightarrow \mathbb{R}_{FR}$ est (ν_{FR}, ν_{FR})-calculable $\Longleftrightarrow$ son prolongement par continuité sur $\mathbb{R}$ est (ρ_C, ρ_C)-calculable.

Cette équivalence n'est malheureusement pas vraie dans le cas général. Toute fonction (ρ_C, ρ_C)-calculable a une restriction à $\mathbb{R}_{FR}$ qui est (ν_{FR}, ν_{FR})-calculable, mais la réciproque n'est pas toujours vraie. Cependant elle a été montrée dans certains cas importants, notamment :

Théorème 4.2.2 (Ceitin) *Soit* $f :\subseteq \mathbb{R}_{FR} \longrightarrow \mathbb{R}_{FR}$. *On suppose qu'il existe un ensemble* $X \subseteq dom(\nu_{FR})$ *r.e. tel que* $\nu_{FR}(X)$ *est dense dans* $dom(f)$. *Alors :*
$f : \mathbb{R}_{FR} \longrightarrow \mathbb{R}_{FR}$ *est* (ν_{FR}, ν_{FR})*-calculable* $\Longleftrightarrow$ *son prolongement par continuité sur* $\mathbb{R}$ *est* (ρ_C, ρ_C)*-calculable.*

Les relations entre l'approche russe et l'approche polonaise ne sont pas encore totalement comprises. On ne sait pas exactement dans quels cas elles sont équivalentes.

Chapitre 5

Autre approche : le modèle « real-RAM »

Les approches présentées jusqu'ici représentent fidèlement ce qui est physiquement réalisable par des machines. Cependant, on a souvent besoin de travailler sur des fonctions non calculables, par exemple discontinues. On est obligé de proposer des modèles permettant une approximation du calcul de telles fonctions.

Une fois l'hypothèse faite que les machines savent travailler sur de telles fonctions, c'est-à-dire en supposant que le modèle de calcul adopté s'en approche suffisamment, on peut proposer une théorie de la calculabilité sur ces objets. C'est ce que fait le modèle *real-RAM*, en considérant des machines capables de travailler sur des nombres réels en tant qu'objets de base. Cela permet d'étendre la notion de calculabilité aux ensembles de réels de manière très proche de la calculabilité classique.

Ce chapitre porte sur les travaux de L. Blum, M. Shub et S. Smale. (voir [BSS 89] et [Sma 92]).

5.1 Exemples

De nombreuses fonctions réelles – ou ensembles de réels – sont décrits de manière algorithmique. Il suffit par exemple de considérer la fonction définie sur $\mathbb{R}$ par

$$f(x) = \begin{cases} 0 \text{ si } x < 0 \\ 1 \text{ sinon.} \end{cases}$$

Cette fonction n'est pas réalisable dans le modèle de Weihrauch. Mais nos ordinateurs sont heureusement capables de la calculer – non pas sur $\mathbb{R}$, mais sur l'ensemble des nombres dénotables en virgule flottante, qui ont un nombre de chiffres significatifs fixe, et une plage limitée.

L'exemple donné dans [BSS 89] est celui des ensembles de Julia :
Soit $g : \mathbb{C} \longrightarrow \mathbb{C}$ une application polynômiale. On notera multiplicativement la composition (donc g^2 représentera $g \circ g$...).

Lemme 5.1.1 *Il existe une constante $C \in \mathbb{R}$ telle que si $|z| > C$, alors $|g^k(z)| \longrightarrow +\infty$ quand $k \longrightarrow +\infty$.*

Considérons l'algorithme suivant :
tant que $|z| < C$ **faire**
$z \leftarrow g(z)$

L'algorithme s'arrête sur les points qui tendent vers $+\infty$ en itérant la fonction g. Cela correspond exactement à la notion d'ensemble *récursivement énumérable* de la calculabilité classique.

En prenant $g(z) = z^2 + c$, l'ensemble sur lequel l'algorithme s'arrête est le complémentaire de l'ensemble de Julia associé à la valeur c.

Un des résultats montrés dans [BSS 89] est l'indécidabilité de la plupart des ensembles de Julia : leurs complémentaires sont r.e., mais ils ne le sont pas eux-mêmes. Un autre exemple d'ensemble non r.e. dont le complémentaire est r.e. est le classique ensemble de Cantor.

5.2 Machines réelles

Afin de formaliser la notion d'algorithme sur un anneau ordonné quelconque R, on doit définir notre modèle de calcul. Dans [BSS 89], les machines sont définies par un graphe, avec des noeuds typés. On peut en donner une définition plus formelle de la manière suivante :

Définition 5.2.1 *Une machine M sur un anneau R est un 14-uplet $(E, S, T, e, s, Q, Q_c, Q_b, Q_f, q_0, \delta_{cq}, \delta_{ct}, \delta_b, \delta_t)$, où :*

- *$E = R^l$ est appelé espace d'entrée,*
- *$S = R^m$ est appelé espace de sortie,*
- *$T = R^n$ est appelé espace de travail,*
- *$e : E \longrightarrow T$ est une application injective linéaire,*
- *$s : T \longrightarrow S$ est une application linéaire,*

- *Q, Q_c, Q_b et Q_f sont des ensembles d'états qui vérifient $Q = Q_c \cup Q_b \cup Q_f$, Q_c, Q_b et Q_f deux à deux disjoints, et $q_0 \in Q$. Les éléments de Q_c sont appelés états de calcul, ceux de Q_b états de branchement, et ceux de Q_f états finaux.*
- *$q_0 \in Q$ est appelé état initial,*
- *$\delta_b : Q_b \longrightarrow Q \times Q$*
- *$\delta_{cq} : Q_c \longrightarrow Q$*
- *$\delta_{ct} : Q_c \times T \longrightarrow T$, telle que $\forall q \in Q_c$, $x \longmapsto \delta_{ct}(q, x)$ est polynômiale,*
- *$\delta_t : Q_b \times T \longrightarrow R$, telle que $\forall q \in Q_b$, $x \longmapsto \delta_t(q, x)$ est polynômiale.*

Pour comprendre le fonctionnement de la machine, on peut supposer qu'elle dispose d'une « case mémoire » (ou *variable*), que l'on notera $\mathcal{M}$, capable de contenir un élément de T. Sa valeur sera appelée *valeur courante*, et sera notée m. L'*état courant* sera noté q. On utilisera les fonctions de projection p_1 et p_2 : $\forall x, y \in R$ $p_1(x, y) = x$ et $p_2(x, y) = y$.

La machine fonctionne de la manière suivante, sur l'entrée $x \in E$:

1. On place $e(x)$ dans $\mathcal{M}$, $q = q_0$.
2. On répète les opérations suivantes jusqu'à ce que $q \in Q_f$:
   - Si $q \in Q_c$, on place la valeur $\delta_{ct}(q, m)$ dans $\mathcal{M}$, et l'état courant devient $\delta_{cq}(q)$.
   - Si $q \in Q_b$, on se place dans l'état $p_1(\delta_b(q))$ si $\delta_t(q, m) < 0$, dans l'état $p_2(\delta_b(q))$ sinon.
3. Le résultat du calcul, si la machine s'arrête, est $s(m)$.

La fonction e peut permettre de réserver de l'espace de travail (prendre par exemple l'injection canonique de $\mathbb{R}^l$ dans $\mathbb{R}^m$, avec $m > l$). Dans le cas où R est un corps, on peut autoriser les fonctions rationnelles au lieu de polynômiales. On peut représenter une machine par un graphe. Celui de la figure 5.1 correspond à l'algorithme de la page 29. Blum, Shub et Smale proposent aussi une modification de cette définition pour permettre de travailler sur $R^{\mathbb{N}}$.

Dans le modèle *real-RAM*, on définit les notions suivantes :

Définition 5.2.2 *Une fonction f est dite* ***calculable*** *par une machine real-RAM s'il existe une machine M telle que : sur l'entrée $x \in dom(f)$, M se place en un temps fini dans un état final avec $f(x)$ comme résultat, et M boucle sur toutes les autres entrées.*

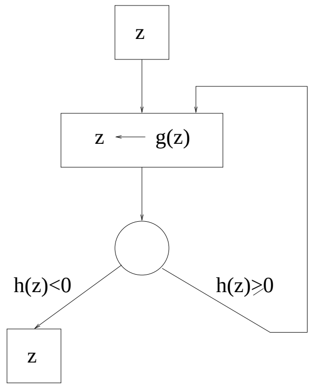

FIG. 5.1 - *Machine qui s'arrête sur le complémentaire de l'ensemble de Julia*

On définit les ensembles *récursivement énumérables* et *récursifs* comme dans la définition 1.3.1, en remplaçant la machine de type 2 par une machine *real-RAM*. Comme d'habitude, un ensemble X est récursif dans W si X et $W\backslash X$ sont r.e. Il suffit pour le démontrer, de travailler sur la machine « produit » de M_1 et M_2 (où X est l'intersection de W avec le domaine de la fonction calculée par M_1 – *idem* pour $W\backslash X$ avec M_2), c'est-à-dire une machine travaillant sur $T' = T_1 \times T_2$, $Q' = Q_1 \times Q_2$, avec $Q'_f = Q_{f1} \times Q_2 \cup Q_1 \times Q_{f2}$...

Le modèle ne permet bien sûr pas de définir la notion de nombre réel calculable. On peut travailler avec tous les réels. On peut même considérer des constantes « de construction », c'est-à-dire des constantes réelles connues par la machine dès le départ. Par exemple, on peut construire une machine qui décide un sous-ensemble A de $\mathbb{N}$ en y introduisant la constante réelle $s = \sum_{i \in \mathbb{N}^*} s_i . 2^{-i}$ définie par :

$$s_i = \begin{cases} 0 \text{ si } i \in A \\ 1 \text{ sinon.} \end{cases}$$

La machine peut savoir ainsi si un nombre appartient où non à A, en effectuant seulement quelques calculs sur le nombre s. Tout ensemble d'entiers est donc récursif dans ce modèle.

Chapitre 6

Complexité

La calculabilité classique est prolongée par une étude de la complexité des algorithmes. Dans le cas des machines de Turing, on peut par exemple définir la complexité en temps (nombre de pas nécessaires pour un calcul) et en espace (nombre de case visitées sur les rubans). Définir la complexité d'un algorithme sur les réels n'est pas aussi simple que pour un algorithme sur des données discrètes. La notion de complexité va évidemment dépendre du modèle.

6.1 Modèle de Weihrauch

Les machines de Turing de type 2 ne s'arrêtent pas toujours. On est obligé de changer la notion de complexité d'un calcul en la faisant dépendre du nombre de caractères écrits en sortie. Il est également intéressant de regarder le nombre de caractères lus sur les rubans d'entrée pour écrire un nombre de caractères donné en sortie. On peut par exemple définir la complexité d'un calcul et la *longueur d'entrée nécessaire* de la manière suivante :

Définition 6.1.1 *Soit M une machine de type 2, qui calcule une fonction $f_M :\subseteq (\Sigma^{\mathbb{N}})^m \longrightarrow \Sigma^{\mathbb{N}}$. Soit $y \in (\Sigma^{\mathbb{N}})^m$, et $k \in \mathbb{N}$.*

- *La **complexité en temps** pour la machine M sur l'entrée y est la fonction $Time_M(y) : \mathbb{N} \longrightarrow \mathbb{N}$ qui à une valeur k associe le nombre de pas nécessaires à M pour écrire le k-ième caractère sur le ruban de sortie, en prenant y en entrée.*
- *La **longueur d'entrée nécessaire** (input lookahead) est la fonction $Len_M(y) : \mathbb{N} \longrightarrow \mathbb{N}$ qui à une valeur k associe le plus grand nombre de caractères lus sur un des rubans d'entrée pendant les $Time_M(y)(k)$ premiers pas de calcul.*

On peut donner des définitions similaires pour Σ^*. On voudrait, comme toujours, étendre cette notion de $\Sigma^{\mathbb{N}}$ à $\mathbb{R}$. On est confronté à un problème : on aimerait pouvoir dire qu'une fonction réelle est calculable en temps $t : \mathbb{N} \longrightarrow \mathbb{N}$ s'il existe une machine qui la calcule en temps t par l'intermédiaire de ρ_C-noms. Hélas la longueur d'un préfixe d'un ρ_C-nom n'a rien à voir avec la précision de l'approximation. En effet, la notation standard de $\mathbb{Q}$ permet d'employer des noms aussi longs que l'on veut. Tout ρ_C-nom peut être arbitrairement étiré sans changer la valeur représentée. En allongeant ainsi la sortie, on peut construire, pour toute fonction calculable, une machine qui la calcule en temps $O(n)$.

On est donc obligé de définir la complexité autrement. Il faudrait que la fonction de complexité dépende non du nombre de caractères écrits, mais de la précision du résultat. Mais là encore, si les entrées peuvent être étirées à volonté, la complexité dépendra toujours du ρ_C-nom choisi, puisque la machine doit au moins effectuer un pas pour lire chaque caractère d'un ruban d'entrée.

Pour résoudre le problème, Weihrauch interdit l'allongement arbitraire des noms en utilisant la représentation particulière définie ci-après. Il s'agit d'une représentation binaire modifiée, où l'on autorise les chiffres 1, 0 et -1. On notera $\overline{1}$ le chiffre -1.

Définition 6.1.2 *Soit $\rho :\subseteq \Sigma^{\mathbb{N}} \longrightarrow \mathbb{R}$ la représentation définie par*

$$\rho(a_n...a_0, a_{-1}a_{-2}...) = \sum_{i \geq n} a_i.2^i$$

sur le domaine suivant :

$$\begin{aligned} dom(\rho) = \quad & \{a_n...a_0, a_{-1}a_{-2}.../n \geq -1,\ a_i \in \{\overline{1}, 0, 1\},\ a_n \neq 0 \ si\ n \geq 0, \\ & et\ a_n a_{n-1} \notin \{\overline{1}1, 1\overline{1}\}\ si\ n \geq 1\} \end{aligned}$$

Pour cette représentation, la précision du résultat dépend directement du nombre de caractères pris en compte. En effet, chaque chiffre supplémentaire divise l'intervalle d'approximation par 2.

Contrairement à la représentation binaire classique, cette représentation est équivalente à ρ_C, et est donc admissible pour la topologie $\tau_{\mathbb{R}}$.

En imposant l'emploi de cette représentation, on peut généraliser les définitions des fonctions $Time$ et Len. Weihrauch montre que l'addition se fait en temps $O(n)$, la multiplication peut se faire dans les mêmes temps que sur des entiers avec des machines de Turing classiques (par exemple $n.\log n.\log\log n$).

Le fait d'imposer une représentation pour pouvoir calculer la complexité ne me paraît pas une solution très satisfaisante. Cela montre que même dans cette approche, qui ressemble beaucoup à la calculabilité classique, la notion de complexité est assez vague et encore mal définie. On peut bien sûr fixer la représentation ρ et imposer de ne travailler qu'avec elle. On obtient alors un modèle cohérent que l'on peut étudier, y compris du point de vue de la complexité. Cependant on ne peut être certain que la complexité ne pourra pas être meilleure avec d'autres représentations. La représentation adéquate peut aussi dépendre du calcul. Le choix de la représentation, même parmi les représentations admissibles, est toujours arbitraire.

6.2 Approche russe

L'approche russe ne permet pas de définir une théorie de la complexité. En effet, il est impossible de lier directement la complexité à un calcul sur une valeur réelle, car cela dépendra encore du nom choisi.

6.3 Modèle *real-RAM*

Pour le modèle *real-RAM*, l'étude de la complexité est beaucoup plus simple. On admet que les calculs polynômiaux sur les réels se font en une unité de temps. Le temps nécessaire pour effectuer un calcul est le nombre de transitions effectuées. Le critère uniforme est la *taille* de l'entrée, qui est définie de la manière suivante :

Définition 6.3.1 *Soit $x \in R^n$. On notera $x = (x_1, x_2, ...)$.*

*La **longueur** de x est le plus grand entier k tel que $x_k \neq 0$ si $x \neq 0$; la longueur de 0 est 1.*

*La **hauteur** de $y \in R$ est :*

- $\log(|y| + 1)$ *si* $R = \mathbb{Z}$,
- $\max(hauteur(p), hauteur(q))$ *où p et q sont deux entiers premiers entre eux tels que $y = p/q$, si $R = \mathbb{Q}$,*
- 1 *si* $R = \mathbb{R}$.

*La **hauteur** de x est le maximum des hauteurs de x_i, $1 \leq i \leq n$.*

*La **taille** de x est la somme de sa longueur et de sa hauteur.*

La hauteur d'un réel est toujours 1, puisque le coût réel d'une opération simple (multiplication) sur un réel ne dépend pas de sa valeur. La hauteur d'un entier est le nombre de chiffres nécessaires pour l'écrire, à un coefficient multiplicatif près en fonction de la base.

On définit enfin le *coût* d'un calcul, comme le produit du temps par la hauteur maximale survenant pendant le calcul.

Le modèle permet également de définir, comme dans le cas classique, les classes P et NP. Nous nous intéresserons plus particulièrement aux problèmes de décision, c'est-à-dire d'appartenance à un ensemble.

Définition 6.3.2 *Un algorithme est dans la classe P s'il se déroule avec un coût polynômial en la taille de l'entrée.*

*Un problème de décision d'un ensemble $Y_1 \subseteq Y$ est dans la classe NP (*non déterministe polynômial*) s'il existe une machine M sur R calculant une fonction f_M et deux entiers naturels c et q tels que :*

- *$dom(f_M) = Y$ et $\forall y, y' \in Y$ $f_M(y, y') = 0$ ou 1*
- *si $y \notin Y_1$, $f_M(y, y') = 0$,*
- *pour tout $y \in Y_1$, il existe un y' tel que $f_M(y, y') = 1$ et $cout_M(y, y') \leq c.(taille(y))^q$.*

Si $R = \mathbb{Z}$, la définition de la classe NP coïncide avec la définition standard.

Les auteurs montrent que le problème sur R de décision du Voyageur de Commerce, analogue du problème classique, est dans la classe NP (voir [BSS 89]). Les auteurs démontrent un théorème analogue au théorème de Cook. L'article aborde également la question $P \stackrel{?}{=} NP$ et tente de la relier avec le problème classique sur $\mathbb{Z}$.

La théorie de complexité « basée sur l'information » présentée dans [TWW 88] utilise un modèle de calcul très proche. Un des problèmes abordés est le calcul d'une valeur f avec avec une précision donnée ε. Les machines utilisées posent une suite de questions sur f puis déterminent le résultat en fonction des réponses obtenues. Toutes les questions sont permises, à condition que la réponse soit toujours oui ou non. On peut voir ceci comme des questions posées à des oracles qui savent déterminer l'appartenance à un ensemble ; on peut utiliser autant d'oracles que l'on veut. On affecte à chaque question un coût.

Une fois toutes les questions posées, on détermine le résultat en effectuant des opérations simples comme les opérations arithmétiques et les comparaisons sur des nombres réels. On fixe un coût pour chaque opération (par exemple 1). On suppose que chaque opération est réalisée exactement. Le coût du calcul est la somme des coûts de la détermination des informations et du calcul du résultat.

Les auteurs étudient successivement les notions de complexité *au pire, en moyenne* et grâce à une méthode probabiliste. Ce modèle permet une étude théorique de la complexité, mais il ne correspond en fait à aucun modèle de calcul bien défini. En effet, ce modèle ne peut être utilisé tel quel pour n'importe quelle opération (il suffirait par exemple de poser la question à un oracle pour déterminer si un nombre est premier – ce qui donnerait une complexité en temps de 1 pour la recherche des nombres premiers !). Dès que l'on utilise un oracle, la notion de complexité expérimentale est faussée. Avant de calculer la complexité, il faut donc définir les questions à poser et les opérations autorisées. Une fois l'algorithme validé, on peut en calculer la complexité.

Conclusion

La diversité des modèles de calcul sur les réels montre bien que la notion de calculabilité en analyse n'est pas encore vraiment établie. Elle témoigne également de la difficulté de travailler avec des machines sur des ensembles non dénombrables. On se contente d'ailleurs actuellement sur les ordinateurs d'approximations qui ne correspondent à aucune théorie de la calculabilité. C'est la raison pour laquelle la validité de tout calcul sur les réels doit être discutée avant de pouvoir en utiliser le résultat.

Les approches russe et polonaise sont les seules qui proposent vraiment un moyen de faire des calculs « exacts » sur les réels. Elles donnent un moyen effectif de travailler non seulement sur les réels, mais aussi sur tout ensemble qui peut être représenté par des suites infinies sur un alphabet fini. Pourrait-on étendre à nouveau ce modèle pour pouvoir travailler sur des ensembles encore plus grands comme l'ensemble $\{0,1\}^{\mathbb{R}}$ des parties de $\mathbb{R}$?

Même s'il n'est pas implantable directement, le modèle *real-RAM* a un intérêt non négligeable, puisqu'il s'affranchit de tout problème de calculabilité des nombres, ce qui permet par exemple de travailler sur les ensembles de réels de manière beaucoup plus naturelle. Par exemple, le théorème 1.3.3, qui rend la notion d'ensemble récursif peu intéressante dans la théorie de Weihrauch n'est pas vérifié. Le modèle permet aussi de calculer des fonctions discontinues, ce qui est utile dans la pratique. C'est donc le modèle qui correspond le mieux à la notion de calcul utilisée en analyse numérique. Cependant de nombreuses fonctions courantes et calculables pour Weihrauch, comme $\sin x$ ou e^x, ne sont pas calculables dans ce modèle... Une variante du modèle *real-RAM*, appelée *feasible real-RAM*, est réalisable physiquement.

D'autres théories existent sur le sujet. On peut par exemple mentionner la logique constructiviste qui donne une version effective de l'analyse en n'autorisant que les preuves constructives. Par exemple, $\exists x$ signifie que l'on dispose d'un algorithme de calcul d'un x vérifiant les conditions précisées. (voir la note page 22).

Bibliographie

[Wei 95] Klaus WEIHRAUCH, *A Simple Introduction to Computable Analysis*, in Informatik Berichte 171, FernUniversität, Hagen, 1995

[Wei 96] Klaus WEIHRAUCH, *A Foundation of Computable Analysis*, FernUniversität, Hagen, 1996

[Wei 87] Klaus WEIHRAUCH, *Computability*, Springer-Verlag, Berlin, Heidelberg, 1987

[Kus 85] B. A. KUSHNER, *Lectures on constructive mathematical analysis*, American Mathematical Society, Translation of mathematical monographs, vol. 60, Providence, 1985

[BSS 89] L. BLUM, M. SHUB, S. SMALE, *On a theory of computation and complexity over the real numbers*, Bulletin of the American Mathematical Society, vol. 21, pp. 1 - 46, 1989

[Grz 55] A. GRZEGORCZYK, *Computable functionals*, Fundamenta mathematica, vol. 42, pp. 168 - 202, 1955

[Grz 57] A. GRZEGORCZYK, *On the definition of computable real continuous functions*, Fundamenta mathematica, vol. 44, pp. 61 - 71, 1957

[TWW 88] J.F. TRAUB, G.W. WASILKOWSKI, H. WOZNIAKOWSKI, *Information-based Complexity*, Academic Press, New York, 1988

[Sma 92] Stephen SMALE, *Theory of computation*, in : C. CASSACUBERTA et al., Mathematical Research today and tomorrow, Springer-Verlag, Berlin, 1992

[Tur 36] Alan M. TURING, On computable real numbers, with an application to the Entscheidungsproblem, in Proceedings of the London Mathematical Society (2) vol. 42, pp. 230 - 265, Londres, 1936

[Tur 37] Alan M. Turing, On computable real numbers, with an application to the Entscheidungsproblem. A correction, in Proceedings of the London Mathematical Society (2) vol. 43, pp. 544 - 546, Londres, 1937

Index