Numéro d'ordre : 188-96
Année 1996

THÈSE

présentée devant

L'UNIVERSITÉ CLAUDE BERNARD - LYON 1

pour l'obtention

du DIPLÔME DE DOCTORAT
(arrêté du 30.3.92)
spécialité : INFORMATIQUE

par

Joëlle THOLLOT

Extension du Modèle IFS pour une Géométrie Fractale Constructive

Soutenue le 9 Septembre 1996, devant le jury composé de :

Président :	Denis Vandorpe
Rapporteurs :	Jacques Lévy Véhel Bernard Peroche Przemyslaw Prusinkiewicz
Examinateurs :	Michel Dekking Jacques Mazoyer Éric Tosan Denis Vandorpe

Remerciements

Je tiens à remercier Monsieur Denis Vandorpe, Professeur à l'Université Claude Bernard, pour m'avoir accueillie dans son équipe. Sa disponibilité et ses conseils pertinents m'ont été d'une aide précieuse tout au long de ma thèse.

Je tiens à exprimer toute ma gratitude à Monsieur Przemyslaw Prusinkiewicz, professeur à l'université de Calgary, Monsieur Jacques Levy-Vehel, directeur de recherche à l'IRIA Rocquencourt et Monsieur Bernard Peroche, professeur à l'École des Mines de St Etienne, pour la confiance qu'il m'ont accordé en acceptant de rapporter cette thèse.

Je remercie vivement Monsieur Michel Dekking, professeur à l'université de Delft et Monsieur Jacques Mazoyer, professeur à l'École Normale Supérieure de Lyon, pour l'interêt qu'ils ont porté à ce travail.

Que Éric Tosan, trouve ici l'expression de toute mon amitié. Il a su m'orienter dans mes recherche et m'apporter les connaisances scientifiques dont j'avais besoin. Je le remercie aussi pour tout le temps qu'il a consacré à des discussions qui m'ont apporté une connaissance scientifique mais aussi une ouverture sur le monde artistique. Son entousiasme constant m'a apporté la confiance nécessaire pour éffectuer cette thèse.

Je tiens à remercier toute l'équipe Art&Fract, et en particulier Martine Rondet-Mignotte pour avoir accepté d'utiliser mon travail à des fins artistiques. Leurs remarques, sortant du cadre scientifique, m'ont permis d'avoir un regard différent sur la démarche que j'ai adoptée.

Ce mémoire à été rendu possible grâce aux recherches réalisées au sein du LIGIM. Merci à toute l'équipe du LIGIM pour m'avoir encouragée dans mes travaux et pour m'avoir fait profiter des compétences de chacuns. En particulier, merci à Thierry Excoffier pour tout le temps qu'il m'a consacré et toutes les réponses qu'il a su me donner, aussi bien à des problèmes pratiques que théoriques.

Merci enfin à tout ceux qui m'ont aidé lors de la rédaction de ce manuscript, en particulier Matthieu Brilman qui m'a soutenue constamment, ainsi que Thomas Chaboud, Renée Thollot et Thierry Chich pour leur relecture minutieuse. Merci aux amis qui ont pris le temps de m'écouter et m'ont encouragée avec toute leur sympathie.

Table des matières

Introduction générale

La géométrie fractale a pris naissance à partir de certaines remarques concernant des objets qui mettaient en défaut la géométrie euclidienne, que ce soient des objets naturels (arbres, côtes de Bretagne ...) ou des objets mathématiques (flocon de Von Koch, courbe de Peano ...). MANDELBROT [Man83], à qui l'on doit le terme "fractal", a posé les premiers éléments de définition des fractales. Ses travaux ont donné naissance à de nombreuses recherches dans tous les domaines scientifiques; par exemple, l'étude des matériaux poreux, des turbulences, des failles géologiques, des fluctuations de la bourse, etc. Toutes ces études tendent à modéliser des phénomènes naturels, principalement à l'aide de la mesure de la dimension fractale. D'autres approches ont mené à des modèles particuliers permettant d'étudier plus précisément certains phénomènes. On peut citer, dans ce cadre, la théorie des L-systèmes, introduite par LINDENMAYER [PLH88] pour modéliser la croissance des plantes.

Nous nous situons dans le cadre de la modélisation géométrique pour la synthèse d'image. Nous avons donc choisi d'étudier un modèle permettant de caractériser de manière rigoureuse la géométrie fractale. Ce modèle est le modèle IFS, ou Iterated Function System, développé par BARNSLEY [Bar88] et s'inspirant de travaux antérieurs tels que ceux de HUTCHINSON [Hut81]. Ce modèle nous a semblé le plus intéressant dans notre démarche car il traduit mathématiquement une des propriétés principales des fractales qui est l'auto-similarité (au sens général). On peut la définir comme suit : un objet est auto-similaire s'il est la réunion de "copies", à une plus petite échelle, de lui-même. Le mot "copie" est à prendre au sens large. On entend par là "image par un opérateur donné". Lorsque cet opérateur est une transformation affine, on parlera d'auto-affinité.

La traduction de cette propriété en termes mathématiques est immédiate. L'objet K doit être tel que

$$K = T_1(K) \cup T_2(K) \ldots \cup T_N(K),$$

c'est-à-dire que K est égal à l'union de N transformées de lui-même. À partir de cette expression, on pourra définir un IFS et visualiser l'objet fractal associé que l'on appelle attracteur de l'IFS.

Ce modèle présente cependant deux inconvénients lorsque l'on veut l'utiliser pour la synthèse d'image. Le premier découle immédiatement de l'auto-similarité. En effet, cette propriété induit des régularités dans l'objet qui peuvent être gênantes, tant du point de vue de la modélisation (les objets naturels ne sont jamais parfaitement auto-similaires) que du point de vue graphique (qualités artistiques de l'image). Le deuxième inconvénient est la difficulté de contrôler le processus de création de l'image. En effet, la création consiste à choisir des transformations puis à calculer l'attracteur correspondant et il n'est pas toujours évident de prévoir le résultat.

Notre travail s'est articulé autour de ces deux limitations, le problème du contrôle étant le plus important à résoudre.

Cette question de la création d'une forme, se pose aussi lorsque l'on travaille en modélisation géométrique classique. Une des réponses proposées dans ce cadre est le modèle CSG (Constructive Solid Geometry). Le principe est de combiner des volumes simples pour obtenir des volumes compliqués. Nous avons donc cherché à trouver un système fondé sur ce principe, applicable à la géométrie fractale. Ce travail est dans la continuité de la thèse de Christian Gentil [Gen92] et permet de compléter cette démarche en proposant un système de géométrie fractale constructive.

Pour cela nous proposons deux modèles d'extension d'IFS. Nous reprenons la démarche entreprise par de nombreux auteurs qui consiste à associer un langage formel au modèle IFS. Mais contrairement à la plupart des travaux antérieurs nous considérons que le langage peut être quelconque. Ceci nous donne un modèle très bien formalisé, qui nous permettra de faire des démonstrations. D'autre part, nous proposons un modèle plus algorithmique, fondé sur un formalisme matriciel. Ces deux modèles vont nous permettre de retrouver la plupart des modèles antérieurs et ainsi d'unifier les différentes démarches existantes.

Nous proposons ensuite des opérations, inspirées de chacun des modèles. Ces opérations vont nous permettre de combiner des formes fractales simples et donc de construire des formes "pas-à-pas".

Une application possible de ce travail est la construction d'un modeleur utilisant la géométrie fractale pour des travaux artistiques. Il est actuellement développé par Chems Eddine Zair, Eric Tosan et moi-même, en collaboration avec deux plasticiennes, Martine Rondet-Mignotte et Hélène Bertin, et un physicien, Jean Fornazero.

Nous allons donc, dans un premier temps, présenter le modèle IFS et les travaux qui en ont découlé. Dans un deuxième temps, nous présenterons deux généralisations du modèle IFS et nous verrons que l'on peut ramener la plupart des modèles existants à notre approche. Enfin, nous définirons des opérations sur les modèles que nous avons définis, afin de permettre une démarche constructive.

Généralités

Chapitre 1

Le modèle IFS

Le modèle IFS a été étudié d'un point de vue mathématique par HUTCHINSON [Hut81] puis approfondi par BARNSLEY [Bar88][BJM+88] dans le cadre de la géométrie fractale. Nous avons choisi ce modèle pour faire de la synthèse d'image car c'est un modèle défini de manière rigoureuse. CHRISTIAN GENTIL [Gen92] a travaillé sur ce modèle lors de sa thèse et nous avons repris ses travaux pour les généraliser.

Le modèle IFS est entièrement fondé sur la notion d'auto-similarité, prise au sens large. L'idée est de coder une fractale par un ensemble de transformations traduisant cette propriété. L'ensemble de transformations sera appelé IFS et la fractale correspondante sera l'attracteur de l'IFS.

La notion d'auto-similarité se traduit simplement par la formule :

$$K = \bigcup_{i=1}^{N} T_i(K),$$

c'est-à-dire que la fractale K est égale à l'union de transformées d'elle-même. Nous allons donc construire des objets qui satisfont cette formule. Pour cela, nous allons définir précisement l'espace dans lequel se situent les objets et les propriétés suffisantes pour obtenir des objet auto-similaires à partir d'un ensemble de transformations.

1.1 Définitions

La théorie des IFS est fondée sur le Théorème du Point Fixe appliqué dans l'ensemble des compacts d'un espace métrique. Nous rappelons ici les principaux résultats utilisés par BARNSLEY dans ses démonstrations, puis la définition d'un IFS et de son attracteur.

1.1.1 Rappel sur le Théorème du Point Fixe

Définition 1.1.1 *Soient $(\mathcal{X}, d)$ un espace métrique et T une application de $\mathcal{X}$ à valeurs dans $\mathcal{X}$. T est dite d-contractante sur un domaine D si :*

$$\exists k < 1 \text{ tel que } \forall p_1, p_2 \in D \quad d(T(p_1), T(p_2)) \leq k d(p_1, p_2).$$

Lorsqu'il n'y a pas d'ambiguïté sur la métrique, nous disons que T est contractante sur D. Si $D = \mathcal{X}$ nous disons que T est contractante.

Théorème 1.1.1 (du Point Fixe) *Soit $(\mathcal{X}, d)$ un espace métrique complet. Soit $T : \mathcal{X} \to \mathcal{X}$ une application contractante. Alors il existe un unique point $c \in \mathcal{X}$, appelé le point fixe de T, tel que $T(c) = c$.*

Notation : On notera $T \circ p = T(p)$, l'action d'une transformation T sur un point p.

1.1.2 Espace des compacts

L'espace sur lequel on travaille est un espace métrique complet $(\mathcal{X}, d)$. On notera $\mathcal{H}(\mathcal{X})$ l'ensemble des compacts non vides de $(\mathcal{X}, d)$. En général, on prendra $\mathcal{X} = \mathbb{R}^2$ ou $\mathbb{R}^3$ mais les définitions que nous allons donner sont valables pour tout espace métrique complet.

On induit une topologie sur $\mathcal{H}(\mathcal{X})$ en définissant la distance de HAUSDORFF d_H.

Définition 1.1.2 (Distance de Hausdorff) *Soient K et K' deux compacts non vides d'un espace métrique $(\mathcal{X}, d)$, la distance de HAUSDORFF de K à K' est définie par*

$$d_H(K, K') = \max\{\max_{p_1 \in K} \min_{p_2 \in K'} d(p_1, p_2), \max_{p_1 \in K'} \min_{p_2 \in K} d(p_1, p_2)\}.$$

Théorème 1.1.2 *Si $(\mathcal{X}, d)$ est un espace métrique complet alors $(\mathcal{H}(\mathcal{X}), d_H)$ est un espace métrique complet.*

Notation : On notera $T \circ K = \{T \circ p/\ p \in K\}$ l'action d'une transformation sur un ensemble de points.

1.1.3 Opérateur de Hutchinson

À partir d'un ensemble de fonctions de $\mathcal{X}$ à valeurs dans $\mathcal{X}$ nous définissons sur $\mathcal{H}(\mathcal{X})$ l'opérateur de Hutchinson qui à une partie de $\mathcal{X}$ associe une autre partie de $\mathcal{X}$ de la façon suivante :

Définition 1.1.3 *Soient* $T_1, \ldots, T_N$, *N fonctions de* $\mathcal{X}$ *dans* $\mathcal{X}$. *L'opérateur défini par :*

$$\begin{array}{rccl} H: & \mathcal{P}(\mathcal{X}) & \longrightarrow & \mathcal{P}(\mathcal{X}) \\ & K & \longmapsto & \bigcup_{i=1}^{N} T_i \circ K = T_1 \circ K \cup \ldots \cup T_N \circ K \end{array}$$

est appelé opérateur de Hutchinson *associé à* $\{T_1, \ldots, T_N\}$.

Proposition 1.1.1 *H opère dans* $\mathcal{H}(\mathcal{X})$, *c'est-à-dire*

$$K \in \mathcal{H}(\mathcal{X}) \Rightarrow H(K) \in \mathcal{H}(\mathcal{X}).$$

Proposition 1.1.2 *Soient* $T_1, \ldots, T_N$, *N fonctions contractantes de* $\mathcal{X}$ *dans* $\mathcal{X}$, *de rapports de contraction* $s_1, \ldots, s_N$.

L'opérateur de Hutchinson *H associé à* $\{T_1, \ldots, T_N\}$, *est* d_H*-contractant, de rapport de contraction* $s = \max\{s_1, \ldots, s_N\}$.

1.1.4 Définition d'un IFS

On peut maintenant définir un IFS et son attracteur.

Définition 1.1.4 *Soit* $(\mathcal{X}, d)$ *un espace métrique complet. Nous appellerons IFS tout ensemble* $\mathcal{T} = \{T_1, \ldots, T_N\}$ *de fonctions contractantes sur* $\mathcal{X}$.

À tout IFS nous pouvons associer un opérateur de Hutchinson qui est contractant. Étant dans un espace métrique complet $(\mathcal{H}(\mathcal{X}), d_H)$, nous pouvons appliquer le Théorème du Point Fixe.

Théorème 1.1.3 (d'existence) *Soit $\mathcal{T}$ un IFS. Il existe un unique compact non vide K de $\mathcal{X}$ tel que :*

$$K = H(K) = T_1 \circ K \cup T_2 \circ K \cup \ldots \cup T_N \circ K.$$

K est appelé attracteur de $\mathcal{T}$ et sera noté $\mathcal{A}(\mathcal{T})$.

Notation : On notera $\mathcal{T} \circ K = \bigcup_{i=1}^{N} T_i \circ K$, l'action de l'opérateur de Hutchinson associé à l'IFS $\mathcal{T}$ sur le compact K.

Remarque : La théorie des IFS développée par Barnsley comporte un élément supplémentaire : à chaque transformation de l'IFS est associée une probabilité, il est alors possible de démontrer qu'il existe une mesure unique ayant pour support $\mathcal{A}(\mathcal{T})$ invariante par H [Bar88]. Cette mesure est utilisée comme texture sur la fractale. Nous n'avons pas abordé cet aspect dans nos travaux.

Exemple : Nous allons utiliser l'exemple d'un arbre, donné dans [PH91], pour illustrer les différents chapitres. Nous noterons les transformations affines de $\mathbb{R}^3$ de la façon suivante :

- $T(a, b, c)$ la translation selon le vecteur (a, b, c);
- $R(a)$ la rotation d'angle a et d'axe Ox;
- $H(a)$ l'homothétie de rapport a et de centre O.

l'IFS que nous allons étudier sera $\mathcal{T}_{arbre} = \{T_1, T_2, T_3, T_4\}$, avec :

$$\begin{aligned} T_1 &= H(0.5); \\ T_2 &= T(0, 0.5, 0) \circ H(0.5); \\ T_3 &= T(0, 1, 0) \circ R(\pi/4) \circ H(0.5); \\ T_4 &= T(0, 1, 0) \circ R(-\pi/4) \circ H(0.5). \end{aligned}$$

L'attracteur de cet IFS, représenté sur la Figure 1.1, est le point fixe de l'opérateur

$$H_{arbre} = T_1 \cup T_2 \cup T_3 \cup T_4.$$

On le notera $\mathcal{A}_{arbre}$ et on l'appelera l'arbre.

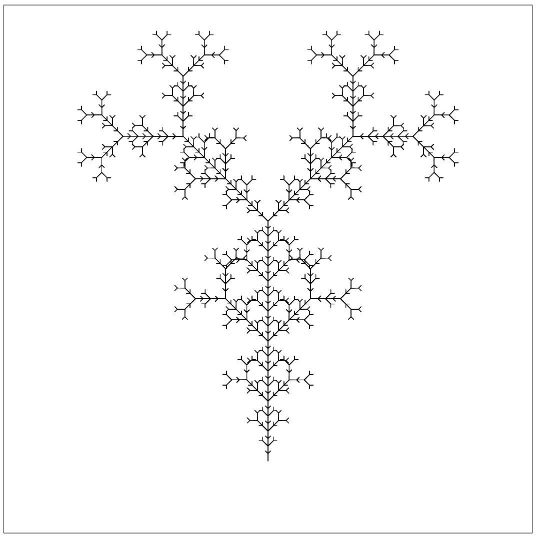


FIG. 1.1 - *Attracteur de* $\mathcal{T}_{arbre}$.

1.2 Méthode de construction

Les théorèmes précédents montrent l'existence et l'unicité du compact associé à un ensemble de fonctions contractantes $\mathcal{T} = \{T_1, \ldots, T_N\}$. Cependant, aucune méthode de construction n'a été donnée. Or, il est indispensable d'avoir de telles méthodes, ne serait-ce que pour visualiser ce compact.

Nous cherchons à construire des suites de compacts $(K_n)_{n>0}$ qui convergent vers l'attracteur [GV90] :

$$\lim_{n\to\infty} K_n = \mathcal{A}(\mathcal{T}).$$

Pour cela nous utilisons un résultat général qui est une conséquence du Théorème du Point Fixe.

Proposition 1.2.1 *Soient* $(\mathcal{X}, d)$ *un espace métrique complet et* T *une fonction contractante sur* $\mathcal{X}$*. Alors :*

$$\forall p \in \mathcal{X} \quad \lim_{n\to\infty} T^n(p) = c,$$

où c *désigne le point fixe de* T*. Donc, toute suite définie par*

$$(p_n)_{n\in\mathbb{N}} = \begin{cases} p_0 = p \in \mathcal{X} \\ p_{n+1} = T(p_n) \end{cases}$$

converge vers c*.*

Cette propriété est utilisée dans l'espace des compacts $(\mathcal{H}(\mathcal{X}), d_H)$ avec l'opérateur de HUTCHINSON et permet d'obtenir le théorème suivant :

Théorème 1.2.1 *Soient $(\mathcal{X}, d)$ un espace métrique complet et $\mathcal{T}$ un IFS. Toute suite définie par*

$$(K_n)_{n\in\mathbb{N}} = \begin{cases} K_0 \in \mathcal{H}(\mathcal{X}) \\ K_{n+1} = H(K_n) \end{cases}$$

converge vers le point fixe K de H et donc vers l'attracteur de l'IFS $\mathcal{T}$.

BARNSLEY appelle cet algorithme *l'algorithme déterministe.*

Exemple : Dans le cas de l'IFS $\mathcal{T}_{arbre}$, nous construisons, par exemple, la suite

$$(K_n)_{n\in\mathbb{N}} = \begin{cases} K_0 = S(O, 1) \\ K_{n+1} = H_{arbre}(K_n) \end{cases},$$

avec $S(O, 1)$ la sphère de centre O et de rayon 1. Les premiers termes de la suite sont donnés Figure 1.2.

Remarque : Trois méthodes de visualisation ont été décrites par BARNSLEY :

1. Par construction d'une suite croissante : cette méthode permet d'approximer la fractale par l'intérieur en remplissant l'attracteur. Cette méthode est appelée méthode du *chaos dynamique* ou *algorithme non déterministe.*

2. Par construction d'une suite décroissante : cette méthode permet d'approximer la fractale par l'extérieur.

3. Par échappement : cette méthode permet de visualiser le complémentaire de la fractale.

On peut trouver d'autres méthodes de visualisation dans [PS88] et [HPS91]. Nous avons travaillé uniquement avec l'algorithme déterministe car les résultats obtenus sont facilement exploitables en lancer de rayons.

1.3 Manipulations

Les auteurs qui ont travaillé sur les IFS se sont intéressés essentiellement à la visualisation, et donc aux divers algorithmes de construction. Le

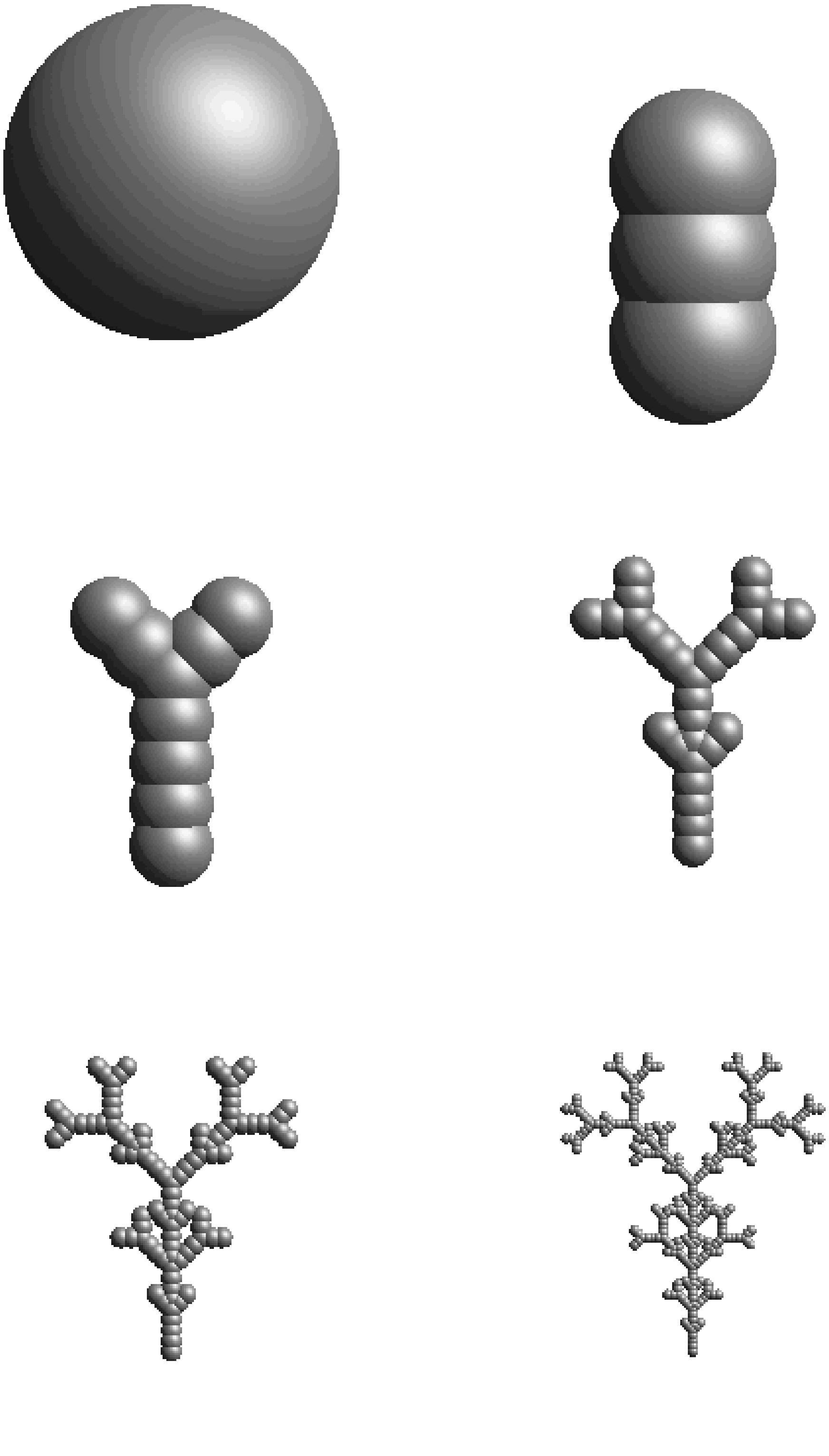


FIG. 1.2 - *Algorithme déterministe.*

problème posé par les IFS, dans le cadre de la synthèse d'image, ne se résume pourtant pas à leur visualisation. En effet, la difficulté de contrôle de l'attracteur est un autre aspect que nous devons prendre en compte.

Quelques résultats ont été trouvés dans ce cadre : le *théorème du collage*, qui propose une méthode pour trouver les transformations associées à un attracteur donné [Bar88], des théorèmes de continuité, garantissant que deux IFS proches ont des attracteurs proches [Bar88][Gen92][Vrs91][DLVM95], des théorèmes sur les englobants de l'IFS, garantissant par exemple que l'attracteur reste bien dans le cadre de l'écran, des opérations permettant de combiner deux IFS [Gen92] et des IFS à pôles [ZT94][ZT96].

Nous allons présenter les opérations particulières au modèle IFS, car notre démarche est une extension de cette approche. Pour les démonstrations, le lecteur pourra se référer à [Gen92]. Ces opérations serviront à fabriquer des objets fractals de manière constructive.

On notera $\mathcal{T} = (T_1, T_2, \ldots, T_N)$ et $\mathcal{T}' = (T'_1, T'_2, \ldots, T'_M)$ deux IFS quelconques, et $\mathcal{T}'' = (T''_1, T''_2, \ldots, T''_L)$ l'IFS résultat de l'opération que nous étudions.

1.3.1 Transformation

On définit le conjugué d'un IFS $\mathcal{T}$ par une transformation inversible R de la facon suivante :

$$T''_i = R \circ T_i \circ R^{-1}.$$

Proposition 1.3.1 *l'attracteur correspondant au nouvel IFS $\mathcal{T}''$ sera le transformé du premier par R :*

$$\mathcal{A}(R \circ \mathcal{T} \circ R^{-1}) = R \circ \mathcal{A}(\mathcal{T}).$$

1.3.2 Symétrisation

Soit un groupe fini de transformations $\mathcal{G} = \{R_0, \ldots R_l\}$. On construit l'IFS suivant :

$$T''_{ji} = R_j \circ T_i \circ R_j^{-1}.$$

On a alors :

Proposition 1.3.2 *L'attracteur de $\mathcal{T}$" est invariant par $\mathcal{G}$:*

$$\forall R \in \mathcal{G} : \ R \circ \mathcal{A}(\mathcal{T}'') = \mathcal{A}(\mathcal{T}'').$$

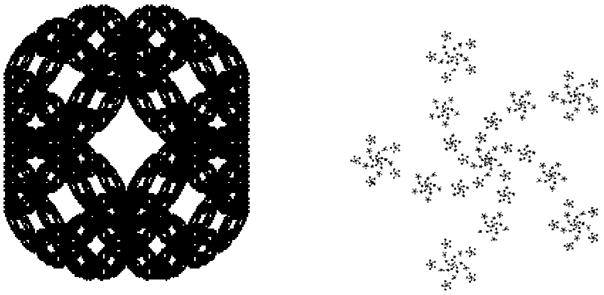


FIG. 1.3 - *Exemples d'IFS à symétries.*

On peut ainsi obtenir des attracteurs possédant des symétries.

Exemple : La Figure 1.3 montre, à gauche, une fractale obtenue en imposant une symétrie de réflexion par rapport à un axe horizontal et, à droite, une fractale obtenue en imposant une symétrie de rotation de $\frac{2\pi}{5}$.

1.3.3 Extrusion

Soient $\mathcal{T}$ et $\mathcal{T}'$ deux IFS opérant respectivement dans le plan (Oxy) et le plan (Oyz), tels que :

$$T_i = S_i \circ D,$$

$$T_i' = S_i' \circ D,$$

avec S_i un cisaillement de Ox par Oy, S_j' un cisaillement de Oy par Oz et D une dilatation selon Oz.

On peut effectuer une extrusion par balayage de $\mathcal{A}(\mathcal{T})$ le long de $\mathcal{A}(\mathcal{T}')$ en définissant un "produit" $\mathcal{T}''$ de la manière suivante :

$$T_{ij}'' = S_i \circ S_j' \circ D.$$

Exemple : La Figure 1.4 montre deux exemples d'extrusion : à gauche, extrusion continue du triangle de SIERPINSKI suivant la courbe de TAKAGI; à droite, construction d'une surface fractale par balayage d'une courbe de TAKAGI le long d'une autre courbe de TAKAGI (au fond à droite se trouve l'attracteur obtenu par composition directe des deux IFS).

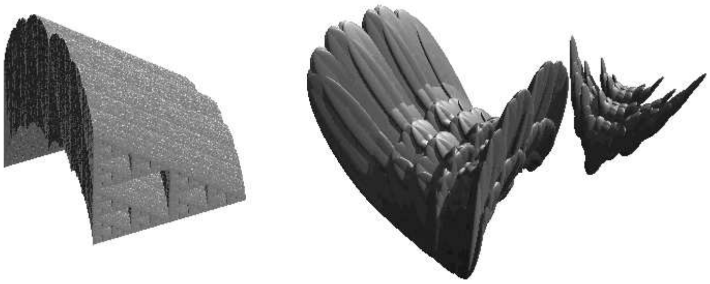


FIG. 1.4 - *Exemples d'extrusions.*

1.3.4 Interpolation

Soit la combinaison convexe de deux transformations définie par :

$$\forall p \in \mathcal{X} \quad T'' \circ p = (1-\lambda) T \circ p + \lambda T' \circ p \quad 0 \leq \lambda \leq 1.$$

On étend cette combinaison aux IFS possédant le même nombre de transformations de la manière suivante :

$$\begin{aligned} \mathcal{T}''_\lambda &= (T''_i), \\ T''_i &= (1-\lambda) T_i + \lambda T'_i. \end{aligned}$$

On a alors :

Proposition 1.3.3 *La famille d'attracteurs $\mathcal{A}(\mathcal{T}''_\lambda)$ est continue par rapport à λ.*

La combinaison convexe de deux IFS permet de réaliser des séquences de fractales passant continûment de l'attracteur du premier à l'attracteur du deuxième. L'interpolation entre fractales a été illustrée par HART dans [Har91].

Exemple : La Figure 1.5 donne un exemple d'interpolation qui passe d'un tétraèdre de SIERPINSKI à un arbre.

1.3.5 Limites de cette approche

Les opérations que nous venons de présenter permettent de combiner deux attracteurs d'IFS et de retrouver des opérations de modélisation géométrique : extrusion, interpolation, symétrisation.

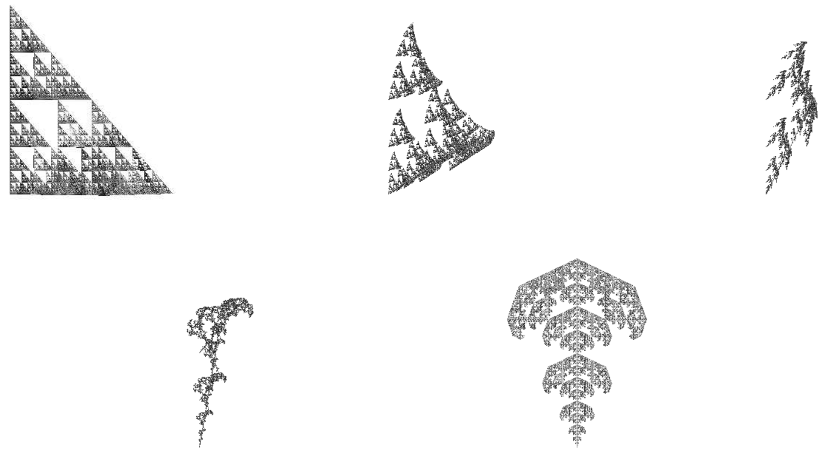


FIG. 1.5 - *Exemple d'interpolation.*

Ces opérations permettent de construire des objets complexes mais elles ne permettent pas une construction pas à pas. La géométrie constructive du solide (CSG) a été introduite dans ce but. Nous nous intéressons par conséquent à une extension de cette approche pour la géométrie fractale. Pour cela nous aurons besoin d'opérations plus simples (l'union, par exemple) que celles définies par CHRISTIAN GENTIL. Cependant, le modèle IFS ne permet pas de définir une telle opération car l'union de deux attracteurs d'IFS ne peut en général pas être définie comme un attracteur d'IFS.

Nous aurons donc besoin d'un modèle plus général que le modèle IFS dans lequel nous puissions effectuer des opérations simples permettant une construction progressive de l'objet fractal.

1.4 Extensions du modèle IFS

Plusieurs auteurs se sont intéressés à des modèles généralisant les IFS. La plupart de ces modèles consistent à associer un langage à un IFS. Les lettres du langage correspondent aux transformations de l'IFS. Les mots du langage correspondent à des compositions de transformations. Ces modèles permettent de construire des sous-ensembles de l'attracteur d'IFS en ne considérant que les mots du langage au lieu de toutes les compositions possibles de transformations, comme c'est cas dans les IFS.

Ces modèles sont fondés sur des représentations différentes suivant les auteurs :

- Berstel, Morcrette et Nait Abdallah [BM89][BNA89] ont travaillé sur des tétrarbres (ou *quadtrees*) engendrés par automates finis, généralisant ainsi l'IFS composé des quatre homothéties associées à un carré. Ils ont développé le lien entre automates finis et formulation par morphismes itérés, également étudié par Shallit et Stolfi [SS89].
- Barnsley [BHH89] a introduit le modèle RIFS (Recurrent IFS) aussi utilisé par Hart [Har92]. Ce modèle est donné par un IFS et un graphe $G = (V, E)$. Son attracteur est défini comme l'union des solutions A_i du système d'équations :

$$A_j = \bigcup_{(i,j)\in E} T_j(A_i).$$

- Mauldin et Williams [MW88] ont introduit un modèle fondé sur un graphe $G = (V, E)$. Son attracteur est défini comme l'union des solutions du système d'équations

$$X_i = \bigcup_{e=(i,j)\in E} T_e \circ Xj.$$

- Bandt [Ban89] a introduit un modèle fondé sur un système d'équations ("sofic sytem") défini par

$$X_i = \bigcup T_k \circ X_j \quad \text{avec } i, j \in \{1, \ldots n\} \ k \in \{1, \ldots, N\}.$$

  Son attracteur est l'union des solutions du système.
- Culik et Dube ont défini plusieurs modèles équivalents :
  - un PAA (Probabilist Affine Automata) [CD93b] est un automate dont chaque transition est étiquetée par une transformation affine et une probabilité.
  - un MRIFS (Mutually Recursive IFS) [CD93b] défini par un système d'équations du type

$$A = T_1(B_1) \cup T_2(B_2) \cup \ldots T_n(B_n).$$

  - un "controlled" MRFS [CD93a] défini comme un MRIFS dont les équations sont étiquetées. Un langage sur ces étiquettes permet alors de contrôler l'ordre dans lequel sont appliquées les équations.

- PEITGEN, JÜRGENS et SAUPE [PJS92] ont introduit une matrice opérateur de HUTCHINSON dont chaque élément est une union de transformations. L'attracteur de ce modèle sera cette fois un vecteur d'images. Nous avons présenté dans [TT93a] [TT93b] une approche différente de ce modèle. Cette approche est reprise dans cette thèse chapitre 3 avec un formalisme différent.

- PRUSINKIEWICZ [PH92] définit le modèle LRIFS (Language-restricted IFS) comme étant un IFS auquel on associe un langage dans le but de restreindre la suite des transformations que l'on applique. Nous avons présenté dans [TT95a] [TT95b] une définition différente de l'attracteur d'un LRIFS. Ces articles seront détaillés au chapitre 2.

Les modèles, autres que le modèle LRIFS, sont fondés sur des langages rationnels (automates, grammaires, expressions rationnelles) ou sur des représentations de type matriciel (graphes, systèmes d'équations, matrices). Nous nous sommes donc intéressés au modèle LRIFS pour sa généralité. Cependant la définition de l'attracteur donnée par PRUSINKIEWICZ impose au langage d'être postfixé. Ceci est contraignant dans la mesure où, par exemple, l'union de deux langages postfixés n'est pas en général un langage postfixé.

La multiplicité des autres modèles rend le choix d'une extension particulièrement difficile. Ceci est d'autant plus vrai que la définition de l'attracteur varie suivant les auteurs, chaque modèle ayant ses particularités propres. Parmis ces définition, l'approche matricielle proposée par PEITGEN, JÜRGENS et SAUPE nous a particulièrement intéressés, car il est plus naturel de définir des opérations sur des matrices que sur des graphes ou des sytèmes d'équations. Cependant, elle ne permet pas de retrouver toutes les autres approches, en particulier celles fondées sur les langages.

On peut noter que d'autres modèles liés à la théorie des langages ont été proposés. Ils ont été référencés dans [Tho93]. On peut distinguer deux grandes classes de modèles : les extensions du modéle IFS que nous venons de présenter et les modèles fondés sur les règles de réécriture, tels que les L-systems proposés par LINDENMAYER [PLH88] [PH90] [Pru86] [CG93] ou les *recurrent sets* proposés par DEKKING [Dek80] [Dek82] [Dek87]. Les liens entre ces deux classes ont été étudiés dans [CD92] et [Mor96].

Étude des modèles de généralisation d'IFS

Nous avons vu, dans le chapitre précédent, qu'aucune des extensions proposées ne convenait parfaitement à notre propos. Notre but a donc été de proposer un modèle suffisamment général, assurant que la combinaison de deux attracteurs reste à l'intérieur du modèle. D'autre part, ce modèle doit être facilement implémentable et englober les modèles existants.

Comme il semble difficile de proposer un modèle à la fois très général et facilement implémentable, nous proposons deux modèles. Pour cela, nous allons reprendre le modèle LRIFS et le modèle de Peitgen, Jürgens et Saupe en proposant dans les deux cas une nouvelle définition de l'attracteur.

Dans le cas du modèle LRIFS, ceci permet de lever la restriction sur le langage dont nous avons parlé précédemment. Dans le cas matriciel, nous obtenons un modèle facilement implémentable, permettant de manipuler des opérations, et qui nous permet de retrouver tous les attracteurs des modèles aussi bien fondés sur les langages rationnels que sur les représentations matricielles. De plus, les deux modèles que nous proposons sont équivalents dans le cas des langages rationnels.

Chapitre 2

IFS et langages

Nous présentons dans ce chapitre une première extension du modèle IFS. Ce modèle est une généralisation du modèle LRIFS introduit par PRUSINKIEWICZ [PH91][PH92]. Nous conserverons le terme de LRIFS car seule la définition de l'attracteur est différente.

Notre approche est fondée sur l'étude de la suite des ensembles de préfixes de longueur n d'un langage donné. En effet, lorsque l'on interprète les lettres comme des transformations contractantes, et les mots comme des compositions de transformations, l'application de cette suite à un compact converge vers un unique compact que nous appellerons *l'attracteur associé au langage*. Le lecteur pourra se reporter à l'annexe A pour les notations et définitions liées à la Théorie des Langages.

2.1 Théorie des Langages et IFS

On définit un IFS comme un N-uplet de fonctions. On peut donc associer à chaque transformation une lettre d'un alphabet.

Définition 2.1.1 *Soit un IFS* $\mathcal{T} = (T_1, \ldots, T_N)$.

$\Sigma = \{1 \ldots N\}$ *sera appelé l'alphabet associé à* $\mathcal{T}$.

On définit ensuite une fonction d'étiquetage qui permet de mettre en bijection transformations et lettres.

Définition 2.1.2 *On appellera fonction d'étiquetage la fonction :*

$$\begin{array}{rrcl} h: & \Sigma & \longrightarrow & \mathcal{T} \\ & i & \longmapsto & T_i \end{array}.$$

La fonction h est étendue aux langages de la façon suivante : soit ϵ le mot vide, $u = u_1u_2\ldots u_k$ un mot de Σ^* et L un langage sur Σ; on pose

$$\begin{array}{rcl} h(\epsilon) & = & \mathrm{Id}; \\ h(u) & = & T_{u_1} \circ T_{u_2} \circ \ldots T_{u_k}; \\ h(L) & = & \{h(w)/w \in L\}; \\ h(\Sigma^*) & = & \{h(w)/w \in \Sigma^*\} = \mathcal{T}^*. \end{array}$$

On définit aussi une fonction d'adressage qui, à un mot infini sur Σ, associe un point de $\mathcal{X}$.

Théorème 2.1.1 ([Bar88]) *Soit $\sigma = \sigma_1\sigma_2\sigma_3\ldots \in \Sigma^\omega$. Pour tout $p \in \mathcal{X}$,*

$$\phi(\sigma) = \lim_{j\to\infty} h(\sigma_1) \circ h(\sigma_2) \circ \ldots h(\sigma_j) \circ p$$

existe et est indépendant de p.

Cette fonction est étendue aux langages de mots infinis de la façon suivante : soit L un langage de mots infinis, on pose

$$\phi(L) = \{\phi(\sigma)/\sigma \in L\}.$$

Nous pouvons maintenant définir un LRIFS :

Définition 2.1.3 *Un LRIFS est un quadruplet $\mathcal{I} = (\mathcal{T}, \Sigma, h, L)$, avec*

- *$\mathcal{T}$ un IFS;*
- *Σ l'alphabet associé à $\mathcal{T}$;*
- *h la fonction d'étiquetage correspondante;*
- *L un langage sur Σ^*.*

2.2 Attracteur associé à un LRIFS

Pour définir l'attracteur associé à un LRIFS nous définissons une suite dont la limite est l'attracteur. Cette suite est définie comme la suite des ensembles de préfixes de longueur n du langage appliqués à un compact. Pour prouver que cette suite converge, nous prouvons qu'elle est de CAUCHY. Pour cela nous aurons besoin des lemmes suivants :

Lemme 2.2.1 (IFS à condensation) *Soit* $\mathcal{T} = \{T_1, \ldots, T_N\}$ *un IFS et* T_0 *une transformation constante, c'est-à-dire telle que*

$$\forall K \in \mathcal{H}(\mathcal{X}),\ T_0 \circ K = K_0,$$

avec K_0 *un compact donné.*

La fonction

$$\begin{array}{rcl} F: \mathcal{H}(\mathcal{X}) & \longrightarrow & \mathcal{H}(\mathcal{X}) \\ K & \longmapsto & (T_0 \cup \mathcal{T}) \circ K = K_0 \cup T_1 \circ K \cup T_2 \circ K \cup \ldots \cup T_N \circ K \end{array}$$

est contractante sur $\mathcal{H}(\mathcal{X})$ *[Bar88]. Donc elle admet un unique point fixe :*

$$\mathcal{A} = F \circ \mathcal{A}.$$

Selon le corollaire du Théorème du Point Fixe donné en 1.2.1

$$\begin{aligned} \mathcal{A} &= \lim_{n\to\infty} F^n \circ K_0 \\ &= \lim_{n\to\infty} K_0 \cup \mathcal{T} \circ K_0 \cup \ldots \cup \mathcal{T}^n \circ K_0 \\ &= \lim_{n\to\infty} \bigcup_{j=0}^{n} \mathcal{T}^j \circ K_0 \end{aligned}$$

Lemme 2.2.2 *Pour tout* $A_i, B_i, 1 \leq i \leq N$ *appartenant à* $\mathcal{H}(\mathcal{X})$

$$d_H(\bigcup_{i=1}^{N} A_i, \bigcup_{i=1}^{N} B_i) \leq \max_{1\leq i\leq N} d_H(A_i, B_i).$$

Démonstration : Voir [Bar88].

Lemme 2.2.3 *Soit* Σ *un alphabet. Soit* E_n *un ensemble de mots de longueur* n *sur* Σ^**, alors*

$$\forall K \in \mathcal{H}(\mathcal{X}) \quad \exists C(K) \text{ tel que } \forall n \ d_H(h(E_n) \circ K, K) < C(K),$$

avec $C(K)$ *une constante dépendant de* K*.*

Démonstration :

$$\begin{array}{rcl}\forall n \quad E_n & \subseteq & \Sigma^n, \\ \forall n \quad h(E_n) \circ K & \subseteq & h(\Sigma^n) \circ K \\ & \subseteq & \displaystyle\bigcup_{j=0}^{n} h(\Sigma^j) \circ K \\ & \subseteq & \displaystyle\lim_{n\to\infty} \bigcup_{j=0}^{n} h(\Sigma^j) \circ K.\end{array}$$

En notant $\lim_{n\to\infty} \bigcup_{j=0}^{n} f(j) = \bigcup_{n=0}^{\infty} f(n)$ on a :

$$\forall n \quad h(E_n) \circ K \subseteq \bigcup_{n=0}^{\infty} h(\Sigma^n) \circ K = K'.$$

K' est l'attracteur de l'IFS $h(\Sigma)$ avec l'ensemble de condensation K. Donc, en utilisant le Lemme 2.2.1, K' est un ensemble compact et donc borné. Soit $C(K)$ le diamètre de K' alors

$$\forall K \in \mathcal{H}(\mathcal{X}) \quad d_H(h(E_n) \circ K, K) < C(K).$$

□

Lemme 2.2.4 *Soit L un langage infini sur Σ. Soit L_n l'ensemble des préfixes de longueur n de L. En notant*

$$E_{n,p,v} = \{w \in \Sigma^* | vw \in L_{n+p}\},$$

on a

$$L_{n+p} = \bigcup_{v\in L_n} vE_{n,p,v}.$$

Démonstration : Les mots de L_{n+p} peuvent s'écrire vw avec $v \in L_n$ et $w \in \Sigma^p$. □

Proposition 2.2.1 *Soit $\mathcal{I} = (\mathcal{T}, \Sigma, h, L)$ un LRIFS. Soit L_n l'ensemble des préfixes de longueur n de L. Soit K un compact de $\mathcal{H}(\mathcal{X})$. Alors la suite $(h(L_n) \circ K)_{n\in\mathbb{N}}$ est une suite de* CAUCHY.

Démonstration :

Nous voulons prouver que

$$\forall \epsilon > 0 \;\; \exists n_0, \text{ tel que } (m > n > n_0) \Rightarrow (d_H(h(L_n) \circ K, h(L_m) \circ K) < \epsilon).$$

En posant $p = m - n$, on peut récrire

$$d_H(h(L_n) \circ K, h(L_m) \circ K) \;\; = \;\; d_H(h(L_n) \circ K, h(L_{n+p}) \circ K).$$

Selon le Lemme 2.2.4 on a :

$$d_H(h(L_n) \circ K, h(L_m) \circ K) = d_H(\bigcup_{u \in L_n} h(u) \circ K, \bigcup_{v \in L_n} h(v)h(E_{n,p,v}) \circ K).$$

L_n étant fini, nous pouvons utiliser le Lemme 2.2.2 :

$$d_H(h(L_n) \circ K, h(L_m) \circ K) \leq \max_{u \in L_n} d_H(h(u) \circ K, h(u)h(E_{n,p,u}) \circ K).$$

Notons $s = \max s_i$, avec s_i le rapport de contraction de la transformation T_i. Comme u est un mot de longueur n, $h(u)$ est un opérateur contractant et son rapport de contraction est inférieur à s^n. Donc,

$$d_H(h(L_n) \circ K, h(L_m) \circ K) \leq s^n d_H(K, h(E_{n,p,u}) \circ K).$$

Selon le Lemme 2.2.3, on a

$$d_H(h(L_n) \circ K, h(L_m) \circ K) \leq s^n C(K).$$

$s < 1$ donc $(s^n C(K))_{n \in \mathbb{N}}$ converge vers 0, c'est à dire

$$\forall \epsilon > 0 \; \exists n_0, n > n_0 \Rightarrow s^n C(K) < \epsilon.$$

Donc

$$\forall \epsilon > 0 \; \exists n_0 \text{ tel que } (m > n > n_0) \Rightarrow (d_H(h(L_n) \circ K, h(L_m) \circ K) < \epsilon).$$

□

La suite $(h(L_n) \circ K)_{n \in \mathbb{N}}$ est une suite de Cauchy, elle possède donc une limite. De plus, cette limite est indépendante de K, car on a

$$d_H(h(L_n) \circ K, h(L_n) \circ K') \leq s^n d_H(K, K'),$$

avec $s < 1$. Cette limite sera l'attracteur associé au langage.

Définition 2.2.1 (Attracteur) *Soit $\mathcal{I} = (\mathcal{T}, \Sigma, h, L)$ un LRIFS. L'attracteur associé à $\mathcal{I}$ sera défini par :*

$$\mathcal{A}(L) = \lim_{n \to \infty} h(L_n) \circ K,$$

pour tout $K \in \mathcal{H}(\mathcal{X})$.

Exemple : On reprend l'exemple de l'IFS $\mathcal{T}_{arbre} = \{T_1, T_2, T_3, T_4\}$. On a alors l'alphabet associé $\Sigma_{arbre} = \{1, 2, 3, 4\}$. Si l'on associe à $\mathcal{T}_{arbre}$ le langage $L_{arbre} = \{3, 4\}^*\{1, 2\}^*$, on a alors le LRIFS

$$\mathcal{I}_{arbre} = (\mathcal{T}_{arbre}, \Sigma_{arbre}, h, L_{arbre}).$$

Comme

$$L_n = \bigcup_{i+j=n} \{3, 4\}^i \{1, 2\}^j$$

alors

$$\begin{aligned} \mathcal{A}(L_{arbre}) &= \lim_{n \to \infty} h(L_n) \circ K \\ &= h(\{3, 4\}^\omega \cup \{3, 4\}^* \{1, 2\}^\omega) \circ K \\ &= \mathcal{A}(\{T_3, T_4\}) \cup \bigcup_{i \in \mathbb{N}} \{T_3, T_4\}^i \circ \mathcal{A}(\{T_1, T_2\}). \end{aligned}$$

On obtient l'attracteur représenté figure 2.1. On le notera $\tilde{A}_{arbre}$ et on l'appelera l'arbre élagué.

2.3 Attracteur dans les compacts de mots

Une définition équivalente de l'attracteur a été donnée par Culik [CD93b] dans le cas des expressions rationnelles. Elle est, en fait, valable pour tous les langages. Cette définition utilise l'adhérence d'un langage infini; nous rappelons ici la définition et le lecteur pourra se référer à l'Annexe A pour plus de précisions.

Définition 2.3.1 (Adhérence) *Un mot infini σ, tel que $\mathrm{Pref}(\sigma) \subseteq \mathrm{Pref}(L)$, est appelé une adhérence de L. L'ensemble des adhérences de L est noté $\mathrm{ls}(L)$. On a donc*

$$\begin{aligned} \mathrm{ls}(L) &= \{\sigma \in \Sigma^\omega / \mathrm{Pref}(\sigma) \subseteq \mathrm{Pref}(L)\} \\ &= \{\sigma \in \Sigma^\omega / \sigma_1 \sigma_2 \ldots \ \sigma_n \in L_n\}. \end{aligned}$$

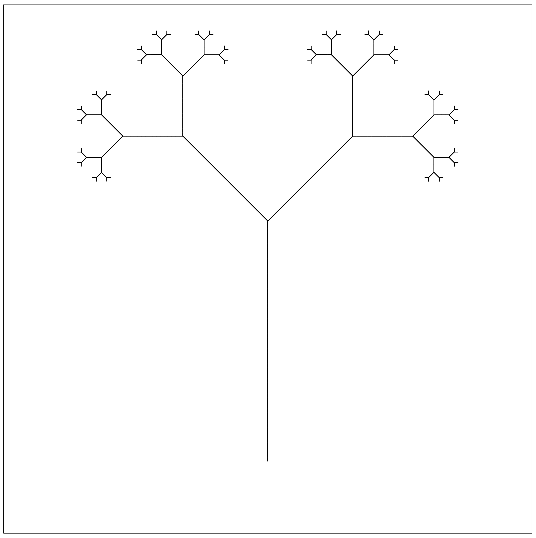


FIG. 2.1 - *Attracteur associé à L_{arbre}.*

Définition 2.3.2 (Attracteur) *Soit $\mathcal{I} = (\mathcal{T}, \Sigma, h, L)$ un LRIFS. L'attracteur associé à L sera donné par :*

$$\phi(\mathrm{ls}(L)).$$

En effet, on a :

Proposition 2.3.1 *Soit L un langage infini, alors*

$$\lim_{n\to\infty} h(L_n) \circ K = \phi(\mathrm{ls}(L)).$$

Démonstration :

$$\begin{aligned}
\mathrm{ls}(L) &= \{\sigma \in \Sigma^\omega / \mathrm{Pref}(\sigma) \subseteq \mathrm{Pref}(L)\} \\
&= \{\sigma \in \Sigma^\omega / \forall n \ \sigma_1\sigma_2\ldots\sigma_n \in L_n\}; \\
\phi(\mathrm{ls}(L)) &= \{\phi(\sigma) / \forall n \ \sigma_1\sigma_2\ldots\sigma_n \in L_n\} \\
&= \{\lim_{n\to\infty} h(\sigma_1)\circ h(\sigma_2)\circ\ldots h(\sigma_n)\circ p / \forall n \ \sigma_1\sigma_2\ldots\sigma_n \in L_n\} \\
&= \{\lim_{n\to\infty} h(\sigma_1)\circ h(\sigma_2)\circ\ldots h(\sigma_n)\circ p / \forall n \ h(\sigma_1)\circ h(\sigma_2)\circ\ldots h(\sigma_n) \in h(L_n)\} \\
&= \lim_{n\to\infty} h(L_n)\circ\{p\}.
\end{aligned}$$

□

2.4 Lien avec la définition de Prusinkiewicz

Prusinkiewicz définit l'attracteur associé à $\mathcal{I}$ de la manière suivante :

Définition 2.4.1 *On définit*

$$\mathcal{A}_L(p) = \mathrm{adh}(h(L) \circ p)\ \forall p \in \mathcal{X}$$

avec adh *l'adhérance au sens topologique.*

En général, $\mathcal{A}_L(p)$ dépend du choix de p, mais le théorème suivant [PH92] permet de définir un attracteur associé à un langage :

Théorème 2.4.1 *Si L est tel que $La \subset L$, alors il existe un plus petit ensemble $\mathcal{A}_L$ défini par :*

$$\mathcal{A}_L = \mathrm{adh}(h(L) \circ p_0),$$

avec p_0 le point fixe de la transformation $h(a)$.

Remarque : La restriction des langages aux langages postfixés pose un problème lorsque l'on veut faire des opérations sur les LRIFS. En effet, l'union de deux langages postfixés n'est en général pas postfixée. Par exemple, si l'on prend $L_1 = \{a^*\}$ et $L_2 = \{b^*\}$, alors $L_1 \cup L_2 = \{a^*\} \cup \{b^*\}$ n'est pas postfixé.

Cette définition est équivalente à celle que nous avons proposée lorsque le langage est postfixé. Afin de prouver cette équivalence, nous avons besoin du lemme suivant :

Lemme 2.4.1 *Soit (K_n) une suite de* Cauchy *dans $\mathcal{H}(\mathcal{X})$. Si $\forall i \quad K_i \subseteq \lim_{n\to\infty} K_n$ alors*

$$\mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n) = \lim_{n\to\infty} K_n.$$

Démonstration :

– $\mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n) \subseteq \lim_{n\to\infty} K_n$:

$$\begin{aligned} \forall n \quad K_n &\subseteq \lim_{n\to\infty} K_n, \\ \bigcup_{n\in\mathbb{N}} K_n &\subseteq \lim_{n\to\infty} K_n, \\ \mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n) &\subseteq \lim_{n\to\infty} K_n. \end{aligned}$$

– $\lim_{n\to\infty} K_n \subseteq \mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n)$:

$$\mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n) \subseteq \lim_{n\to\infty} K_n.$$

$\mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n)$ est fermé et borné, c'est donc un compact. Ainsi, c'est un espace métrique complet, donc

$$K_n \in \mathcal{H}(\mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n)) \Rightarrow \lim_{n\to\infty} K_n \subseteq \mathrm{adh}(\bigcup_{n\in\mathbb{N}} K_n).$$

□

On peut alors prouver l'équivalence entre les deux définitions :

Proposition 2.4.1 *Si le langage L est postfixé (i.e. $La \subseteq L$) alors*

$$\mathcal{A}_L = \mathcal{A}(L).$$

Démonstration :

Soit p_0 le point fixe de la transformation $h(a)$. Le Lemme 2.4.1 nous permet de prouver la proposition. En effet $(h(L_n) \circ \{p_0\})_{n\in\mathbb{N}}$ est une suite de Cauchy et

$$\begin{aligned} h(L_n) \circ p_0 &= h(L_n) \circ h(a) \circ p_0 \\ &\subseteq h(L_{n+1}) \circ p_0 \\ &\subseteq \lim_{n\to\infty} h(L_n) \circ p_0 \\ &\subseteq \mathcal{A}(L). \end{aligned}$$

Alors

$$\mathrm{adh}(\bigcup_{n\in\mathbb{N}} h(L_n) \circ \{p_0\}) = \lim_{n\to\infty} h(L_n) \circ \{p_0\}.$$

Donc

$$\mathcal{A}_L(p_0) = \mathcal{A}_L = \mathcal{A}(L).$$

□

Remarque : La définition de Prusinkiewicz se rapproche de celle des IFS à condensation. Notre définition est plus générale dans le sens où elle ne pose aucune restriction sur le langage, en revanche elle ne permet pas d'appliquer l'algorithme de l'échappement, tel que l'a donné Prusinkiewicz.

2.5 Conclusion

Nous avons défini un modèle généralisant le modèle IFS, ne nécessitant aucune restriction sur le type de langage. Ce modèle nous permettra de définir des combinaisons d'attracteurs à partir d'opérations sur les langages. En effet, si l'on considère une opération $\odot$ sur les langages, nous construirons l'attracteur

$$\mathcal{A}(L \odot L').$$

D'autre part, ce modèle permet de représenter toutes les figures dessinables sur un écran. En effet, il permet d'atteindre tous les compacts de $[0,1]^2$. Il suffit de considérer l'IFS composé des transformations du tétrarbre. Chaque point d'un compact de $[0,1]^2$ pourra être adressé par un mot sur l'alphabet du tétrarbre. On obtient ainsi un langage infini, qui n'est autre que le langage des branches du tétrarbre.

Chapitre 3

IFS et matrices

Le modèle présenté au chapitre précédent est très général. Cependant, il n'est pas facilement implémentable, car les langages quelconques ne se manipulent pas simplement d'un point de vue algorithmique. La plupart des auteurs se sont donc restreints aux langages rationnels ou à des formalismes de type matriciel (graphe, matrice ou système d'équations). Nous avons cherché un modèle qui permette d'unifier ces approches, de visualiser les attracteurs, tout en étant équivalent au modèle précédent dans le cas d'un langage rationnel.

Nous avons choisi le formalisme matriciel car il convient bien à une approche constructive. Cette approche va nous permettre, de plus, de mettre en évidence les équivalences entre les modèles qui ont été proposés, et de visualiser les attracteurs par l'algorithme déterministe.

3.1 Matrice d'attracteurs

Notre approche est fondée sur une généralisation de l'opérateur de HUTCHINSON, proposée par PEITGEN, JÜRGENS et SAUPE [PJS92]. Cet opérateur permet de construire une matrice d'attracteurs.

3.1.1 Espace des matrices de compacts

Nous allons définir des matrices ayant des compacts pour éléments. Comme la distance de HAUSDORFF n'est définie que pour un couple de compacts non vides, il n'est pas possible d'en déduire directement une distance entre matrices, sans tenir compte de leurs composantes vides.

On notera $M_R(\mathcal{H}(\mathcal{X}))$ l'ensemble des matrices de compacts de $\mathcal{X}$ dont les éléments non vides sont indicés par $R \subseteq \{1, \ldots n\} \times \{1, \ldots n\}$. C'est-à-dire que pour toute matrice $\tilde{K} \in M_R(\mathcal{H}(\mathcal{X}))$,

$$\tilde{K}_{ij} \in \mathcal{H}(\mathcal{X}) \cup \emptyset,$$

$$\tilde{K}_{ij} \neq \emptyset \Leftrightarrow (i,j) \in R.$$

On définit la distance suivante sur $M_R(\mathcal{H}(\mathcal{X}))$:

Définition 3.1.1 *Pour $\tilde{K}$ et $\tilde{K}' \in M_R(\mathcal{H}(\mathcal{X}))$,*

$$d_\infty(\tilde{K}, \tilde{K}') = \max_{(i,j)\in R} d_H(\tilde{K}_{ij}, \tilde{K}'_{ij}).$$

Proposition 3.1.1 *d_∞ est une distance sur les matrices de compacts.*

Démonstration : d_∞ est déduite de d_H de la manière usuelle et vérifie trivialement les axiomes d'une distance. □

Comme R est fini, on a

Proposition 3.1.2 *L'espace $(M_R(\mathcal{H}(\mathcal{X})), d_\infty)$ est un espace métrique complet.*

3.1.2 Matrice de Hutchinson

Nous allons définir une matrice dont les éléments sont des opérateurs de Hutchinson. Pour simplifier, nous appellerons cette matrice "une matrice de Hutchinson".

Définition 3.1.2 *Soit un $\mathcal{S}$ un sous-semigroupe de transformations affines. On appellera matrice de* Hutchinson *toute matrice $n \times n$ dont les éléments sont des ensembles finis de transformations de $\mathcal{S}$. Ces ensembles sont éventuellement vides.*

$$\tilde{H} = (\tilde{H}_{ij});$$

$$\tilde{H}_{ij} = \begin{cases} \bigcup_{k\in I_{ij}} T_k \ si\ (i,j) \in R \\ \emptyset \ sinon \end{cases},$$

avec $I_{ij} \subseteq [1 \ldots N]$ et $T_k \in \mathcal{S}$. On notera $\mathcal{P}_+(\mathcal{S})$, l'ensemble des parties finies de $\mathcal{S}$, et $M_R(\mathcal{P}_+(\mathcal{S}))$, l'ensemble des matrices de Hutchinson.

Exemple : Soit l'IFS $\mathcal{T}_{arbre}$ défini au chapitre 1. On choisit comme matrice de HUTCHINSON :

$$\tilde{H}_{arbre} = \begin{pmatrix} \{T_3, T_4\} & \{T_1, T_2\} \\ \emptyset & \{T_1, T_2\} \end{pmatrix}.$$

$\tilde{H}_{arbre} \in M_R(\mathcal{P}_+(\mathcal{S}))$ avec $R = \{(1,1), (1,2), (2,2)\}$.

Proposition 3.1.3 *Les matrices de* HUTCHINSON *agissent sur les matrices de compacts. En posant*

$$(\tilde{H} \circ \tilde{K})_{ij} = \bigcup_{k=1}^{n} \tilde{H}_{ik} \circ \tilde{K}_{kj},$$

on a

$$\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S})), \tilde{K} \in M_S(\mathcal{H}(\mathcal{X})) \Rightarrow \tilde{H} \circ \tilde{K} \in M_{R \circ S}(\mathcal{H}(\mathcal{X})),$$

avec $R \circ S = \bigcup_{(i,k)\in R,(k,j)\in S} (i,j)$.

Démonstration :

$$\begin{aligned} (\tilde{H} \circ \tilde{K})_{ij} &= \bigcup_{k} \tilde{H}_{ik} \circ \tilde{K}_{kj}; \\ (\tilde{H} \circ \tilde{K})_{ij} \neq \emptyset &\Leftrightarrow \exists k, \tilde{H}_{ik} \neq \emptyset \text{ et } \tilde{K}_{kj} \neq \emptyset \\ &\Leftrightarrow \exists k, (i,k) \in R \text{ et } (k,j) \in S \\ &\Leftrightarrow (i,j) \in R \circ S. \end{aligned}$$

□

Exemple :

$$\tilde{H}_{arbre} \circ \tilde{K} = \begin{pmatrix} \{T_3, T_4\} \circ \tilde{K}_{11} \cup \{T_1, T_2\} \circ \tilde{K}_{21} & \{T_3, T_4\} \circ \tilde{K}_{12} \cup \{T_1, T_2\} \circ \tilde{K}_{22} \\ \{T_1, T_2\} \circ \tilde{K}_{21} & \{T_1, T_2\} \circ \tilde{K}_{22} \end{pmatrix}.$$

Proposition 3.1.4 *Les matrices de* HUTCHINSON *se composent entre elles. En posant*

$$(\tilde{H} \circ \tilde{H}')_{ij} = \bigcup_{k=1}^{n} \tilde{H}_{ik} \circ \tilde{H}'_{kj},$$

on a

$$\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S})), \tilde{H}' \in M_S(\mathcal{P}_+(\mathcal{S})) \Rightarrow \tilde{H} \circ \tilde{H}' \in M_{R \circ S}(\mathcal{P}_+(\mathcal{S})).$$

Démonstration : Même démonstration que la proposition précédente. □

Corollaire 3.1.1 *Si R est transitive ($R = R \circ R$), alors $M_R(\mathcal{P}_+(\mathcal{S}))$ agit sur $M_R(\mathcal{H}(\mathcal{X}))$, et $M_R(\mathcal{P}_+(\mathcal{S}))$ est stable par $\circ$. C'est-à-dire*

$$\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S})), \tilde{K} \in M_R(\mathcal{H}(\mathcal{X})) \Rightarrow \tilde{H} \circ \tilde{K} \in M_R(\mathcal{H}(\mathcal{X}));$$

$$\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S})), \tilde{H}' \in M_R(\mathcal{P}_+(\mathcal{S})) \Rightarrow \tilde{H} \circ \tilde{H}' \in M_R(\mathcal{H}(\mathcal{X})).$$

Proposition 3.1.5 *Soient $R \neq \emptyset$ une relation transitive et $\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S}))$ une matrice de* Hutchinson*. $\tilde{H}$ est contractante sur $(M_R(\mathcal{H}(\mathcal{X})), d_\infty)$, et de rapport de contraction $s_{\tilde{H}}$ tel que :*

$$s_{\tilde{H}} \leq \max_{i \in \{1,\ldots,N\}} \{s_i\},$$

avec s_i le rapport de contraction de T_i pour $i \in \{1, \ldots, N\}$.

Démonstration : on a

$$\tilde{H}_{ij} = \begin{cases} \bigcup_{k \in I_{ij}} T_k \text{ si } (i,j) \in R \\ \emptyset \text{ sinon} \end{cases}.$$

Or, on a vu que l'opérateur de Hutchinson associé à N fonctions contractantes est contractant de rapport $\max_{i \in \{1,\ldots,N\}} \{s_i\}$, donc $\tilde{H}_{ij}$ est contractant de rapport

$$s_{ij} = \max_{k \in I_{ij}} \{s_k\}.$$

Et donc

$$s_{ij} \leq \max_{k \in \{1,\ldots,N\}} \{s_k\}.$$

On pose $s = \max_{k \in \{1,\ldots,N\}} \{s_k\}$.

Soient $\tilde{K}, \tilde{K}' \in M_R(\mathcal{H}(\mathcal{X}))$

$$\begin{array}{rcl} d_\infty(\tilde{H} \circ \tilde{K}, \tilde{H} \circ \tilde{K}') & = & \max_{(i,j) \in R} d_H \left((\tilde{H} \circ \tilde{K})_{ij}, (\tilde{H} \circ \tilde{K}')_{ij} \right) \\ & = & \max_{(i,j) \in R} d_H \left(\bigcup_{k \in \{1,\ldots,N\}} \tilde{H}_{ik} \circ \tilde{K}_{kj}, \bigcup_{k \in \{1,\ldots,N\}} \tilde{H}_{ik} \circ \tilde{K}'_{kj} \right) \\ & \leq & \max_{(i,j) \in R} \max_{k/(i,k),(k,j) \in R} d_H(\tilde{H}_{ik} \circ \tilde{K}_{kj}, \tilde{H}_{ik} \circ \tilde{K}'_{kj}) \\ & \leq & \max_{(i,j),(i,k),(k,j) \in R} s_{ik} d_H(\tilde{K}_{kj}, \tilde{K}'_{kj}) \\ & \leq & s \max_{(i,j),(i,k),(k,j) \in R} d_H(\tilde{K}_{kj}, \tilde{K}'_{kj}) \\ & \leq & s \max_{(k,j) \in R} d_H(\tilde{K}_{kj}, \tilde{K}'_{kj}) \\ & \leq & s d_\infty(\tilde{K}, \tilde{K}') \end{array}$$

Et donc $\tilde{H}$ est bien contractant de rapport

$$s_{\tilde{H}} \leq \max_{k \in \{1,\ldots,N\}} \{s_k\} < 1.$$

□

3.1.3 Lien entre matrice et automate

On définit la matrice associée à un automate comme suit :

Définition 3.1.3 ([GM86]) *Soit $\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$ un automate fini et $\mathcal{T} = \{T_1, \ldots, T_N\}$ un IFS. La matrice d'adjacence associée à $\mathcal{M}$ est*

$$\Delta(\mathcal{M}) = (\Delta_{ij}),$$

$$\Delta_{ij} = \{a \in \Sigma / \delta(q_i, a) \ni q_j\}.$$

En posant $h(\Delta(\mathcal{M}))_{ij} = h(\Delta_{ij})$, on pourra voir les matrices de Hutchinson comme des matrices d'adjacence d'un graphe d'automate :

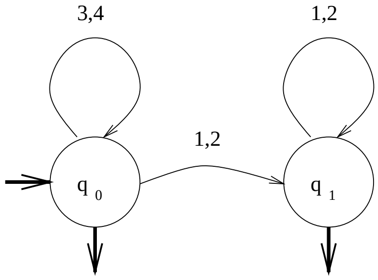
$$\tilde{H} = h(\Delta(\mathcal{M})).$$

La donnée de la matrice $\Delta(\mathcal{M})$ définit parfaitement les arêtes et les sommets du graphe $G(\mathcal{M})$, mais ne rend pas compte des états initiaux et finaux. Pour cela, on va ajouter à la matrice la donnée de deux vecteurs I et F définis par :

$$I_j = \begin{cases} \{\epsilon\} \text{ si } q_j \in Q_I \\ \emptyset \text{ si } q_j \notin Q_I \end{cases},$$

$$F_j = \begin{cases} \{\epsilon\} \text{ si } q_j \in Q_F \\ \emptyset \text{ si } q_j \notin Q_F \end{cases}.$$

Exemple : Si l'on considère l'automate $\mathcal{M}_{arbre}$:



on a

$$\Delta(\mathcal{M}_{arbre}) = \begin{pmatrix} \{3,4\} & \{1,2\} \\ \emptyset & \{1,2\} \end{pmatrix},$$

$$h(\Delta(\mathcal{M}_{arbre})) = \begin{pmatrix} h(\{3,4\}) & h(\{1,2\}) \\ \emptyset & h(\{1,2\}) \end{pmatrix} = \begin{pmatrix} \{T_3, T_4\} & \{T_1, T_2\} \\ \emptyset & \{T_1, T_2\} \end{pmatrix} = \tilde{H}_{arbre}.$$

$$\tilde{I}_{arbre} = \begin{pmatrix} \mathrm{Id} \\ \emptyset \end{pmatrix}, \quad \tilde{F}_{arbre} = \begin{pmatrix} \mathrm{Id} \\ \mathrm{Id} \end{pmatrix}.$$

3.1.4 Définition de la matrice d'attracteurs

Nous allons définir une matrice d'attracteurs comme limite de l'itération d'une matrice de Hutchinson. Pour que cette limite existe, il faut que la matrice de Hutchinson soit apériodique. Intuitivement, on dit qu'un graphe est apériodique si, à partir d'un certain rang m, lorsqu'il existe un chemin d'un état i à un état j de longueur m, alors il existe des chemins de i à j de toutes les longueurs supérieures à m. Cela va se traduire sur la matrice d'adjacence du graphe par la stabilité des composantes vides (qui correspondent à "il n'existe pas de chemin") à partir d'un certain nombre d'itérations. Plus précisément, on a

Proposition 3.1.6 *Soit $\tilde{H} \in M_R(\mathcal{P}_+(\mathcal{S}))$ une matrice de* Hutchinson *telle que toutes les composantes connexes du graphe d'automate correspondant soient apériodiques. On dira que $\tilde{H}$ est apériodique, c'est-à-dire qu'il existe $m \in \mathbb{N}$ tel que*

$$\forall k > m, R^m = R^k.$$

En posant $S = R^m$, $M_S(\mathcal{P}_+(\mathcal{S}))$ est stable par $\circ$ et

$$\forall k > m, \tilde{H}^k \in M_S(\mathcal{H}(\mathcal{X})).$$

Par ailleurs, si $\tilde{K} \in M_I(\mathcal{H}(\mathcal{X}))$ avec $I = \{(i,i)/1 \leq i \leq n\}$, on a

$$\forall k > m, \tilde{H}^k \circ \tilde{K} \in M_S(\mathcal{H}(\mathcal{X})).$$

On considère dans la suite des matrices de Hutchinson apériodiques.

Proposition 3.1.7 *La suite $(\tilde{H}^k \circ \tilde{K})_{k>m}$ est une suite de* Cauchy *dans l'espace métrique complet $(M_S(\mathcal{P}_+(\mathcal{S})), d_\infty)$.*

Démonstration : D'après la proposition 3.1.6,

$$\tilde{H}^m \circ \tilde{K} \in M_S(\mathcal{H}(\mathcal{X}))$$

et

$$\forall l > m, \tilde{H}^l \in M_S(\mathcal{H}(\mathcal{X})).$$

Donc, en posant $\tilde{K}_0 = \tilde{H}^m \circ \tilde{K}$ et $\tilde{K}_i = \tilde{H}^i \circ \tilde{K}_0$, on a

$$d_\infty(\tilde{K}_l, \tilde{K}_{l+p}) = d_\infty(\tilde{H}^l \circ \tilde{K}_0, \tilde{H}^l \circ \tilde{K}_p).$$

Comme $\tilde{H}$ est contractante de rapport $s_{\tilde{H}}$, $\tilde{H}^l$ est contractante de rapport $s^l_{\tilde{H}}$. Donc

$$d_\infty(\tilde{K}_l, \tilde{K}_{l+p}) \leq s^l_{\tilde{H}} d_\infty(\tilde{K}_0, \tilde{K}_p).$$

Ainsi, la suite $(\tilde{K}_l)$ est une suite de CAUCHY. □

Définition 3.1.4 *On définit la matrice d'attracteurs associée à une matrice de* HUTCHINSON $\tilde{H}$ *de la façon suivante :*

$$\tilde{\mathcal{A}}(\tilde{H}) = \lim_{l \to \infty} \tilde{H}^l \circ (\tilde{H}^m \circ \tilde{K}).$$

Par souci de simplicité, on notera $\tilde{\mathcal{A}}(\tilde{H}) = \lim_{n \to \infty} \tilde{H}^n \circ \tilde{K}$.

Définition 3.1.5 *On définit l'attracteur "scalaire" à partir de la matrice attracteur en considérant l'union de toutes les composantes de la matrice :*

$$\mathcal{A}(\tilde{H}) = \bigcup_{(i,j) \in R} \tilde{\mathcal{A}}(\tilde{H}).$$

Cet attracteur nous permettra de comparer le résultat des diverses opérations que nous effectuons sur le modèle langage et sur modèle matriciel.

Exemple : Dans le cas de l'IFS $\mathcal{T}_{arbre} = \{T_1, T_2, T_3, T_4\}$, avec la matrice de HUTCHINSON

$$\tilde{H}_{arbre} = \begin{pmatrix} \{T_3, T_4\} & \{T_1, T_2\} \\ \emptyset & \{T_1, T_2\} \end{pmatrix}.$$

On a

$$\tilde{H}^k_{arbre} = \begin{pmatrix} \{T_3, T_4\}^k & \bigcup_{i+j=k-1} \{T_3, T_4\}^i \{T_1, T_2\}^j \\ \emptyset & \{T_1, T_2\}^k \end{pmatrix},$$

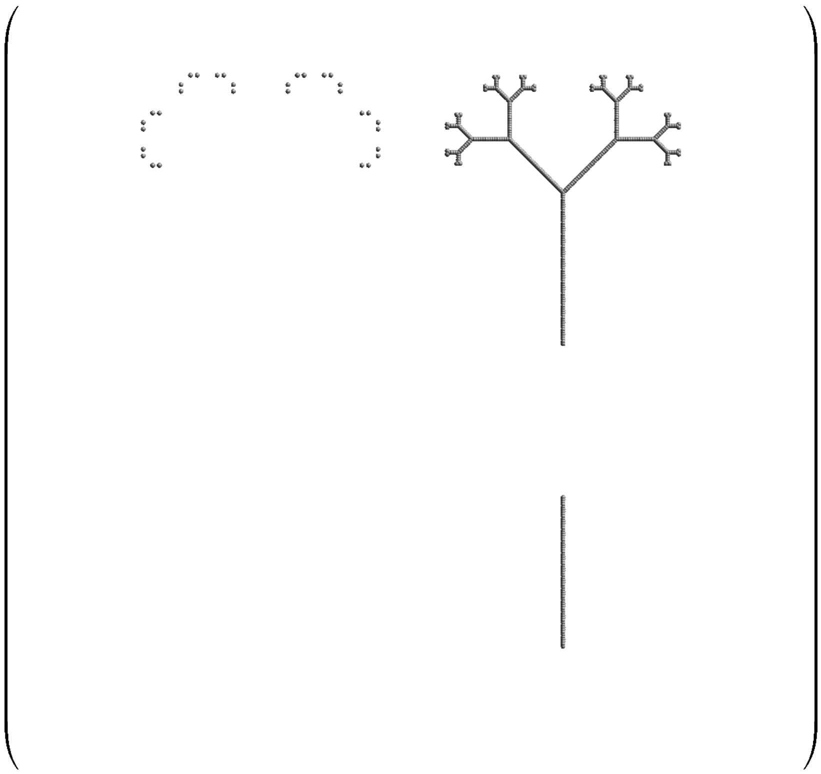


FIG. 3.1 - *Matrice d'attracteurs $\tilde{\mathcal{A}}(\tilde{H}_{arbre})$.*

et donc

$$\tilde{\mathcal{A}}(\tilde{H}_{arbre}) = \lim_{n\to\infty} \tilde{H}^n \circ \tilde{K} = \begin{pmatrix} \mathcal{A}(T_3, T_4) & \mathcal{A}(T_3, T_4) \cup \{T_3, T_4\}^* \circ \mathcal{A}(T_1, T_2) \\ \emptyset & \mathcal{A}(T_1, T_2) \end{pmatrix}.$$

Les éléments de $\tilde{\mathcal{A}}(\tilde{H}_{arbre})$ sont donnés Figure 3.1.

$$\mathcal{A}(\tilde{H}_{arbre}) = \bigcup_{(i,j)\in R} \tilde{\mathcal{A}}(\tilde{H}) = \mathcal{A}(T_3, T_4) \cup \{T_3, T_4\}^* \circ \mathcal{A}(T_1, T_2) = \tilde{\mathcal{A}}_{arbre}.$$

3.1.4.1 Visualisation

Nous pouvons utiliser l'algorithme déterministe pour visualiser la matrice d'attracteurs. On visualisera alors $\tilde{H}^n \circ \tilde{K}$, avec $\tilde{K} \in M_I(\mathcal{H}(\mathcal{X}))$ (la matrice identité).

Exemple : La Figure 3.2 montre les premières étapes de l'algorithme déterministe.

3.2 Lien avec le modèle LRIFS

Dans le cas d'un langage rationnel, on va pouvoir utiliser le modèle matriciel pour obtenir l'attracteur associé au langage tel que nous l'avons défini au chapitre précédent.

3.2.1 Attracteur associé à un automate

Définition 3.2.1 *L'attracteur associé à un automate $\mathcal{M}$ est égal à l'union des éléments de la matrice d'attracteurs, qui correspondent à un état initial et à un état final :*

$$\mathcal{A}(\mathcal{M}) = \bigcup_{q_i \in Q_I, q_j \in Q_F} \tilde{\mathcal{A}}(h(\Delta(\mathcal{M})))_{ij}.$$

Exemple : L'attracteur associé au langage L_{arbre} reconnu par l'automate $\mathcal{M}_{arbre}$ sera égal à l'union des composantes de la matrice d'attracteurs $\tilde{\mathcal{A}}(h(\Delta(\mathcal{M}_{arbre}))) = \tilde{\mathcal{A}}(\tilde{H}_{arbre})$ qui correspondent à un état final et un état initial, c'est-à-dire à $\tilde{\mathcal{A}}_{11} \cup \tilde{\mathcal{A}}_{12}$.

3.2.2 Équivalence avec le modèle LRIFS

Proposition 3.2.1 *Si L est un langage rationnel reconnu par un automate $\mathcal{M}$, alors*

$$\mathcal{A}(\mathcal{M}) = \mathcal{A}(L).$$

C'est-à-dire que l'attracteur obtenu par le modèle LRIFS est égal à celui obtenu par le modèle "matrice".

Démonstration : On sait que Δ^n va contenir tous les chemins de longueur n du graphe de l'automate $\mathcal{M}$. On peut alors remarquer que $I^t\Delta^n F$ est égal à l'ensemble des mots associés aux chemins de longueur n reliant un état initial à un état final, c'est-à-dire à l'ensemble des mots de longueur n reconnus par $\mathcal{M}$.

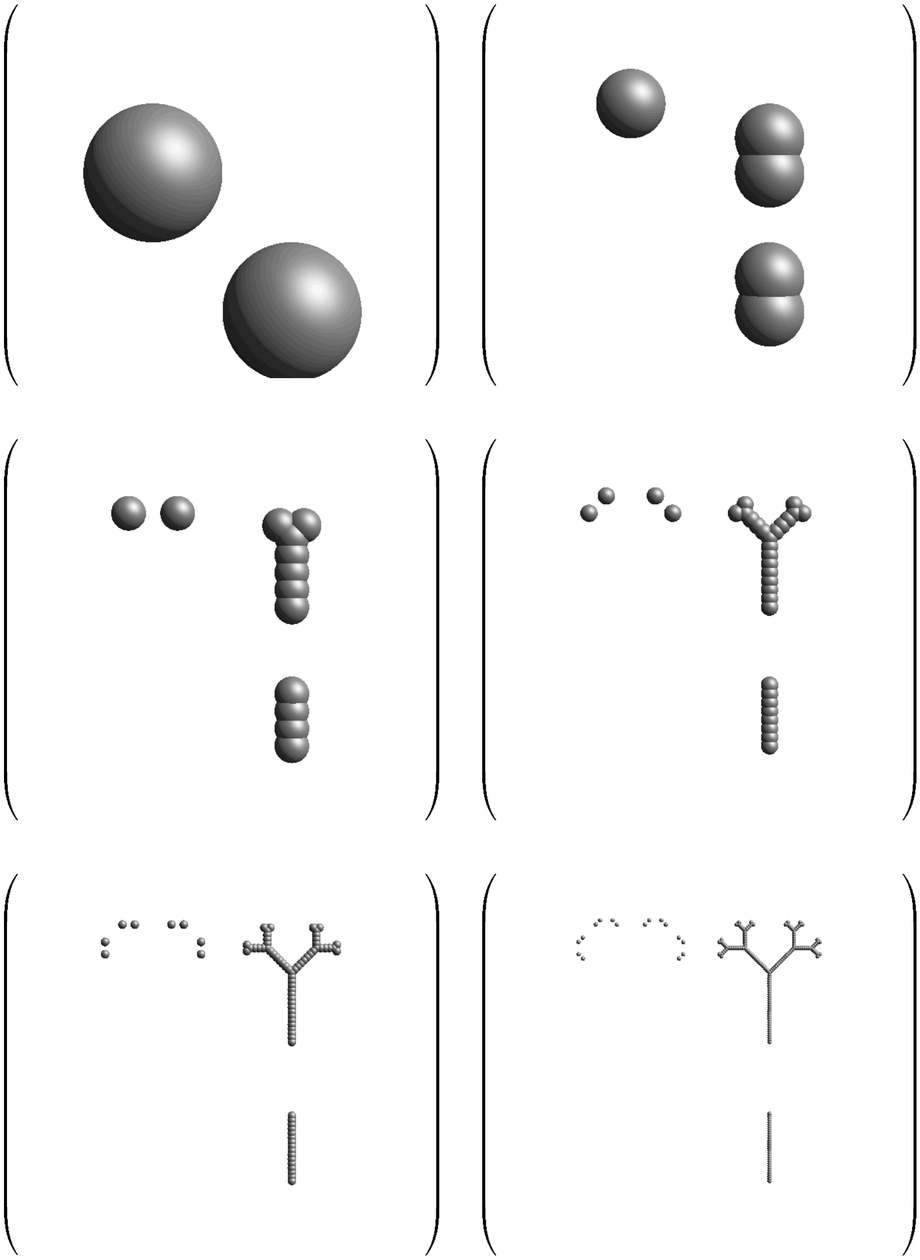


FIG. 3.2 - *Algorithme déterministe pour la matrice d'attracteurs.*

On sait que, pour reconnaître les préfixes d'un langage à l'aide d'un automate fini, il suffit de rendre finaux les états de l'automate qui sont sur un chemin allant d'un état initial à un état final. On obtient alors un nouveau vecteur final, que l'on note F'.

On a alors

$$\begin{aligned}
L_n &= I^t \Delta^n F', \\
\lim_{n\to\infty} (h(L_n) \circ K) &= \lim_{n\to\infty} (h(I^t \Delta^n F') \circ K) \\
&= \lim_{n\to\infty} (h(I)^t \circ h(\Delta^n) \circ h(F') \circ K) \\
&= h(I)^t \circ \lim_{n\to\infty} (h(\Delta^n) \circ h(F') \circ K).
\end{aligned}$$

Pour K compact, on pose $K \circ \mathrm{Id} = K$ et $K \circ \emptyset = \emptyset$. Cette notation est étendue aux matrices et vecteurs de la façon usuelle. La composition d'une matrice de compacts par un vecteur dont les éléments sont égaux à Id ou $\emptyset$, correspond à l'extraction de certaines composantes de la matrice.

Comme $h(F')$ est un vecteur dont les éléments sont égaux à Id ou $\emptyset$ on peut écrire

$$\begin{aligned}
h(I)^t \circ \lim_{n\to\infty} (h(\Delta^n) \circ h(F') \circ K) &= h(I)^t \circ \lim_{n\to\infty} (h(\Delta^n) \circ K) \circ h(F') \\
&= h(I)^t \circ \tilde{\mathcal{A}}(h(\Delta)) \circ h(F') \\
&= \bigcup_{q_i \in Q_I, q_j \in Q_F} \tilde{\mathcal{A}}(h(\Delta))_{ij} \\
&= \mathcal{A}(\mathcal{M}).
\end{aligned}$$

C'est-à-dire :

$$\mathcal{A}(L) = \mathcal{A}(\mathcal{M}),$$

l'attracteur obtenu avec le modèle LRIFS est égal à celui obtenu avec le modèle matriciel.

$\square$

L'attracteur associé à un automate tient compte des états initiaux et finaux. On définira donc cet attracteur avec le formalisme matriciel de la façon suivante :

Définition 3.2.2

$$\mathcal{A}(\mathcal{M}) = \mathcal{A}(\tilde{H}, \tilde{I}, \tilde{F}),$$

avec $\tilde{H} = h(\Delta(\mathcal{M}))$, $\tilde{I} = h(I)$ *et* $\tilde{F} = h(F)$.

Exemple :

$$\mathcal{A}(\mathcal{M}_{arbre}) = \mathcal{A}(\tilde{H}_{arbre}, \tilde{I}_{arbre}, \tilde{F}_{arbre}),$$

avec

$$\tilde{H}_{arbre} = \begin{pmatrix} \{T_3, T_4\} & \{T_1, T_2\} \\ \emptyset & \{T_1, T_2\} \end{pmatrix}.$$

$$\tilde{I}_{arbre} = \begin{pmatrix} \mathrm{Id} \\ \emptyset \end{pmatrix}, \quad \tilde{F}_{arbre} = \begin{pmatrix} \mathrm{Id} \\ \mathrm{Id} \end{pmatrix}.$$

3.3 Résumé

Nous avons défini un deuxième modèle permettant de généraliser le modèle IFS. Ce modèle nous permet de donner un algorithme de visualisation, et de combiner deux attracteurs en définissant des opérations sur les matrices qui les génèrent. Si l'on considère une opération $\diamondsuit$ sur les matrices de Hutchinson, nous construirons l'attracteur

$$\tilde{\mathcal{A}}(\tilde{H} \diamondsuit \tilde{H}').$$

En résumé, nous avons obtenu plusieurs définitions d'attracteurs:

$$\begin{aligned} \mathcal{A}(L) &= \lim_{n \to \infty} h(L_n) \circ K; \\ \tilde{\mathcal{A}}(\tilde{H}) &= \lim_{n \to \infty} \tilde{H}^n \circ K; \\ \mathcal{A}(\tilde{H}) &= \bigcup_{(i,j) \in R} \tilde{\mathcal{A}}(\tilde{H}); \\ \mathcal{A}(\mathcal{M}) &= \mathcal{A}(\tilde{H}, I, F) \\ &= I^t \circ \tilde{\mathcal{A}}(\tilde{H}) \circ F \\ &= \bigcup_{q_i \in Q_I, q_j \in Q_F} \tilde{\mathcal{A}}(\tilde{H})_{ij}. \end{aligned}$$

Ces différentes définitions permettront de comparer (au sens de l'inclusion) un attracteur obtenu par composition de langages à un autre obtenu par composition de matrices.

Chapitre 4

Équivalence avec les autres modèles

Nous avons décrit, au Chapitre 1, les différents modèles d'extension d'IFS que l'on trouve dans la littérature. Nous nous intéressons maintenant à leur lien avec notre modèle matriciel. Nous allons prouver leur équivalence en les ramenant à notre formalisme.

Nous avons classé ces modèles par le type de représentation utilisée. Nous allons voir que toutes ces représentations peuvent se ramener à un choix des composantes de la matrice d'attracteurs.

4.1 Vecteur

Nous reprenons ici la démarche de Peitgen, Jürgens et Saupe [PJS92] avec un formalisme un peu différent. Soit $\tilde{H}$ une matrice de Hutchinson. Au lieu de construire la matrice d'attracteurs correspondant à cet opérateur, on va construire son point fixe.

Nous détaillons cette partie car les modèles fondés sur les graphes et les systèmes d'équations s'y ramènent de manière directe.

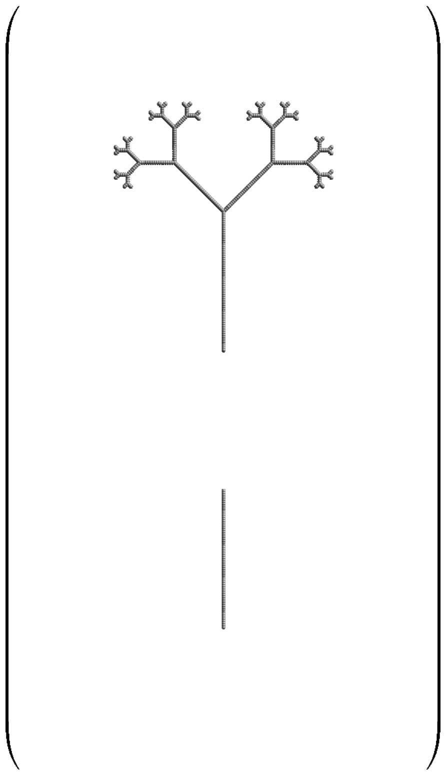


FIG. 4.1 - *Vecteur attracteur associé à $\tilde{H}_{arbre}$.*

4.1.1 Attracteur

On peut définir l'attracteur associé à une matrice de HUTCHINSON en appliquant le Théorème du Point Fixe dans $\mathcal{H}(\mathcal{X})^n$.

Proposition 4.1.1 *Soit $\tilde{H}$ une matrice de* HUTCHINSON *associé à l'IFS $\mathcal{T}$, il existe un unique point fixe $X \in \mathcal{H}(\mathcal{X})^n$ tel que*

$$X = \tilde{H} \circ X.$$

X sera appelé le vecteur attracteur associé à $\tilde{H}$.

Exemple : Le vecteur attracteur associé à $\tilde{H}_{arbre}$ est donné Figure 4.1.

4.1.2 Équivalence avec le modèle matriciel

Le vecteur attracteur associé à une matrice de HUTCHINSON se déduit très simplement de la matrice d'attracteurs, en regroupant les composantes de la matrice par ligne. Plus précisément, on a :

Proposition 4.1.2 *Soit $\tilde{H}$ une matrice de* HUTCHINSON *et X son point fixe. Alors :*

$$X_i = \bigcup_{j\in\{1,\ldots n\}} \tilde{\mathcal{A}}(\tilde{H})_{ij}.$$

Démonstration : X est le point fixe de $\tilde{H}$, donc on peut écrire :

$$X = \lim_{n\to\infty} (\tilde{H}^n \circ Y),$$

avec $Y \in \mathcal{H}(\mathcal{X})^n$. On a donc

$$\begin{aligned} X_i &= \bigcup_{j\in\{1,\ldots n\}} \tilde{H}_{ij} \circ Y_j \\ &= \bigcup_{j\in\{1,\ldots n\}} \tilde{\mathcal{A}}(\tilde{H})_{ij}. \end{aligned}$$

□

Exemple : Dans le cas de l'arbre, on a

$$\tilde{\mathcal{A}}(\tilde{H}_{arbre}) = \begin{pmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathcal{A}(T_3,T_4) & \mathcal{A}(T_3,T_4) \cup \{T_3,T_4\}^* \circ \mathcal{A}(T_1,T_2) \\ \emptyset & \mathcal{A}(T_1,T_2) \end{pmatrix},$$

et

$$X = \begin{pmatrix} \mathcal{A}_{11} \cup \mathcal{A}_{12} \\ \mathcal{A}_{21} \cup \mathcal{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathcal{A}(T_3,T_4) \cup \{T_3,T_4\}^* \circ \mathcal{A}(T_1,T_2) \\ \mathcal{A}(T_1,T_2) \end{pmatrix}.$$

4.2 Graphes

Nous allons maintenant étudier les modèles fondés sur des graphes. Ils se ramènent de manière simple au modèle précédent, en prenant comme matrice de HUTCHINSON la matrice d'adjacence du graphe.

4.2.1 Modèle de MAULDIN et WILLIAMS

MAULDIN et WILLIAMS [MW88] ont défini un modèle fondé sur les graphes. Soit un graphe $G = (V, E)$ à n sommets. Chaque arc $e \in E$ du graphe est étiqueté par une transformation $T_e \in \mathcal{T}$ de l'IFS.

Soit X le vecteur qui satisfait :

$$X_i = \bigcup_{e=(i,j)\in E} T_e \circ X_j.$$

Par définition, l'attracteur associé au graphe est l'union des composantes de X :

$$\mathcal{A}(G) = \bigcup_{i=1}^{n} X_i.$$

Proposition 4.2.1

$$\mathcal{A}(G) = \mathcal{A}(\tilde{H}),$$

avec $\tilde{H}$ une matrice de Hutchinson *construite comme la matrice d'adjacence de G.*

Démonstration : En notant le graphe différemment : $G = (V, E)$, avec $E = \{(i, e, j)\}$, on a

$$X_i = \bigcup_{(i,e,j)\in E} T_e \circ Xj. \tag{4.1}$$

On définit $\tilde{H}$ comme la matrice d'adjacence de G :

$$\tilde{H}_{ij} = \{T_e/(i, e, j) \in E\}.$$

Le vecteur X satisfaisant (4.1) est le vecteur point fixe de $\tilde{H}$, donc on a

$$X_i = \bigcup_{j\in\{1,\dots n\}} \tilde{\mathcal{A}}(\tilde{H})_{ij},$$

$$\mathcal{A}(G) = \bigcup_{i\in\{1,\dots n\}} X_i = \bigcup_{i,j\in\{1,\dots n\}} \tilde{\mathcal{A}}(\tilde{H})_{ij} = \mathcal{A}(\tilde{H}).$$

□

Exemple : Le graphe donné figure 4.2 est équivalent à $\tilde{H}_{arbre}$. L'attracteur associé est égal à l'union des composantes du vecteur d'attracteur.

$$\mathcal{A}(G_{arbre}) = \bigcup_{i,j\in\{1,\dots n\}} \tilde{\mathcal{A}}(\tilde{H}_{arbre})_{ij} = \mathcal{A}(\tilde{H}_{arbre}) = \tilde{\mathcal{A}}_{arbre}.$$

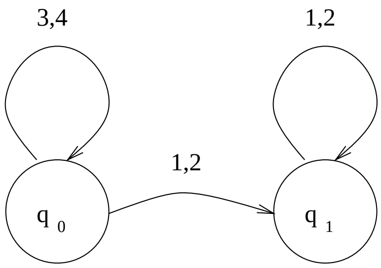


FIG. 4.2 - *Graphe de* MAULDIN *et* WILLIAMS.

4.2.2 Modèle de BARNSLEY

BARNSLEY [BHH89] a introduit le modèle RIFS (Recurrent IFS). Ce modèle est fondé sur un graphe $G = (V, E)$ à N sommets. Contrairement au modèle précédent, ce sont cette fois les sommets qui correspondent aux transformations de l'IFS.

L'attracteur associé à G sera donné par l'union des solutions du système :

$$X_j = \bigcup_{(i,j)\in E} T_j \circ X_i.$$

La matrice de HUTCHINSON correspondante sera donnée par la matrice associée au système d'équations. De même que pour le modèle précédent, en terme matriciel, l'attracteur associé au graphe G sera égal à l'union de toutes les composantes de $\tilde{\mathcal{A}}(\tilde{H})$.

Exemple : Le graphe donné Figure 4.3 admet $\tilde{\mathcal{A}}_{arbre}$ comme attracteur.

4.3 Système d'équations

La troisième catégorie de modèles se ramenant simplement à la matrice de HUTCHINSON est celle des modèles fondés sur des systèmes d'équations. En effet, il suffira de construire la matrice qui est associée à un système d'équations, de manière usuelle.

4.3.1 Modèle de BANDT

BANDT [Ban89] représente la matrice de HUTCHINSON par un système d'équations, qu'il appelle "*a sofic system*". Ce système à n inconnues est

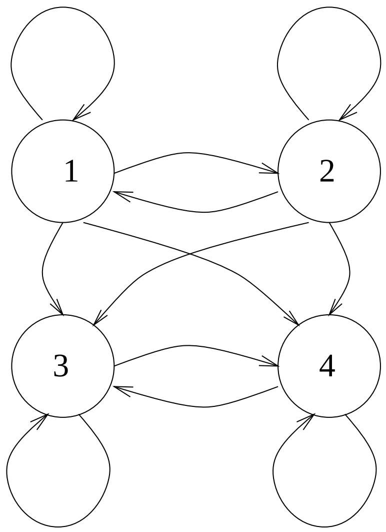


FIG. 4.3 - *Graphe de* BARNSLEY.

donné par des équations du type :

$$X_i = \bigcup T_k \circ X_j \quad \text{avec } i, j \in \{1, \ldots n\} \; k \in \{1, \ldots, N\}.$$

L'attracteur associé à un tel système d'équations est par définition égal à l'union des compacts solutions du système :

$$\mathcal{A} = \bigcup_{i=1}^{n} X_i.$$

Proposition 4.3.1

$$\mathcal{A} = \mathcal{A}(\tilde{H}),$$

avec $\tilde{H}$ la matrice de HUTCHINSON *associée au système.*

Démonstration : On peut reformuler la définition de BANDT de la manière suivante :

$$X_i = \bigcup_{(i,k,j)} T_k \circ X_j.$$

On retrouve alors la même démarche qu'en 4.2.1. □

Exemple : Le "*sofic system*"

$$\begin{cases} X_1 = T_3 \circ X_1 \cup T_4 \circ X_1 \cup T_1 \circ X_2 \cup T_2 \circ X_2 \\ X_2 = T_1 \circ X_2 \cup T_2 \circ X_2 \end{cases}$$

est équivalent à $\tilde{H}_{arbre}$. L'attracteur associé est égal à $\mathcal{A}(\tilde{H}_{arbre})$ et donc à $\tilde{\mathcal{A}}_{arbre}$.

4.3.2 Modèle de Culik et Dube

Culik et Dube [CD93b] ont défini un modèle fondé sur un système d'équations. Il diffère du précédent car ce modèle distingue certaines variables comme variables d'affichage. Elles permettront simplement de sélectionner certaines composantes du point fixe.

Définition 4.3.1 *Un MRFS (ou* Mutually Recursive Function System*) est un système de n équations de la forme :*

$$A = T_1 \circ B_1 \cup T_2 \circ B_2 \cup \ldots \cup T_n \circ B_n,$$

avec A et B_i des compacts, T_i des transformations affines de $\mathcal{T}$.

Théorème 4.3.1 (Attracteur) *Un MRFS possède un attracteur unique donné par :*

$$\mathcal{A} = \bigcup_{A_i \in D} A'_i,$$

avec A'_i l'instanciation de la variable A_i qui reste inchangée par l'application du MRFS, et $D \subseteq \{A_i / 1 \leq i \leq n\}$ un ensemble de variables d'affichage.

Proposition 4.3.2

$$\mathcal{A} = \bigcup_{i/A_i \in D, j \in \{1,\ldots n\}} \mathcal{A}(\tilde{H})_{ij},$$

avec $\tilde{H}$ la matrice de Hutchinson *associée au système d'équation.*

Démonstration : La démonstration est la même que précédemment. □

Exemple : On obtient l'attracteur $\tilde{\mathcal{A}}_{arbre}$ comme solution du système

$$\begin{cases} X_1 = T_3 \circ X_1 \cup T_4 \circ X_1 \cup T_1 \circ X_2 \cup T_2 \circ X_2 \\ X_2 = T_1 \circ X_2 \cup T_2 \circ X_2 \end{cases}$$

et l'ensemble de variable d'affichage $D = \{X_1\}$.

4.4 Automates

Les modèles fondés sur les automates n'utilisent pas toujours l'algorithme déterministe. En particulier, CULIK [CD93b] utilise l'algorithme de chaos dynamique. Cependant, il a prouvé que son modèle était équivalent au modèle MRFS, nous le citerons donc seulement pour mémoire. Dans les autres cas, la preuve de l'équivalence se ramène à la définition que nous avons donnée de l'attracteur associé à un automate (voir Section 3.2.1).

4.4.1 Modèle de CULIK et DUBE

Définition 4.4.1 *Un PAA (ou* Probabilistic Affine Automata*) est un quintuplet* $\mathcal{M} = (Q, \Sigma, \delta, P, F)$*, avec :*

- $Q = \{q_1, q_2, \ldots, q_n\}$ *un ensemble fini appelé l'ensemble des états;*
- $\Sigma = \{T_1, T_2, \ldots, T_N\}$ *un alphabet de transformations affines;*
- δ *la fonction de transition de l'automate telle que le graphe de l'automate soit fortement connexe et de cycles contractants;*
- P *une matrice stochastique* $n \times m \times n$ *telle que*

$$\forall i \sum_{k=1}^{n} \sum_{j=1}^{n} P(i, j, k) = 1;$$

- $F \subset Q$ *l'ensemble des états terminaux.*

La valeur $P(i, j, k)$ *est la probabilité associée à la transition* (q_i, T_j, q_k)*. La somme des probabilités des transitions partant d'un même nœud est égale à 1.*

L'attracteur est construit selon l'algorithme non déterministe : on choisit aléatoirement un point courant p, et un état courant q. À chaque itération, on choisit aléatoirement une transition (q, T, q') partant de q. Le point courant devient alors $T \circ p$, et l'état courant devient q'. Le nouveau point courant est affiché si q' est un état final.

La suite des points générés aux états finaux est alors une approximation de l'attracteur.

4.4.2 Modèle de Morcrette

Morcrette [Mor96] a introduit le modèle IFSA (IFS à automate). Elle associe un automate fini à un IFS et définit l'attracteur associé de la manière suivante :

Définition 4.4.2 *Soit $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$ un IFS et $\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$ un automate. Une image d'ordre n est donnée par :*

$$\mathcal{A}_n(K) = h(L(\mathcal{M}) \cap \Sigma^n) \circ K,$$

et l'attracteur est alors

$$\mathcal{A}(K) = \lim_{n \to \infty} \mathcal{A}_n(K).$$

L'attracteur est donc défini par la limite des mots de longueur n, reconnus par $\mathcal{M}$, appliqués à un compact. Comme nous l'avons montré, cette définition est équivalente à la définition que nous avons donnée Section 3.2.1.

Remarque : Morcrette a prouvé l'équivalence du modèle IFSA avec le modèle des "*recurrent sets*" défini par Dekking [Dek87].

4.5 Expressions rationnelles

4.5.1 Modèle de Culik

Culik [CD93b] a défini un modèle fondé sur les expressions rationnelles de la manière suivante :

Définition 4.5.1 *Un ARS (ou* Affine Regular Set*) est un ensemble généré par une expression rationnelle sur Σ.*

Définition 4.5.2 *Soit $R \subseteq \Sigma^*$ un ARS. L'attracteur associé à R est*

$$\mathcal{A} = \phi(\mathrm{ls}(R)).$$

Exemple : Si $R = \{3, 4\}^*\{1, 2\}^*$ l'attracteur associé à R est $\mathcal{A}_{arbre}$.

Nous avons vu au Chapitre 2 que cette définition est équivalente à celle que nous avons proposée.

4.6 Résumé des équivalences

Type de modèle	Auteur	Équivalence avec
Langage quelconque	PRUSINKIEWICZ	Langage
Vecteur	PEITGEN, JÜRGENS et SAUPE	Matrice
Graphe	MAULDIN et WILLIAMS	Matrice
	BARNSLEY	Matrice
Système d'équations	BANDT	Matrice
	CULIK et DUBE	Matrice
Automate	CULIK et DUBE	MRFS
	MORCRETTE	Matrice
Expression rationnelle	CULIK et DUBE	Langage

4.7 Conclusion

Nous avons montré l'équivalence entre notre modèle matriciel et les modèles existants. Le modèle matriciel présente en outre l'avantage de permettre un calcul en parallèle de toutes les composantes de la matrice d'attracteur et une sélection *a posteriori* des composantes intéressantes.

Le choix des composantes de la matrice d'attracteurs va nous permettre d'obtenir plusieurs types d'attracteurs :

- Attracteur matriciel :
$$\tilde{\mathcal{A}}(\tilde{H});$$
- Attracteur vectoriel :
$$X_i = \bigcup_{j\in\{1,\ldots,n\}} \tilde{\mathcal{A}}(\tilde{H})_i;$$
- Attracteur scalaire :
$$\mathcal{A}(\tilde{H}) = \bigcup_{i,j\in\{1,\ldots,n\}} \tilde{\mathcal{A}}(\tilde{H})_{ij};$$
- Attracteur langage :
$$\mathcal{A}(\mathcal{M}) = \bigcup_{q_i\in Q_I, q_j\in Q_F} \tilde{\mathcal{A}}(\tilde{H})_{ij}.$$

Contrairement aux modèles antérieurs, nous pouvons, à partir d'un seul formalisme, obtenir tous les types d'attracteurs présents dans la littérature liée aux IFS.

Approche constructive

Nous allons maintenant définir un modèle de géométrie fractale constructive, inspirée de la géométrie constructive du solide (*Constructive Solid Geometry*), telle que définie par REQUICHA [Req80] ou ROTH [Rot82]. La géométrie constructive est, en effet, déjà largement utilisée pour la construction de scènes classiques. Il serait donc intéressant de pouvoir créer des objets fractals de manière constructive.

Dans la géométrie constructive volumique classique, un solide est représenté par la composition, à l'aide d'un ensemble d'opérations régularisées, d'autres solides. Au plus bas niveau se trouvent les solides les plus simples, appelés primitives.

De la même manière, nous pouvons définir un objet fractal par la composition, à l'aide d'un ensemble d'opérations particulières, d'autres objets fractals. Au plus bas niveau se trouvent les primitives. Nous définissons, ainsi, des opérations s'appuyant sur les deux modèles que nous avons présentés. Nous aurons donc, d'une part, des opérations sur les langages et, d'autre part, des opérations sur les matrices. On construira alors les attracteurs correspondant aux résultats de ces opérations.

Parmi ces opérations, nous en choisissons ensuite certaines, qui nous permettent de définir un système de géométrie constructive.

Chapitre 5

Opérations

Nous avons proposé un modèle généralisant les IFS. À partir de ce modèle, nous allons définir des opérations permettant de composer deux attracteurs. Pour cela nous utilisons, d'une part, des opérations spécifiques à la Théorie des Langages : union, concaténation, mélange et intersection, et, d'autre part, des opérations directes sur les matrices de HUTCHINSON. Dans le cas d'un langage rationnel, nous donnons un procédé de construction et de visualisation à partir d'un automate ou d'une matrice de HUTCHINSON.

5.1 Cas d'un langage quelconque

Nous nous posons la question : étant donné une opération $\odot$ et deux langages L et L', quelle est la relation entre $\mathcal{A}(L \odot L')$, $\mathcal{A}(L)$ et $\mathcal{A}(L')$?

Nous travaillons avec des langages infinis, car l'attracteur associé à un langage fini est vide. Dans le cas d'un langage infini L, on a $\forall n \; L_n \neq \emptyset$.

5.1.1 Union

L'opération la plus naturelle en Théorie des Langages ainsi qu'en synthèse d'images est l'union ensembliste. L'union de deux langages L et L' est définie par :

$$L \cup L' = \{w \in \Sigma^* / w \in L \text{ ou } w \in L'\}.$$

On a alors la proposition

Proposition 5.1.1 *L'attracteur associé à l'union de deux langages est égal à l'union des attracteurs associés à chacun des langages.*

$$\mathcal{A}(L \cup L') = \mathcal{A}(L) \cup \mathcal{A}(L').$$

Démonstration : $L \cup L'$ est un langage infini, donc $(h((L \cup L')_n) \circ K)$ est une suite de CAUCHY, et est telle que :

$$(L \cup L')_n = L_n \cup L'_n.$$

De plus, la fonction

$$\begin{array}{rccc} \cup : & \mathcal{H}(\mathcal{X}) \times \mathcal{H}(\mathcal{X}) & \longrightarrow & \mathcal{H}(\mathcal{X}) \\ & (K, K') & \longmapsto & K \cup K' \end{array}$$

est une fonction continue. On a donc :

$$\begin{array}{rcl} \mathcal{A}(L \cup L') & = & \lim_{n\to\infty} h(L_n \cup L'_n) \circ K \\ & = & \lim_{n\to\infty} (h(L_n) \cup h(L'_n)) \circ K \\ & = & \lim_{n\to\infty} (h(L_n) \circ K \cup h(L'_n) \circ K) \\ & = & \lim_{n\to\infty} h(L_n) \circ K \cup \lim_{n\to\infty} h(L'_n) \circ K \\ & = & \mathcal{A}(L) \cup \mathcal{A}(L'). \end{array}$$

□

Corollaire 5.1.1

$$L \subseteq L' \Rightarrow \mathcal{A}(L) \subseteq \mathcal{A}(L').$$

5.1.2 Concaténation

L'autre opération naturelle en Théorie des Langages est la concaténation. La concaténation de deux mots $u = u_1 u_2 \dots u_n$ et $v = v_1 v_2 \dots v_m$ est notée $u.v$, et consiste à mettre les deux mots bout à bout :

$$u.v = u_1 u_2 \dots u_n v_1 v_2 \dots v_m.$$

La concaténation de deux langages est alors donnée par :

$$L.L' = \{w \in \Sigma^* / w = u.v \;\; u \in L \;\; v \in L'\}.$$

On a alors

Proposition 5.1.2 *L'attracteur associé à la concaténation de deux langages est égal à l'union de l'attracteur associé au premier langage, et des transformés selon le premier langage de l'attracteur associé au deuxième langage.*

$$\mathcal{A}(L.L') = \mathcal{A}(L) \cup h(L) \circ \mathcal{A}(L').$$

Démonstration : Soit $L'' = L.L'$. L'' est un langage infini, donc $(h(L''_n)\circ K)$ est une suite de Cauchy. On a

$$\begin{aligned}\mathcal{A}(L.L') &= \lim_{n\to\infty} h(L''_n) \circ K \\ &= \{\lim_{n\to\infty} h(u_1u_2\dots u_n) \circ p \mid u_1u_2\dots u_n \in L''_n,\ p \in K\}.\end{aligned}$$

Il y a deux types de suites $(h(u_1u_2\dots u_n) \circ p)$:

- $\forall n,\ u_1u_2\dots u_n \in L_n$;
- $\exists k,\ u_1u_2\dots u_k \in L_k$ et $u_{k+1}u_{k+2}\dots u_n \in L'_{n-k}$.

L'ensemble $\mathcal{A}(L.L') = \{\lim_{n\to\infty} h(u_1u_2\dots u_n)\circ p \mid u_1u_2\dots u_n \in L''_n,\ p \in K\}$ est donc partionné en :

$$\begin{aligned}&\mathcal{A}(L.L') = \{\lim_{n\to\infty} h(u_1u_2\dots u_n) \circ p \mid u_1u_2\dots u_n \in L_n,\ p\in K\} \\ &\cup \{\lim_{n\to\infty} h(u_1u_2\dots u_k)h(v_1v_2\dots v_n) \circ p \mid u_1u_2\dots u_k \in L,\ v_1v_2\dots v_n \in L'_n,\ p \in K\}.\end{aligned}$$

On a alors :

$$\{\lim_{n\to\infty} h(u_1u_2\dots u_n) \circ p \mid u_1u_2\dots u_n \in L_n,\ p \in K\} = \mathcal{A}(L)$$

et

$$\begin{aligned}&\{\lim_{n\to\infty} h(u_1u_2\dots u_k)h(v_1v_2\dots v_n) \circ p \mid u_1u_2\dots u_k \in L,\ v_1v_2\dots v_n \in L'_n,\ p \in K\} \\ &= \{h(u_1u_2\dots u_k) \circ \lim_{n\to\infty} h(v_1v_2\dots v_n) \circ p \mid u_1u_2\dots u_k \in L,\ v_1v_2\dots v_n \in L'_n,\ p\in K\} \\ &= \{h(u_1u_2\dots u_k) \mid u_1u_2\dots u_k \in L\} \circ \{\lim_{n\to\infty} h(v_1v_2\dots v_n) \circ p \mid v_1v_2\dots v_n \in L'_n,\ p \in K\} \\ &= h(L) \circ \mathcal{A}(L')\end{aligned}$$

Ainsi, $\mathcal{A}(L.L') = \mathcal{A}(L) \cup h(L) \circ \mathcal{A}(L')$.

□

5.1.3 Mélange

La troisième opération que nous allons étudier est le mélange de deux langages. Le mélange des deux mots, u et v est notée $u \sqcup v$ et consiste à intercaler les lettres de chacun des mots. On autorise tous les mélanges en ajoutant le mot vide dans l'écriture des mots u et v. On aura

$$u \sqcup v = \{w| \quad w = w_1 w_2 \dots w_M \text{ et } \quad \begin{array}{l} u = w_{a_1} w_{a_2} \dots w_{a_n}, \\ v = w_{b_1} w_{b_2} \dots w_{b_n}, \\ \{a_i\} \cup \{b_j\} = \{1, \dots M\}, \{a_i\} \cap \{b_j\} = \emptyset\}. \end{array}$$

Exemple : Si $u = ab$ et $v = cde$ alors on a :

$$u \sqcup v = \{abcde, acbde, cabde, acdbe, cadbe, cdabe, acdeb, cadeb, adaeb, cdeab\}$$

Le mélange de deux langages est alors l'ensemble des mélanges possibles de deux mots du langage :

$$L \sqcup L' = \{w \in \Sigma^*/w \in \{u \sqcup v\} \;\; u \in L \;\; v \in L'\}.$$

On a alors

Proposition 5.1.3

$$\mathcal{A}(L \sqcup L') \supseteq \mathcal{A}(L) \cup h(L) \circ \mathcal{A}(L') \cup \mathcal{A}(L') \cup h(L') \circ \mathcal{A}(L).$$

Démonstration : $L \sqcup L' \supseteq L.L'$, $L \sqcup L' \supseteq L'.L$ et Proposition 5.1.2. □

5.1.4 Intersection

L'intersection est une opération utilisée en CSG. Nous avons donc étudié son équivalent en Théorie des Langages. L'intersection de deux langages est donnée par :

$$L \cap L' = \{w \in \Sigma^*/w \in L \text{ et } w \in L'\}.$$

On a alors

Proposition 5.1.4 *Si* $\forall n \;\; (L \cap L')_n \neq \emptyset$ *alors*

$$\mathcal{A}(L \cap L') \subseteq \mathcal{A}(L) \cap \mathcal{A}(L').$$

Démonstration : $L \cap L'$ est un langage infini, donc $(h((L \cap L')_n) \circ K)$ est une suite de Cauchy, telle que

$$(L \cap L')_n \subseteq L_n \cap L'_n.$$

On a donc

$$\begin{array}{rcl} (L\cap L')_n & \subseteq & L_n \cap L'_n, \\ h((L\cap L')_n)\circ K & \subseteq & h(L_n\cap L'_n)\circ K \\ & \subseteq & h(L_n)\circ K \cap h(L'_n)\circ K, \\ \mathcal{A}(L\cap L') & = & \lim_{n\to\infty} h((L\cap L')_n)\circ K \\ & \subseteq & \lim_{n\to\infty} h(L_n)\circ K \cap \lim_{n\to\infty} h(L'_n)\circ K \\ & \subseteq & \mathcal{A}(L)\cap\mathcal{A}(L'). \end{array}$$

□

5.2 Cas d'un langage rationnel

Dans le cas de langages rationnels, nous allons avoir une méthode algorithmique pour construire le résultat de l'opération. En effet, la classe des langages rationnels est close par union, concaténation, mélange et intersection. De plus, il existe des méthodes permettant de construire les opérations sur les automates (resp. les matrices), correspondant à l'union, la concaténation, le mélange ou l'intersection de deux langages. Nous allons ensuite pouvoir visualiser l'attracteur associé à l'automate (resp. la matrice) résultat, en utilisant l'algorithme donné au Chapitre 3.

Notation : On notera, dans la suite, $\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$ et $\mathcal{M}' = (Q', \Sigma, \delta', Q'_I, Q'_F)$ deux automates, et $\mathcal{M}'' = (Q'', \Sigma, \delta'', Q''_I, Q''_F)$ l'automate résultat de l'opération que l'on étudie. On suppose l'alphabet commun à tous les automates et $Q\cap Q' = \emptyset$.

De même, on notera $(\tilde{H}, \tilde{I}, \tilde{F})$, $(\tilde{H}', \tilde{I}', \tilde{F}')$, et $(\tilde{H}'', \tilde{I}'', \tilde{F}'')$, les triplets matrices de Hutchinson, vecteurs initiaux et finaux correspondant aux automates. Les triplets sont construits comme décrit au Chapitre 3, c'est-à-dire :

$$\tilde{H} = h(\Delta(\mathcal{M})),$$

$$\tilde{I}_j = \begin{cases} \{\mathrm{Id}\} \text{ si } q_j \in Q_I \\ \emptyset \text{ si } q_j \notin Q_I \end{cases}, \quad \tilde{F}_j = \begin{cases} \{\mathrm{Id}\} \text{ si } q_j \in Q_F \\ \emptyset \text{ si } q_j \notin Q_F \end{cases},$$

avec $\Delta(\mathcal{M})$ la matrice d'adjacence de l'automate $\mathcal{M}$.

Les opérations sur les triplets $(\tilde{H}, \tilde{I}, \tilde{F})$ pourront s'écrire en général (sauf pour la concaténation)

$$(\tilde{H}, \tilde{I}, \tilde{F})\diamondsuit(\tilde{H}', \tilde{I}', \tilde{F}') = (\tilde{H}\diamondsuit\tilde{H}', \tilde{I}\diamondsuit\tilde{I}', \tilde{F}\diamondsuit\tilde{F}').$$

On les construit de telle sorte qu'elles correspondent aux opérations sur les langages :

$$\mathcal{A}(\mathcal{M}\odot\mathcal{M}') = \mathcal{A}((\tilde{H}, \tilde{I}, \tilde{F})\diamondsuit_{\odot}(\tilde{H}', \tilde{I}', \tilde{F}')) = \mathcal{A}(L(\mathcal{M})\odot L(\mathcal{M}')).$$

Pour cela, il suffit de définir $(\tilde{H}'', \tilde{I}'', \tilde{F}'')$ comme le triplet associé à $\mathcal{M}''$.

5.2.1 Union

L'automate $\mathcal{M}''$ acceptant $L(\mathcal{M}'') = L(\mathcal{M}) \cup L(\mathcal{M}')$ est :

$$\mathcal{M}'' = (Q'', \Sigma, \delta'', Q''_I, Q''_F),$$

où

$$\begin{array}{rcl} Q'' & = & Q \cup Q'; \\ Q''_I & = & Q_I \cup Q'_I; \\ Q''_F & = & Q_F \cup Q'_F; \\ \delta''(q,a) & = & \left\{ \begin{array}{l} \delta(q,a) \text{ si } q \in Q \\ \delta'(q,a) \text{ si } q \in Q' \end{array} \right. . \end{array}$$

Le triplet $(\tilde{H}'', \tilde{I}'', \tilde{F}'')$ associé à l'union de deux langages est donné par :

$$\tilde{H}'' = h(\Delta(\mathcal{M}'')).$$

La partie graphe de l'automate $\mathcal{M}''$ étant indépendante des états initiaux et finaux de $\mathcal{M}$ et $\mathcal{M}'$, on peut noter :

$$\tilde{H}'' = \tilde{H} \diamondsuit_\cup \tilde{H}',$$

avec $\tilde{H}''$ une matrice carrée de taille $n+m$ et

$$\tilde{H}''_{ij} = \left\{ \begin{array}{l} \tilde{H}_{ij} \text{ si } i,j \leq n \\ \tilde{H}'_{i-nj-n} \text{ si } n \leq i,j \\ \emptyset \text{ sinon} \end{array} \right. ,$$

c'est-à-dire

$$\tilde{H}'' = \begin{pmatrix} \tilde{H}_{11} & \dots & \tilde{H}_{1n} & \emptyset & \dots & \emptyset \\ \vdots & & \vdots & \vdots & & \vdots \\ \tilde{H}_{n1} & \dots & \tilde{H}_{nn} & \emptyset & \dots & \emptyset \\ \emptyset & \dots & \emptyset & \tilde{H}'_{11} & \dots & \tilde{H}'_{1m} \\ \vdots & & \vdots & \vdots & & \vdots \\ \emptyset & \dots & \emptyset & \tilde{H}'_{m1} & \dots & \tilde{H}'_{mm} \end{pmatrix},$$

et

$$\tilde{I}'' = \begin{pmatrix} \tilde{I}_1 \\ \vdots \\ \tilde{I}_n \\ \tilde{I}'_1 \\ \vdots \\ \tilde{I}'_m \end{pmatrix}, \quad \tilde{F}'' = \begin{pmatrix} \tilde{F}_1 \\ \vdots \\ \tilde{F}_n \\ \tilde{F}'_1 \\ \vdots \\ \tilde{F}'_m \end{pmatrix}.$$

Exemple : Soit l'IFS $\mathcal{T} = \{S_1, \ldots S_8\}$ avec

$$\begin{array}{rcl} S_1 & = & H(0.5); \\ S_2 & = & T(0.5,0,0) \circ H(0.5); \\ S_3 & = & T(0,0.5,0) \circ H(0.5); \\ S_4 & = & T(0,0,0.5) \circ H(0.5); \\ S_5 & = & T(0.5,0.5,0) \circ H(0.5); \\ S_6 & = & T(0.5,0,0.5) \circ H(0.5); \\ S_7 & = & T(0,0.5,0.5) \circ H(0.5); \\ S_8 & = & T(0.5,0.5,0.5) \circ H(0.5). \end{array}$$

Les S_i sont les transformations de l'octarbre (ou *octree*), c'est-à-dire un cube divisé en 8. On considère les deux langages :

$$L_1 = \{1,3,4\}^*,$$

$$L_2 = \{1,3,5\}^*.$$

L_1 est reconnu par un automate $\mathcal{M}_1$ et L_2 par un automate $\mathcal{M}_2$. Soit $L_u = L_1 \cup L_2$. L_u est reconnu par un automate $\mathcal{M}_u$. La Figure 5.1 donne les graphes d'automates de $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_u$ ainsi que les attracteurs associés.

En termes matriciels on note $(\tilde{H}_1, \tilde{I}_1, \tilde{F}_1)$, $(\tilde{H}_2, \tilde{I}_2, \tilde{F}_2)$, et $(\tilde{H}_u, \tilde{I}_u, \tilde{F}_u)$, les triplets associés à $\mathcal{M}_1$, $\mathcal{M}_2$, et $\mathcal{M}_u$. On a alors :

$$\begin{array}{rcl} \tilde{H}_1 & = & (\{T_1, T_3, T_4\}) \quad I_1 = (\mathrm{Id}) \quad F_1 = (\mathrm{Id}); \\ \tilde{H}_2 & = & (\{T_1, T_3, T_5\}) \quad I_2 = (\mathrm{Id}) \quad F_2 = (\mathrm{Id}); \\ \tilde{H}_u & = & \tilde{H}_1 \diamondsuit_{\cup} \tilde{H}_2 \\ & = & \left(\begin{array}{cc} \{T_1, T_3, T_4\} & \emptyset \\ \emptyset & \{T_1, T_3, T_5\} \end{array} \right); \\ \tilde{I}_u & = & \left(\begin{array}{c} \mathrm{Id} \\ \mathrm{Id} \end{array} \right); \\ \tilde{F}_u & = & \left(\begin{array}{c} \mathrm{Id} \\ \mathrm{Id} \end{array} \right). \end{array}$$

5.2.2 Concaténation

L'automate $\mathcal{M}''$ acceptant $L(\mathcal{M}).L(\mathcal{M}')$ est construit en connectant chaque état final de $\mathcal{M}$ à chaque état initial de $\mathcal{M}'$ par une ϵ-transition.

En termes matriciels, on construit la matrice de HUTCHINSON, résultat de la concaténation, de la même manière que pour l'union, en ajoutant

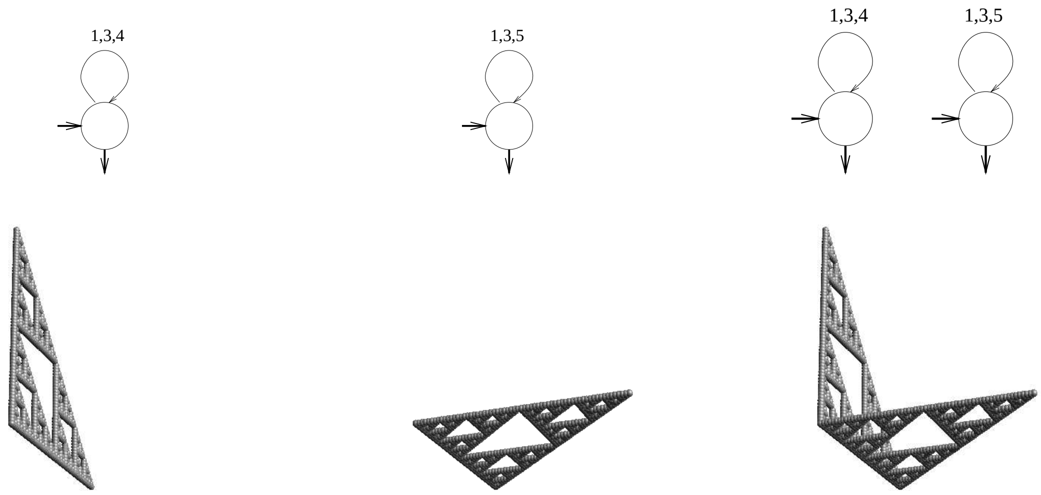


FIG. 5.1 - *Union*

l'identité aux coordonnées correspondant à un état final de $\mathcal{M}$ et à un état initial de $\mathcal{M}'$. On a cette fois une opération qui est dépendante des états initiaux et finaux. On écrira donc

$$(\tilde{H}'', \tilde{I}'', \tilde{F}'') = (\tilde{H}, \tilde{I}, \tilde{F}) \diamondsuit . (\tilde{H}', \tilde{I}', \tilde{F}'),$$

avec $\tilde{H}''$ une matrice carrée de taille $n+m$, et

$$\tilde{H}''_{ij} = \begin{cases} \tilde{H}_{ij} \text{ si } i, j \leq n \\ \tilde{H}'_{i-nj-n} \text{ si } n \leq i, j \\ \{\mathrm{Id}\} \text{ si } q_i \in Q_F \text{ et } q'_{j-n} \in Q'_I \\ \emptyset \text{ sinon} \end{cases},$$

et

$$\tilde{I}'' = \begin{pmatrix} \tilde{I}_1 \\ \vdots \\ \tilde{I}_n \\ \emptyset \\ \vdots \\ \emptyset \end{pmatrix}, \quad \tilde{F}'' = \begin{pmatrix} \emptyset \\ \vdots \\ \emptyset \\ \tilde{F}'_1 \\ \vdots \\ \tilde{F}'_m \end{pmatrix}.$$

Exemple : Soit $L_c = L_1.L_2$. Le langage L_c est reconnu par un automate $\mathcal{M}_c$. Les automates $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_c$, ainsi que les attracteurs associés, sont donnés Figure 5.2.

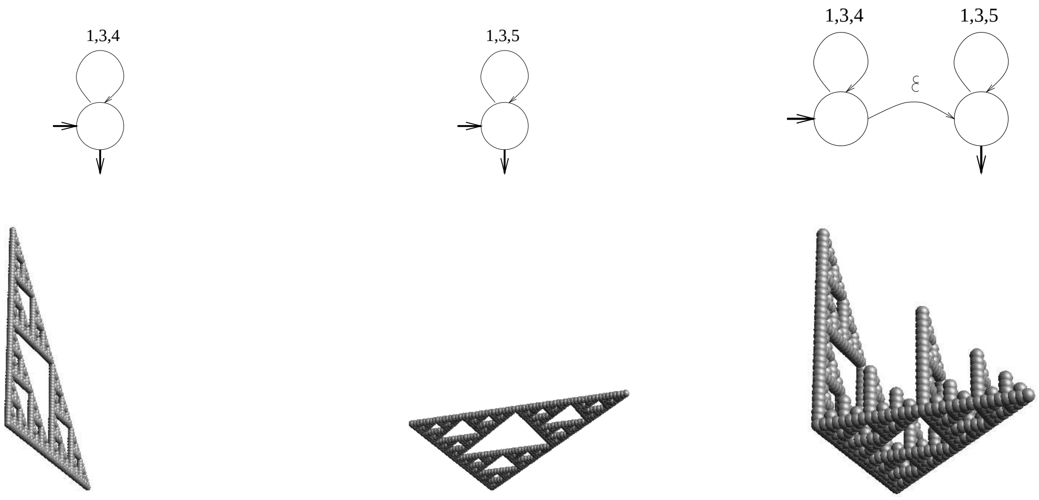


FIG. 5.2 - *Concaténation*

En termes matriciels on a alors :

$$\begin{aligned}
\tilde{H}_c &= \begin{pmatrix} \{T_1, T_3, T_4\} & \{\mathrm{Id}\} \\ \emptyset & \{T_1, T_3, T_5\} \end{pmatrix} ; \\
\tilde{I}_c &= \begin{pmatrix} \mathrm{Id} \\ \emptyset \end{pmatrix} ; \\
\tilde{F}_c &= \begin{pmatrix} \emptyset \\ \mathrm{Id} \end{pmatrix} .
\end{aligned}$$

Interprétation géométrique : L'attracteur résultat de la concaténation peut être vu comme l'attracteur associé au premier langage fabriqué avec des itérations de l'attracteur associé au deuxième langage. Sur l'exemple, on a un triangle de SIERPINSKY horizontal construit avec diverses itérations d'un triangle de SIERPINSKY vertical.

5.2.3 Mélange

L'automate $\mathcal{M}''$, acceptant $L(\mathcal{M}'') = L(\mathcal{M}) \sqcup L(\mathcal{M}')$, est :

$$\mathcal{M}'' = (Q'', \Sigma, \delta'', Q''_I, Q''_F),$$

où

$$\begin{array}{rcl} Q'' & = & Q \times Q'; \\ Q''_I & = & Q_I \times Q' \cup Q \times Q'_I; \\ Q''_F & = & Q_F \times Q' \cup Q \times Q'_F; \\ \delta''((q,q'),a) & = & \delta(q,a) \times \{q'\} \cup \{q\} \times \delta(q',a). \end{array}$$

En termes matriciels, la matrice de HUTCHINSON associée au mélange de deux matrices de HUTCHINSON correspond à une somme tensorielle des matrices, selon l'opération union. On a :

$$\tilde{H}'' = \tilde{H} \diamondsuit_{\sqcup} \tilde{H}' = \begin{pmatrix} \widetilde{\mathrm{Id}}.\tilde{H}_{11} \cup \tilde{H}' & \widetilde{\mathrm{Id}}.\tilde{H}_{12} & \dots & \widetilde{\mathrm{Id}}.\tilde{H}_{1n} \\ \widetilde{\mathrm{Id}}.\tilde{H}_{21} & \widetilde{\mathrm{Id}}.\tilde{H}_{22} \cup \tilde{H}' & \dots & \widetilde{\mathrm{Id}}.\tilde{H}_{2n} \\ \vdots & & & \vdots \\ \widetilde{\mathrm{Id}}.\tilde{H}_{n1} & \widetilde{\mathrm{Id}}.\tilde{H}_{n2} & \dots & \widetilde{\mathrm{Id}}.\tilde{H}_{nn} \cup \tilde{H}' \end{pmatrix},$$

avec $\tilde{H}''$ une matrice carrée de taille $n.m$ et $\widetilde{\mathrm{Id}}$ la matrice identité de taille m.

Ainsi,

$$\tilde{H}''_{i+km,j+k'm} = \delta_{kk'}\tilde{H}'_{ij} \cup \delta_{ij}\tilde{H}_{kk'},$$

si l'on note $\delta_{i,j}$ une fonction qui vaut Id si $i = j$, et $\emptyset$ sinon.

Les vecteurs initiaux et finaux seront construits de la manière suivante :

$$\tilde{I}'' = \begin{pmatrix} \tilde{I}_1 \cup \tilde{I}' \\ \vdots \\ \tilde{I}_n \cup \tilde{I}' \end{pmatrix},$$

avec

$$\tilde{I}_i \cup \tilde{I}' = \begin{pmatrix} \tilde{I}_i \cup \tilde{I}'_1 \\ \vdots \\ \tilde{I}_i \cup \tilde{I}'_m \end{pmatrix};$$

et

$$\tilde{F}'' = \begin{pmatrix} \tilde{F}_1 \cup \tilde{F}' \\ \vdots \\ \tilde{F}_n \cup \tilde{F}' \end{pmatrix},$$

avec

$$\tilde{F}_i \cup \tilde{F}' = \begin{pmatrix} \tilde{F}_i \cup \tilde{F}'_1 \\ \vdots \\ \tilde{F}_i \cup \tilde{F}'_m \end{pmatrix}.$$

Exemple : Soit $L_m = L_1 \sqcup L_2$. Le langage L_m est reconnu par un automate $\mathcal{M}_m$. Les automates $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_m$, ainsi que les attracteurs associés sont donnés Figure 5.3.

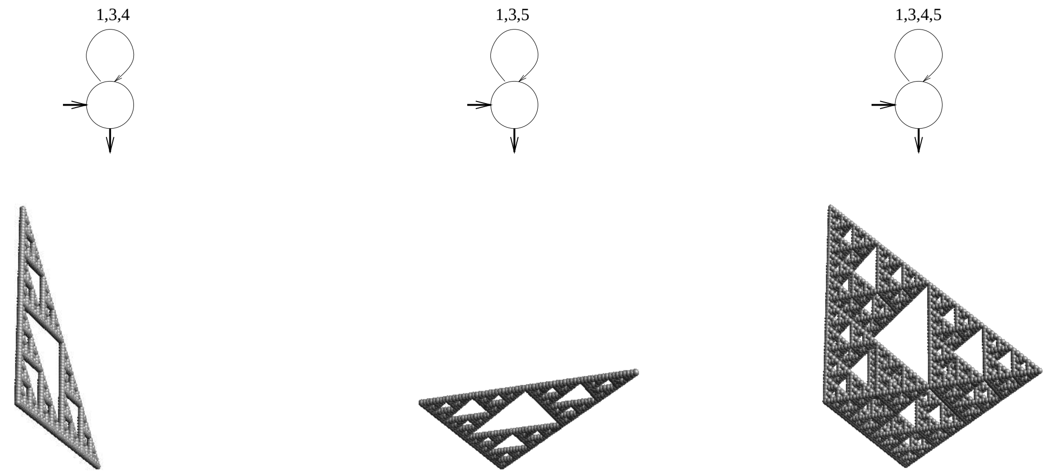


FIG. 5.3 - *Mélange*

En termes matriciels on a alors :

$$\begin{aligned}\tilde{H}_m &= \tilde{H}_1 \diamondsuit_{\sqcup} \tilde{H}_2 \\ &= (\{T_1, T_3, T_4, T_5\})\,; \\ \tilde{I}_m &= (\mathrm{Id})\,; \\ \tilde{F}_m &= (\mathrm{Id})\,.\end{aligned}$$

Remarque : Comme on peut le voir sur l'exemple précédent, le mélange de deux IFS est égal à l'union des deux ensembles de transformations des IFS.

Interprétation géométrique : Le mélange est une opération qui va remplir l'espace entre les deux attracteurs de départ. Dans l'exemple précédent, on obtient une pyramide.

5.2.4 Intersection

Dans le cas d'un langage rationnel, l'automate $\mathcal{M}''$ acceptant $L(\mathcal{M}'') = L(\mathcal{M}) \cap L(\mathcal{M}')$ est :

$$\mathcal{M}'' = (Q'', \Sigma, \delta'', Q''_I, Q''_F),$$

où

$$\begin{array}{rcl} Q'' & = & Q \times Q'; \\ Q''_I & = & Q_I \times Q'_I; \\ Q''_F & = & Q_F \times Q'_F; \\ \delta''((q_i, q_j), a) & = & (q_k, q_l) \text{ si } \delta(q_i, a) = q_k \text{ et } \delta'(q_j, a) = q_l. \end{array}$$

En termes matriciels, la matrice de HUTCHINSON associée à l'intersection langage de deux matrices de HUTCHINSON, correspond à un produit tensoriel des matrices selon l'opération intersection. On a :

$$\tilde{H}'' = \tilde{H} \Diamond_\cap \tilde{H}' = \begin{pmatrix} \tilde{H}_{11} \cap \tilde{H}' & \dots & \tilde{H}_{1n} \cap \tilde{H}' \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cap \tilde{H}' & \dots & \tilde{H}_{nn} \cap \tilde{H}' \end{pmatrix},$$

avec $\tilde{H}''$ une matrice carrée de taille $n.m$ et

$$\tilde{H}_{ij} \cap \tilde{H}' = \begin{pmatrix} \tilde{H}_{ij} \cap \tilde{H}'_{11} & \dots & \tilde{H}_{1n} \cap \tilde{H}'_{1m} \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cap \tilde{H}'_{m1} & \dots & \tilde{H}_{nn} \cap \tilde{H}'_{mm} \end{pmatrix}.$$

Les vecteurs initiaux et finaux seront construits de la manière suivante :

$$\tilde{I}'' = \begin{pmatrix} \tilde{I}_1 \cup \tilde{I}' \\ \vdots \\ \tilde{I}_n \cup \tilde{I}' \end{pmatrix},$$

avec

$$\tilde{I}_i \cap \tilde{I}' = \begin{pmatrix} \tilde{I}_i \cap \tilde{I}'_1 \\ \vdots \\ \tilde{I}_i \cap \tilde{I}'_m \end{pmatrix};$$

et

$$\tilde{F}'' = \begin{pmatrix} \tilde{F}_1 \cap \tilde{F}' \\ \vdots \\ \tilde{F}_n \cap \tilde{F}' \end{pmatrix},$$

avec

$$\tilde{F}_i \cap \tilde{F}' = \begin{pmatrix} \tilde{F}_i \cap \tilde{F}'_1 \\ \vdots \\ \tilde{F}_i \cap \tilde{F}'_m \end{pmatrix}.$$

Exemple : Soit $L_i = L_1 \cap L_2$. Le langage L_i est reconnu par un automate $\mathcal{M}_i$. Les automates $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_i$, ainsi que les attracteurs associés sont donnés Figure 5.4.

En termes matriciels, on a alors :

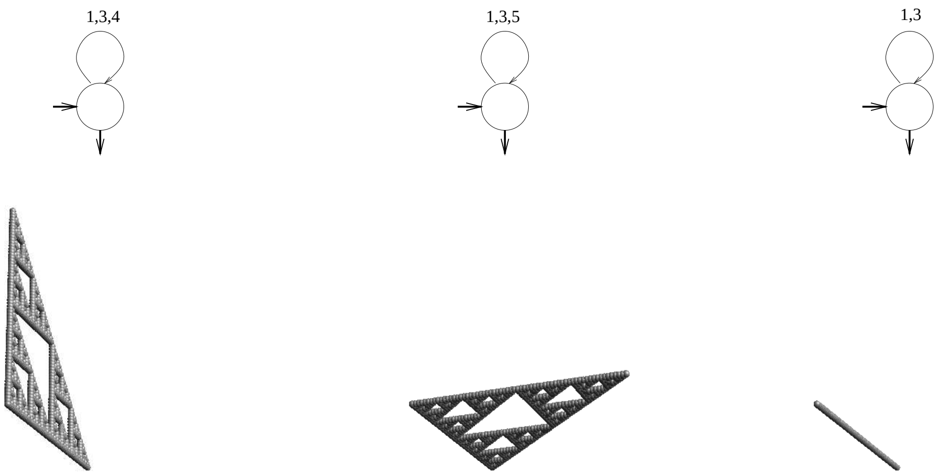


FIG. 5.4 - *Intersection*

$$\begin{aligned}\tilde{H}_i &= \tilde{H}_1 \Diamond_{\cap} \tilde{H}_2 \\ &= (\{T_1, T_3\})\,; \\ \tilde{I}_i &= (\mathrm{Id})\,; \\ \tilde{F}_i &= (\mathrm{Id})\,.\end{aligned}$$

Remarque : Comme on peut le voir sur l'exemple précédent, l'intersection "langage" de deux IFS est égale à l'intersection des deux ensembles de transformations des IFS.

Remarque : L'intersection langage n'est pas un équivalent de l'intersection classique, c'est-à-dire de l'intersection ensembliste. En effet, deux langages différents peuvent donner le même attracteur. Le résultat de l'intersection va donc dépendre du choix des langages.

Par exemple, si $T_1 = H(0.5)$ et $T_2 = H(0.3)$. Si $L(\mathcal{M}) = \{1\}^*$ et $L(\mathcal{M}') = \{2\}^*$, alors $\mathcal{A}(\mathcal{M}) = \mathcal{A}(\mathcal{M}') = \{O\}$: l'origine du système de coordonnées. Donc $L(\mathcal{M}) \cap L(\mathcal{M}') = \emptyset$ alors que $\mathcal{A}(\mathcal{M}) = \mathcal{A}(\mathcal{M}')$.

5.2.5 Exemples

Nous donnons quelques exemples obtenus par concaténation ou mélange. Pour cela, nous avons utilisé, en plus des transformations de l'octarbre, les transformations suivantes :

$$\begin{aligned}
S_9 &= T(0,1,0);\\
S_{10} &= H(1/3);\\
S_{11} &= H(1/3)\circ T(1,0,0)\circ Ry(\pi/3);\\
S_{12} &= S_{11}\circ T(1,0,0)\circ Ry(-2\pi/3);\\
S_{13} &= H(1/3)\circ T(2,0,0);\\
S_{14} &= T(0,0,1)\circ Ry(\pi/4)\circ H(1/2);\\
S_{15} &= T(0,0,1)\circ Ry(-\pi/4)\circ H(1/2).
\end{aligned}$$

La Figure 5.5 montre un exemple de concaténation :

$$L_1 = \{1,4,5\}^*,\ L_1' = \{1,2,4\}^*.9$$

et un exemple de mélange :

$$L_2 = \{1,4\}^*.\{14,15\}^*,\ L_2' = \{1,4\}^*.\{14,15\}^*.9.$$

Ces exemples sont une illustration du découpage en tranche qui apparait lorsque l'on travaille sur deux LRIFS 2D "face-à-face". Cet effet permet de faire une extrusion à condition de choisir une primitive suffisamment "large" ce qui permet de gommer les tranches.

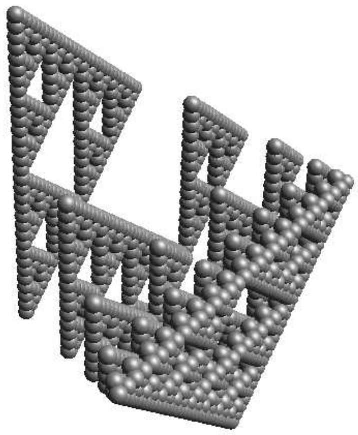
La Figure 5.6 montre une autre utilisation de la concaténation :

$$L_3 = \{1,4,7\}^*,\ L_3' = \{10,11,12,13\}^*.$$

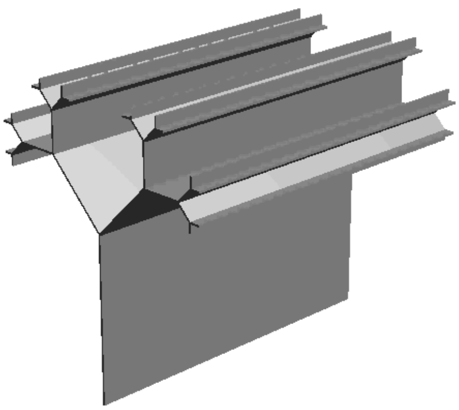
En effet, si l'on concatène un LRIFS avec une courbe fractale (ici, un triangle de SIERPINSKY avec une courbe de type VON KOCH), les itérations de l'attracteur du LRIFS vont suivre la courbe.

5.3 Opérations directes sur les matrices

Une autre manière de combiner deux fractales est de combiner deux matrices de HUTCHINSON, et de construire l'attracteur associé à la matrice résultat. Nous donnons ici la liste des opérations que nous considérons. Nous verrons, dans la partie suivante, une comparaison entre ces opérations et celles sur les automates.

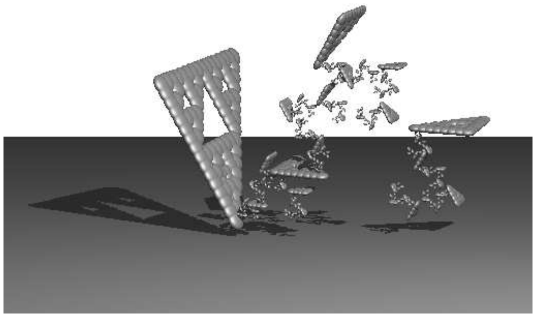


$L_1.L'_1$



$L_2 \sqcup L'_2$

FIG. 5.5 - *Exemple de concaténation et mélange*



$L_3.L_3'$

FIG. 5.6 - *Exemple de concaténation et mélange*

5.3.1 Union directe

L'union directe de deux matrices est donnée par :

$$\tilde{H} \cup \tilde{H}' = \begin{pmatrix} \tilde{H}_{11} \cup \tilde{H}'_{11} & \dots & \tilde{H}_{1n} \cup \tilde{H}'_{1n} \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cup \tilde{H}'_{n1} & \dots & \tilde{H}_{nn} \cup \tilde{H}'_{nn} \end{pmatrix}.$$

5.3.2 Composition directe

La composition directe de deux matrices est donnée par :

$$\tilde{H} \circ \tilde{H}' = \tilde{H}'',$$

avec

$$\tilde{H}_{ij} = \bigcup_k \tilde{H}_{ik} \circ \tilde{H}'_{kj}.$$

5.3.3 Intersection directe

L'intersection directe de deux matrices est donnée par :

$$\tilde{H} \cap \tilde{H}' = \begin{pmatrix} \tilde{H}_{11} \cap \tilde{H}'_{11} & \dots & \tilde{H}_{1n} \cap \tilde{H}'_{1n} \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cap \tilde{H}'_{n1} & \dots & \tilde{H}_{nn} \cap \tilde{H}'_{nn} \end{pmatrix}.$$

5.3.4 Produit tensoriel

Le produit tensoriel de deux matrices est donné par :

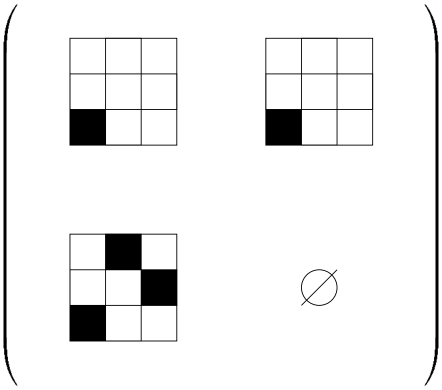
$$\tilde{H}'' = \tilde{H} \otimes_{\cup} \tilde{H}' = \begin{pmatrix} \tilde{H}_{11} \cup \tilde{H}' & \dots & \tilde{H}_{1n} \cup \tilde{H}' \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cup \tilde{H}' & \dots & \tilde{H}_{nn} \cup \tilde{H}' \end{pmatrix},$$

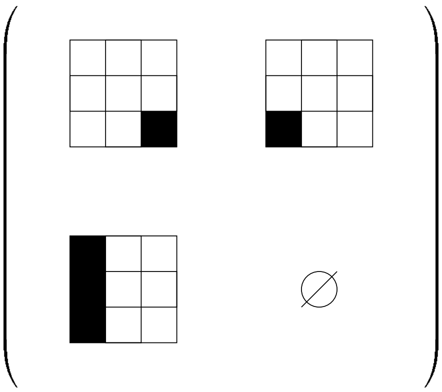
avec $\tilde{H}''$ une matrice carrée de taille $n.m$, et

$$\tilde{H}_{ij} \cup \tilde{H}' = \begin{pmatrix} \tilde{H}_{ij} \cup \tilde{H}'_{11} & \dots & \tilde{H}_{1n} \cup \tilde{H}'_{1m} \\ \vdots & & \vdots \\ \tilde{H}_{n1} \cup \tilde{H}'_{m1} & \dots & \tilde{H}_{nn} \cup \tilde{H}'_{mm} \end{pmatrix}.$$

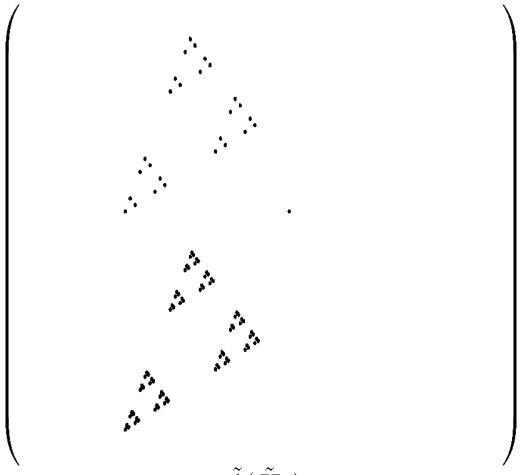
5.3.5 Exemple

Soient les deux matrices de HUTCHINSON :

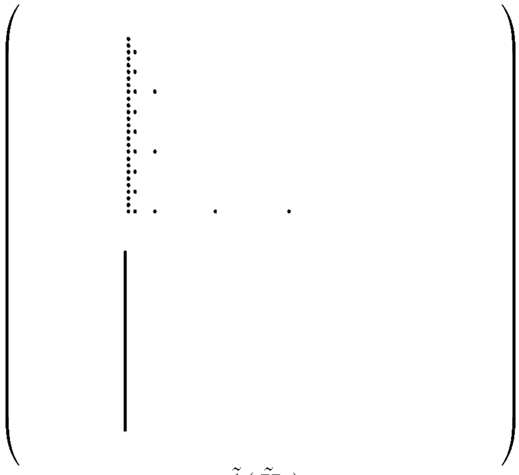
$\tilde{H}_1 =$  ,

$\tilde{H}_2 =$  .

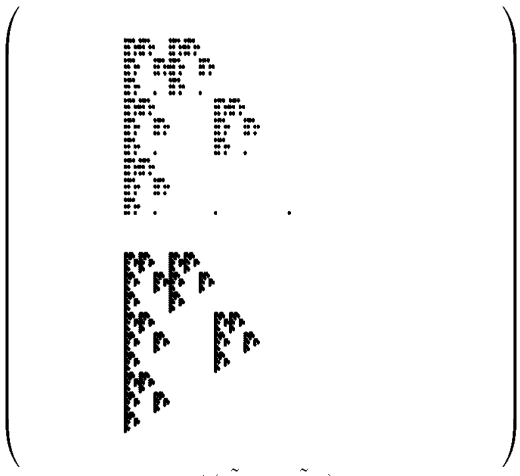
Les transformations utilisées sont celles de la grille 3×3. Les carrés noircis correspondent aux transformations sélectionnées. Les attracteurs correspondants et le résultat des diverses opérations sont donnés ci-dessous. Nous ajoutons à cette liste l'attracteur matriciel correspondant à la concaténation et à l'union langage.



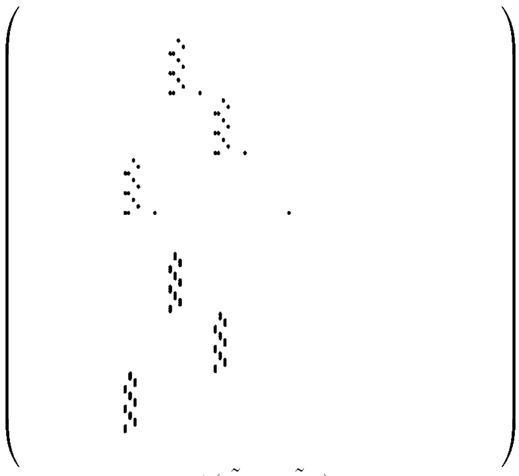
$\tilde{A}(\tilde{H}_1)$



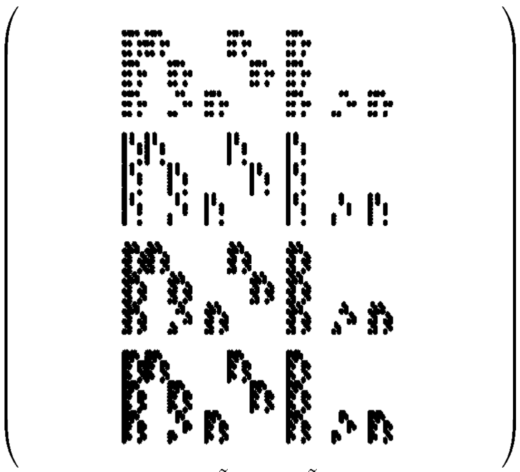
$\tilde{A}(\tilde{H}_2)$



$\mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2)$

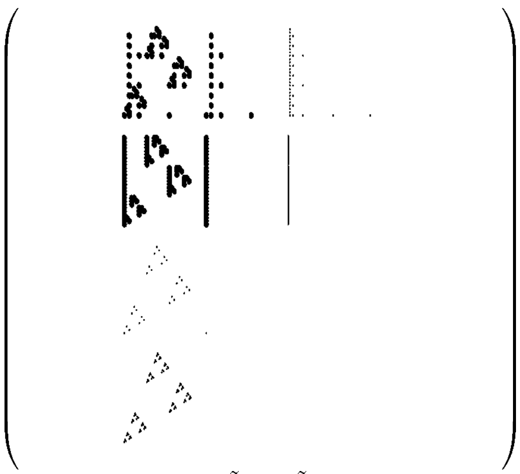


$A(\tilde{H}_1 \circ \tilde{H}_2)$

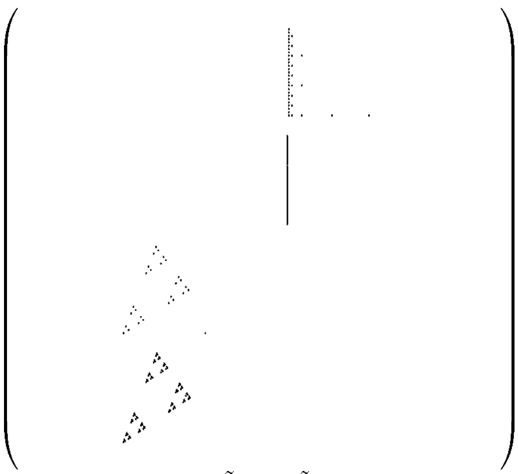


$\mathcal{A}(\tilde{H}_1 \otimes_{\cup} \tilde{H}_2)$

$\mathcal{A}(\tilde{H}_1 \cap \tilde{H}_2)$



$\mathcal{A}(\tilde{H}_1 \diamondsuit . \tilde{H}_2)$



$\mathcal{A}(\tilde{H}_1 \diamondsuit_{\cup} \tilde{H}_2)$

5.4 Comparaison entre les opérations

Nous avons défini des opérations, sur les langages et sur les matrices, permettant de composer deux attracteurs. Nous allons maintenant comparer le résultat de ces opérations en terme d'inclusion. En effet, il est intéressant de savoir quelle opération utiliser pour obtenir un effet donné. En particulier, on aimerait savoir comment obtenir un attracteur plus ou moins "rempli". Nous allons,pour cela, ordonner le résultat des différentes opérations, sachant que tous les attracteurs que nous obtenons sont des sous-ensemble d'un même IFS : l'IFS correspondant à l'union des deux alphabets dans le cas de deux langages, et l'IFS correspondant à l'ensemble des transformations utilisées dans le cas de deux matrices.

5.4.1 Relation d'ordre

Pour pouvoir comparer les opérations issues des langages et celles issues des matrices, nous allons considérer uniquement l'attracteur scalaire. Dans le premier cas, cet attracteur correspond à des langages dans lesquels tous les états sont initiaux et finaux.

Plus précisément, soit Δ la matrice d'adjacence d'un automate $\mathcal{M}$ tel que tous les états sont initiaux et finaux. On considère L le langage reconnu par $\mathcal{M}$ et $\tilde{H} = h(\Delta)$ la matrice de HUTCHINSON associée à $\mathcal{M}$. On a bien alors

$$\mathcal{A}(L) = \mathcal{A}(\mathcal{M}) = \mathcal{A}(\tilde{H}).$$

Ainsi, on pourra comparer les diverses opérations entre elles, qu'elles soient définies sur les langages ou sur les matrices. On notera L_1, L_2, deux langages rationnels, et $\tilde{H}_1, \tilde{H}_2$, deux matrices de HUTCHINSON leur correspondant.

Proposition 5.4.1 *On a les propriétés suivantes :*

- $L_1 \subseteq L_2 \Rightarrow \mathcal{A}(L_1) \subseteq \mathcal{A}(L_2)$.
- $\mathcal{A}(\tilde{H}^n) = \mathcal{A}(\tilde{H})$.
- *En posant* $\tilde{H}_1 \subseteq \tilde{H}_2 \Leftrightarrow \forall i, j\ (\tilde{H}_1)_{ij} \subseteq (\tilde{H}_2)_{ij}$, *on a*

$$\tilde{H}_1 \subseteq \tilde{H}_2 \Rightarrow \mathcal{A}(\tilde{H}_1) \subseteq \mathcal{A}(\tilde{H}_2).$$

Démonstration : Les preuves viennent directement de la définition de l'attracteur.

□

On a aussi le lemme suivant :

Lemme 5.4.1 *Soient* $\tilde{H}_1$ *et* $\tilde{H}_2$ *deux matrices de* HUTCHINSON *de tailles respectivement* $n_1 < n_2$. *On dit que* $\tilde{H}_1$ *est une matrice* extraite *de* $\tilde{H}_2$ *s'il existe*

$$1 \leq \alpha_1 < \alpha_2 < \ldots < \alpha_{n_1} \leq n_2,$$

de telle sorte que

$$(\tilde{H}_1)_{ij} = (\tilde{H}_2)_{\alpha_i \alpha_j}.$$

Dans ce cas, on a alors

$$\mathcal{A}(\tilde{H}_1) \subseteq \mathcal{A}(\tilde{H}_2).$$

Démonstration : En termes de graphes, l'hypothèse du lemme signifie que $G(\tilde{H}_1)$ est la restriction aux sommets $\alpha_1, \alpha_2, \ldots, \alpha_{n_1}$ de $G(\tilde{H}_2)$.

En termes matriciels, il suffit de montrer que

$$(\tilde{H}_1^n)_{ij} \subseteq (\tilde{H}_2^n)_{\alpha_i \alpha_j}.$$

Cette démonstration se fait par récurrence. Par hypothèse, le résultat est vrai pour $n = 1$. Pour $n > 1$ on a

$$\tilde{H}_1^{n+1} = \tilde{H}_1^n \circ \tilde{H}_1.$$

Pour $i, j \leq n_1$, on a

$$\begin{aligned}
(\tilde{H}_1^{n+1})_{ij} &= \bigcup_{k=1}^{n_1} (\tilde{H}_1^n)_{ik} \circ (\tilde{H}_1)_{kj} \\
&\subseteq \bigcup_{k=1}^{n_1} (\tilde{H}_2^n)_{\alpha_i \alpha_k} \circ (\tilde{H}_2)_{\alpha_k \alpha_j} \\
&\subseteq \bigcup_{k=1}^{n_2} (\tilde{H}_2^n)_{\alpha_i \alpha_k} \circ (\tilde{H}_2)_{\alpha_k \alpha_j} \\
&\subseteq (\tilde{H}_2^{n+1})_{\alpha_i \alpha_j}.
\end{aligned}$$

Ainsi,

$$\begin{aligned}
\tilde{\mathcal{A}}_{ij}(\tilde{H}_1) = \lim (\tilde{H}_1^n)_{ij} \circ K &\subseteq \lim (\tilde{H}_2^n)_{\alpha_i \alpha_j} \circ K = \tilde{\mathcal{A}}_{\alpha_i \alpha_j}(\tilde{H}_2), \\
\bigcup_{ij} \tilde{\mathcal{A}}_{ij}(\tilde{H}_1) &\subseteq \bigcup_{ij} \tilde{\mathcal{A}}_{\alpha_i \alpha_j}(\tilde{H}_2) \\
&\subseteq \bigcup_{ij} \tilde{\mathcal{A}}_{ij}(\tilde{H}_2), \\
\tilde{\mathcal{A}}(\tilde{H}_1) &\subseteq \tilde{\mathcal{A}}(\tilde{H}_2).
\end{aligned}$$

□

5.4.2 Relation entre les opérations

Proposition 5.4.2

$$\mathcal{A}(L_1 \cup L_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

Démonstration :

$$\begin{aligned}\tilde{H}_1 &\subseteq \tilde{H}_1 \cup \tilde{H}_2 \text{ et}\\ \tilde{H}_2 &\subseteq \tilde{H}_1 \cup \tilde{H}_2,\end{aligned}$$

donc

$$\begin{aligned}\mathcal{A}(\tilde{H}_1) &\subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2) \text{ et}\\ \mathcal{A}(\tilde{H}_2) &\subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2),\end{aligned}$$

et donc

$$\mathcal{A}(\tilde{H}_1) \cup \mathcal{A}(\tilde{H}_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

On a supposé que $\mathcal{A}(\tilde{H}_i) = \mathcal{A}(L_i)$, donc

$$\mathcal{A}(L_1) \cup \mathcal{A}(L_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

On a démontré que

$$\mathcal{A}(L_1) \cup \mathcal{A}(L_2) = \mathcal{A}(L_1 \cup L_2),$$

on a donc bien

$$\mathcal{A}(L_1 \cup L_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

□

Proposition 5.4.3

$$\mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2) \subseteq \mathcal{A}(\tilde{H}_1 \otimes_{\cup} \tilde{H}_2).$$

Démonstration : La matrice $\tilde{H}_1 \cup \tilde{H}_2$ est une matrice extraite de $\tilde{H}_1 \otimes_{\cup} \tilde{H}_2$, avec :

- $\tilde{H}_1 \cup \tilde{H}_2$ une matrice de taille n;
- $\tilde{H}_1 \otimes_{\cup} \tilde{H}_2$ une matrice de taille n^2;
- $\alpha_i = (i-1)n + i$.

On a donc

$$\mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2) \subseteq \mathcal{A}(\tilde{H}_1 \otimes_{\cup} \tilde{H}_2).$$

□

Proposition 5.4.4

$$\mathcal{A}(\tilde{H}_1 \cap \tilde{H}_2) \subseteq \mathcal{A}(L_1 \cap L_2).$$

Démonstration : La démonstration est semblable à celle que nous venons de voir car $\tilde{H}_1 \cap \tilde{H}_2$ et $\tilde{H}_1 \Diamond_\cap \tilde{H}_2$ vérifient les hypothèses du lemme 5.4.1. □

Proposition 5.4.5 *Si le mot vide appartient aux langages L_1 et L_2, alors*

$$\mathcal{A}(L_1 \cup L_2) \subseteq \mathcal{A}(L_1.L_2) \subseteq \mathcal{A}(L_1 \sqcup L_2).$$

Démonstration : Si $\epsilon \in L_1 \cap L_2$, alors

$$L_1 \cup L_2 \subseteq L_1.L_2 \subseteq L_1 \sqcup L_2.$$

□

Proposition 5.4.6

$$\mathcal{A}(\tilde{H}_1 \circ \tilde{H}_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

Démonstration :

$$\mathcal{A}(\tilde{H}_1 \circ \tilde{H}_2) \subseteq \mathcal{A}((\tilde{H}_1 \cup \tilde{H}_2)^2) = \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

□

5.4.3 Résumé

En résumé, on a la suite d'inclusions suivante :

$$\begin{array}{ll} & \mathcal{A}(\tilde{H}_1 \cap \tilde{H}_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1 \Diamond_\cap \tilde{H}_2) = \mathcal{A}(L_1 \cap L_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1) \cap \mathcal{A}(\tilde{H}_2) = \mathcal{A}(L_1) \cap \mathcal{A}(L_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1) \cup \mathcal{A}(\tilde{H}_2) = \mathcal{A}(\tilde{H}_1 \Diamond_\cup \tilde{H}_2) = \mathcal{A}(L_1 \cup L_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1 \otimes \tilde{H}_2). \end{array}$$

De plus, si $\epsilon \in L_1 \cap L_2$ alors

$$\begin{array}{ll} & \mathcal{A}(\tilde{H}_1 \Diamond_\cup \tilde{H}_2) = \mathcal{A}(L_1 \cup L_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1 \Diamond . \tilde{H}_2) = \mathcal{A}(L_1.L_2) \\ \subseteq & \mathcal{A}(\tilde{H}_1 \Diamond_\sqcup \tilde{H}_2) = \mathcal{A}(L_1 \sqcup L_2), \end{array}$$

et

$$\mathcal{A}(\tilde{H}_1 \circ \tilde{H}_2) \subseteq \mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2).$$

Ce qui donne le diagramme suivant :

$$\begin{tikzcd}
 & & \boxed{\mathcal{A}(\tilde{H}_1 \cap \tilde{H}_2)} \arrow[d] \\
 & & \boxed{\mathcal{A}(\tilde{H}_1 \diamondsuit_{\cap} \tilde{H}_2)} \arrow[d] \\
 & & \boxed{\mathcal{A}(\tilde{H}_1) \cap \mathcal{A}(\tilde{H}_2)} \arrow[d] \\
\boxed{\mathcal{A}(\tilde{H}_1 \circ \tilde{H}_2)} \arrow[d] & & \boxed{\mathcal{A}(\tilde{H}_1) \cup \mathcal{A}(\tilde{H}_2) = \mathcal{A}(\tilde{H}_1 \diamondsuit_{\cup} \tilde{H}_2)} \arrow[dll] \arrow[d] \\
\boxed{\mathcal{A}(\tilde{H}_1 \cup \tilde{H}_2)} \arrow[d] & & \boxed{\mathcal{A}(\tilde{H}_1 \diamondsuit_{.} \tilde{H}_2)} \arrow[d] \\
\boxed{\mathcal{A}(\tilde{H}_1 \otimes \tilde{H}_2)} \arrow[dr] & & \boxed{\mathcal{A}(\tilde{H}_1 \diamondsuit_{\sqcup} \tilde{H}_2)} \arrow[dl] \\
 & \boxed{\mathcal{A}(\mathcal{T})} &
\end{tikzcd}$$

5.4.4 Exemple

La Figure 5.7 donne un exemple de comparaison d'attracteurs.

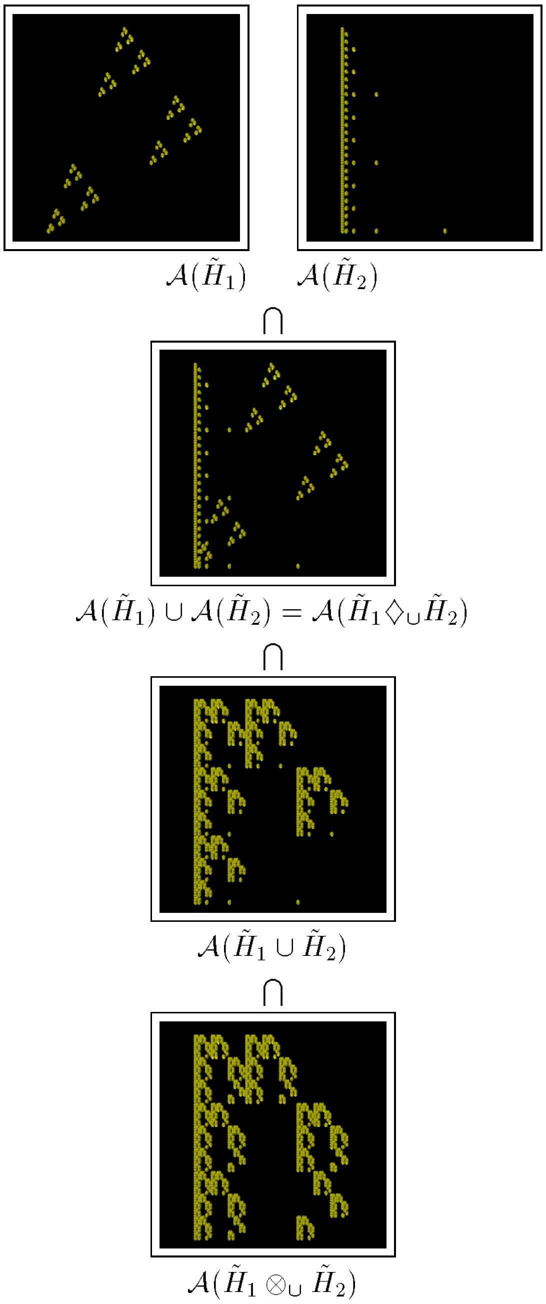


FIG. 5.7 - *Comparaisons d'attracteurs.*

Chapitre 6

Géométrie constructive

6.1 Système de géométrie fractale constructive

Les modèles que nous avons présentés nous ont permis de développer des opérations, ouvrant ainsi la voie à une géométrie fractale constructive. Nous discutons dans cette partie des possibilités offertes par notre approche et des problèmes spécifiques qu'elle engendre.

6.1.1 Arbre de construction

Nous avons défini des opérations permettant de combiner deux attracteurs. Ces opérations vont nous permettre de fabriquer des arbres de constructions. Ainsi, une scène pourra être construite "pas à pas".

Exemple : En utilisant l'IFS $\mathcal{T} = \{S_1, \ldots, S_8\}$, donné au chapitre précédent, nous donnons Figure 6.1 l'arbre de construction des langages, et Figure 6.2 l'arbre de construction des attracteurs associés.

En utilisant des opérations langages et des opérations matrices, on va pouvoir disposer d'une gamme assez complète d'opérations. Les figures 6.3 et 6.4 donne d'autres exemples d'arbres de construction. On peut remarquer que ces deux exemples qui sont fondés sur les tranformations d'une grille 3×3 donnent des formes très fouillées à partir d'attracteurs qui sont de simples IFS.

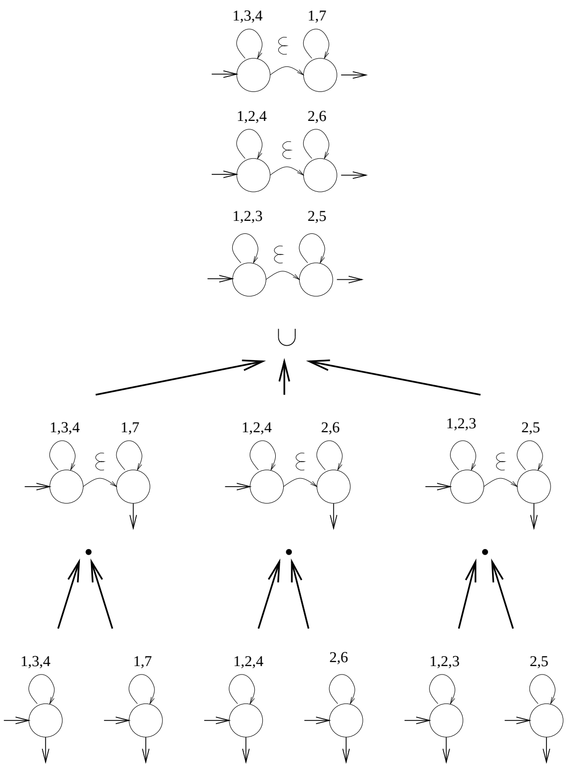


FIG. 6.1 - *Arbre de construction des langages.*

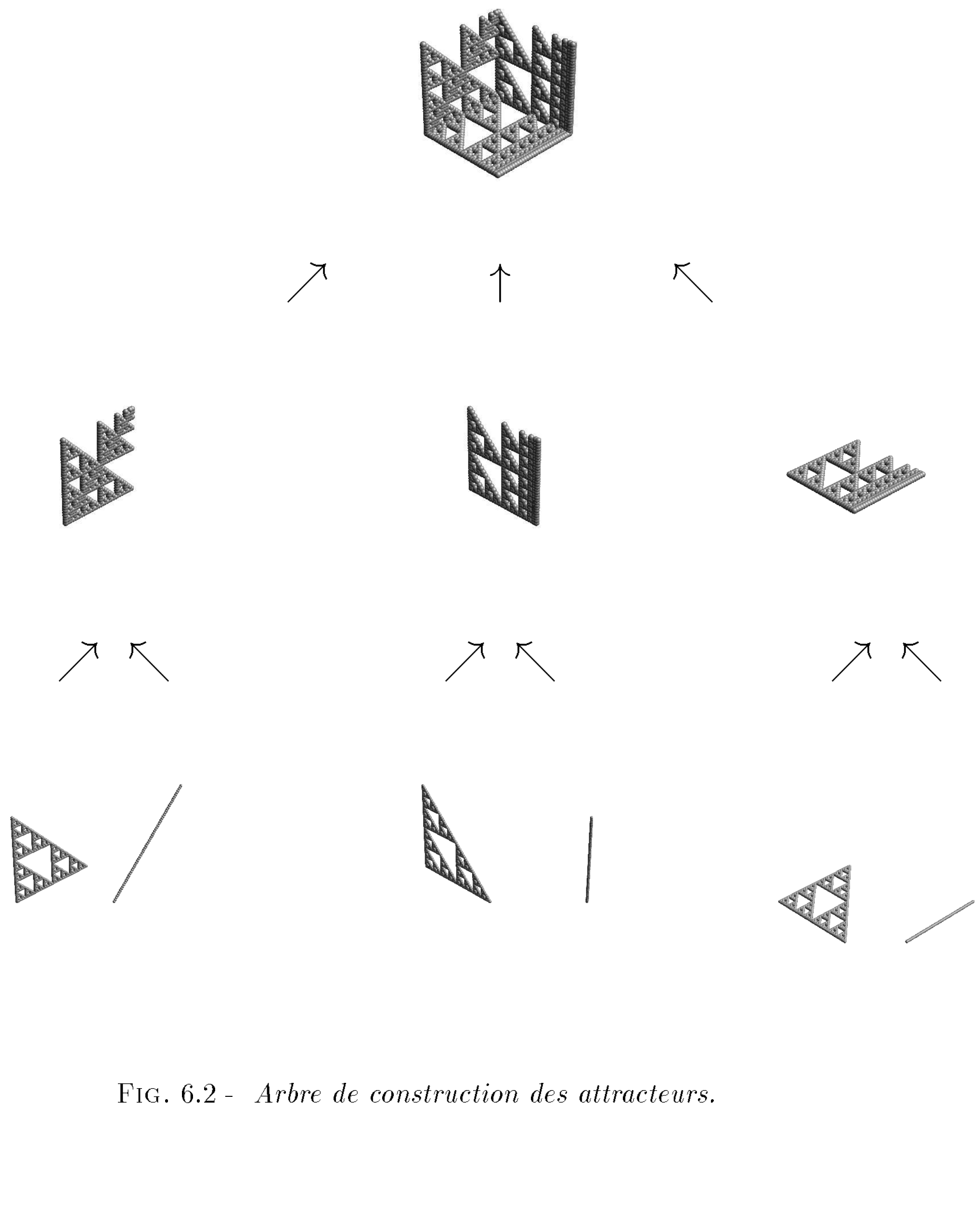


FIG. 6.2 - *Arbre de construction des attracteurs.*

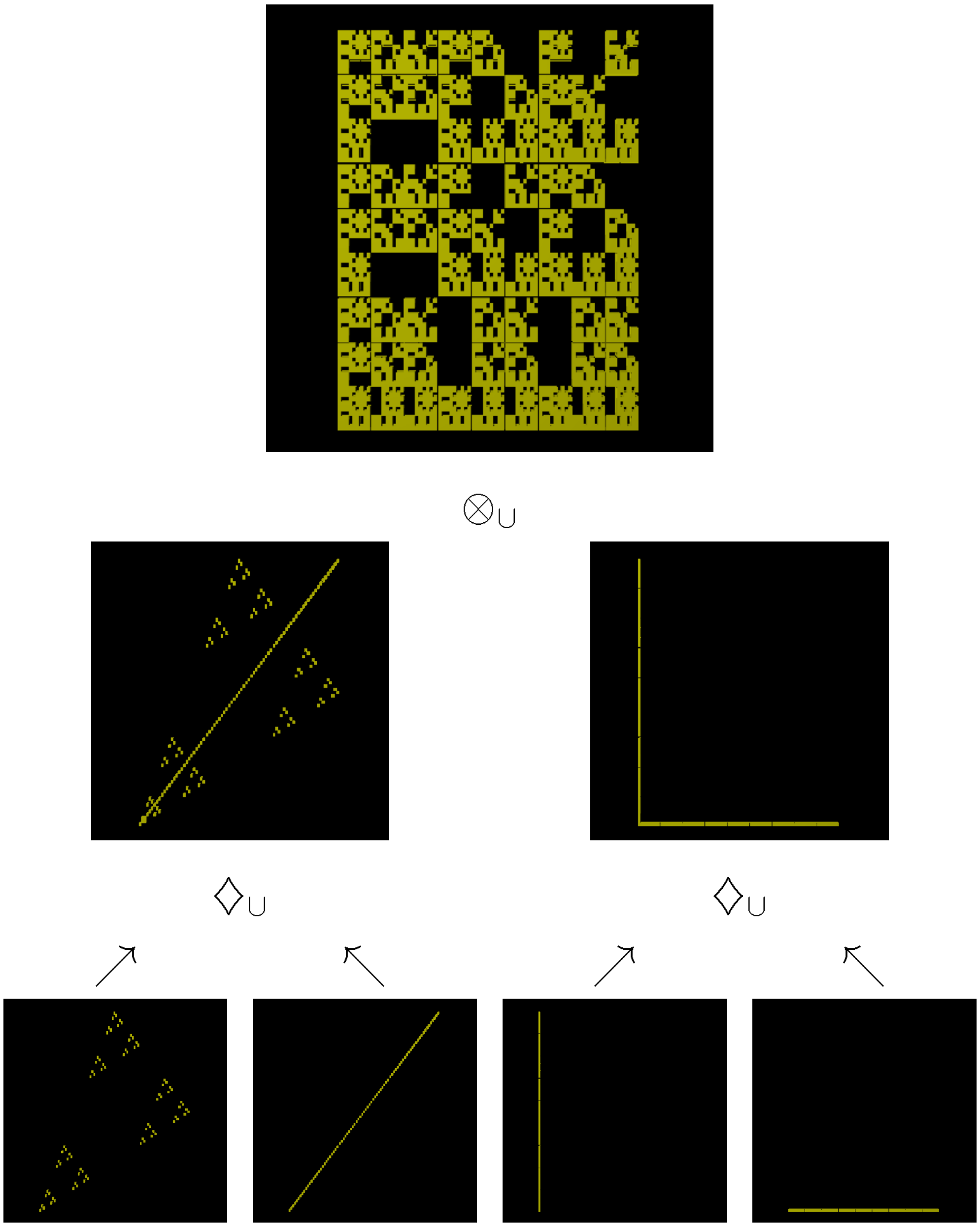


FIG. 6.3 - *Arbre de construction.*

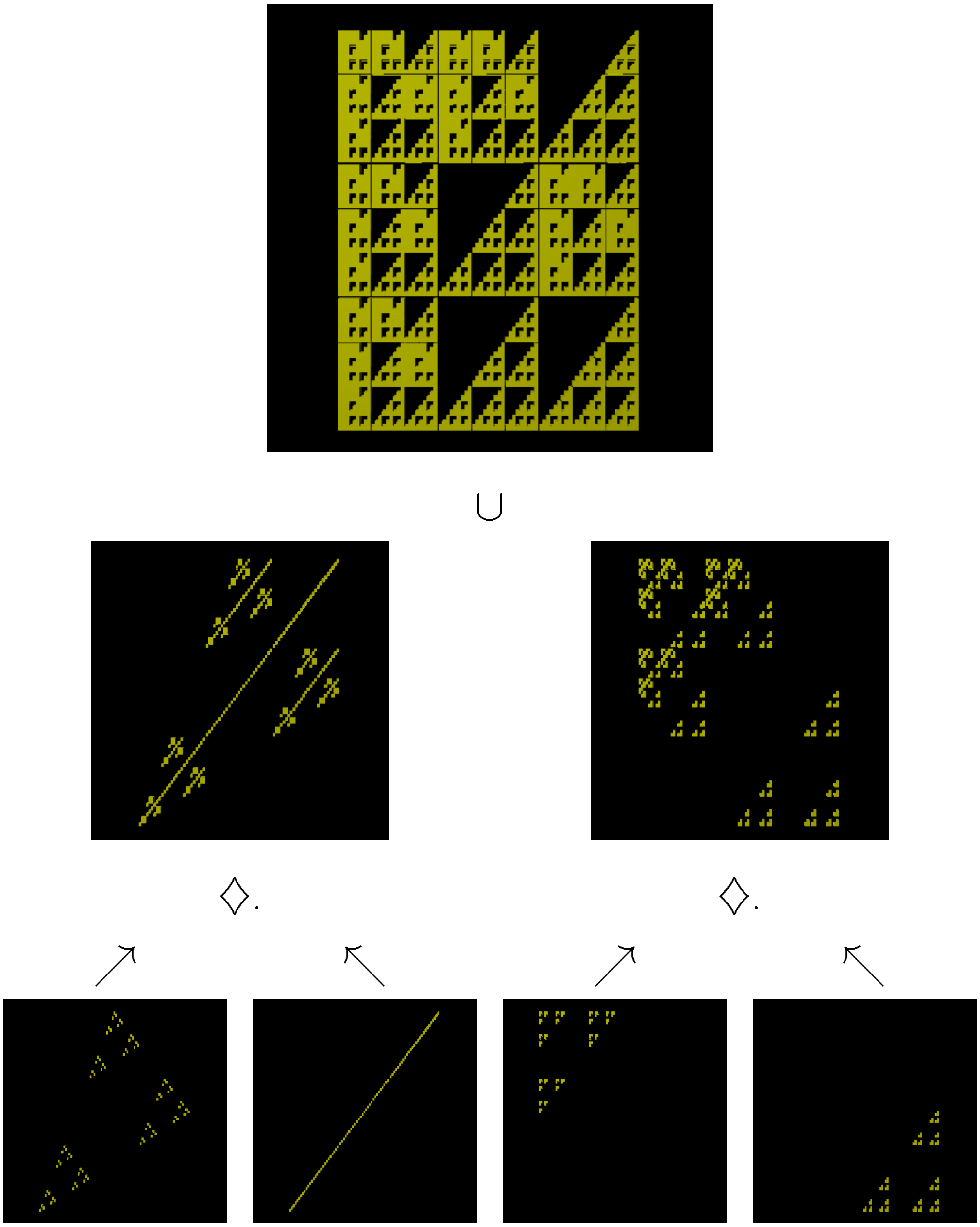


FIG. 6.4 - *Arbre de construction.*

6.1.2 Comparaison avec le CSG

Le modèle CSG se compose de trois opérations binaires : union, intersection, différence, d'opérations unaires : les transformations affines, et de primitives.

Pour ce qui est des opérations, notre modèle nous permet de retrouver l'union ainsi que les transformations. L'union correspond à l'union des langages, et les transformations correspondent à la concaténation d'un langage fini et d'un langage infini. En effet, en posant

$$L \text{ fini} \Rightarrow \mathcal{A}(L) = \emptyset,$$

on a, si L est fini et L' infini,

$$h(L) \circ \mathcal{A}(L') = \mathcal{A}(L.L').$$

En particulier, pour appliquer une transformation $h(a)$ à un attracteur $\mathcal{A}(L')$, il suffit de calculer $\mathcal{A}(a.L')$.

En revanche, on ne retrouve pas l'intersection et *a fortiori* la différence. On obtient à la place de nouvelles opérations : la concaténation, le mélange et les opérations directes sur les matrices. Le mélange pourra, par exemple, permettre de faire des extrusions linéaires.

Pour ce qui est des primitives, nous pouvons choisir soit des primitives langages, soit des primitives matrices. Par exemple, si l'on considère $\Sigma = \{1, 2, \ldots, N\}$ un alphabet. Alors l'ensemble des langages :

$$\{\{\epsilon\}, \{1\}, \{2\}, \ldots \{N\}\}$$

va nous permettre de définir les opérateurs de HUTCHINSON. L'ensemble des langages :

$$\{\{1\}^*, \{2\}^*, \ldots \{N\}^*\}$$

permet de construire les IFS grâce au mélange. En ajoutant les autres opérations on obtient une famille de LRIFS. Le choix des primitives n'a pas fait l'objet d'une étude théorique plus poussée. En pratique, nous avons utilisé des automates à un ou deux états et des matrices de taille deux, sachant que la taille augmente assez rapidement dès que l'on utilise le mélange ou le produit tensoriel.

6.2 Problème du choix des langages

Les opérations que nous avons définies ne sont pas des morphismes. Or, nous aimerions que la combinaison de deux attracteurs donnés, via leur représentation (IFS ou LRIFS), donne un résultat unique. Ce n'est pas le cas, en général, pour deux raisons. Tout d'abord, deux IFS peuvent avoir le même attracteur, et, d'autre part, deux langages sur le même alphabet (associés au même IFS) peuvent avoir le même attracteur.

Exemple : Si l'on considère le segment $[0, 1]$, on peut le diviser en deux ou en trois segments. Il est donc l'attracteur des deux IFS :

$$\mathcal{T} = \{s_{21}, s_{22}\},$$

avec

$$\begin{aligned} s_{21} &= H(0.5), \\ s_{22} &= T(0.5, 0, 0) \circ H(0.5); \end{aligned}$$

et

$$\mathcal{T}' = \{s_{31}, s_{32}, s_{33}\},$$

avec

$$\begin{aligned} s_{31} &= H(\tfrac{1}{3}), \\ s_{32} &= T(\tfrac{1}{3}, 0, 0) \circ H(\tfrac{1}{3}), \\ s_{33} &= T(\tfrac{2}{3}, 0, 0) \circ H(\tfrac{1}{3}). \end{aligned}$$

Si l'on concatène ce segment avec un segment perpendiculaire, on obtiendra deux résultats différents suivant la représentation choisie.

On peut formuler ce cas en termes de langages en prenant l'IFS

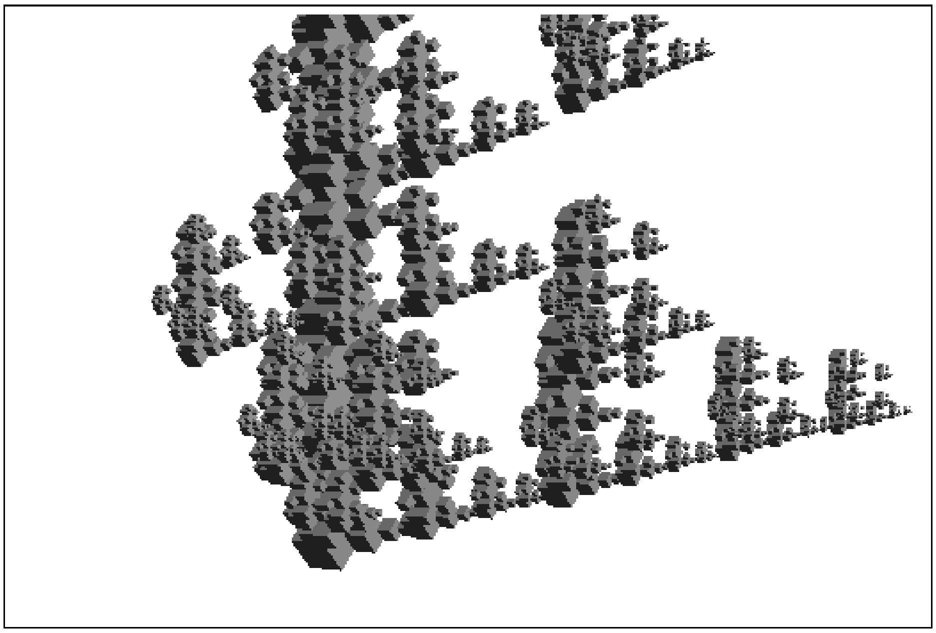
$$\mathcal{T} = \{s_{21}, s_{22}, s_{31}, s_{32}, s_{33}\}$$

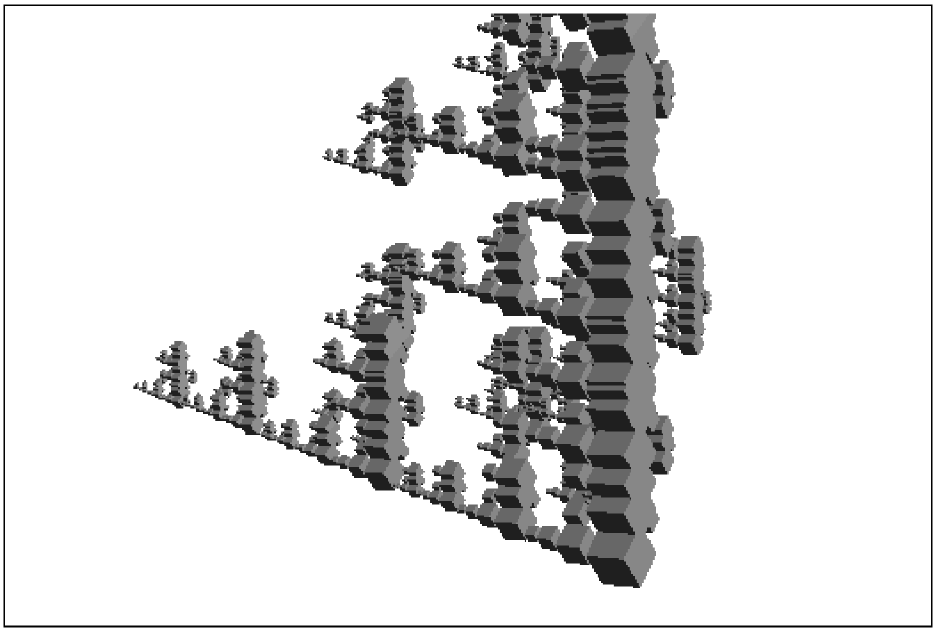
et les langages $L = \{21, 22\}^*$ et $L' = \{31, 32, 33\}^*$. Ces deux langages donnent le même attracteur.

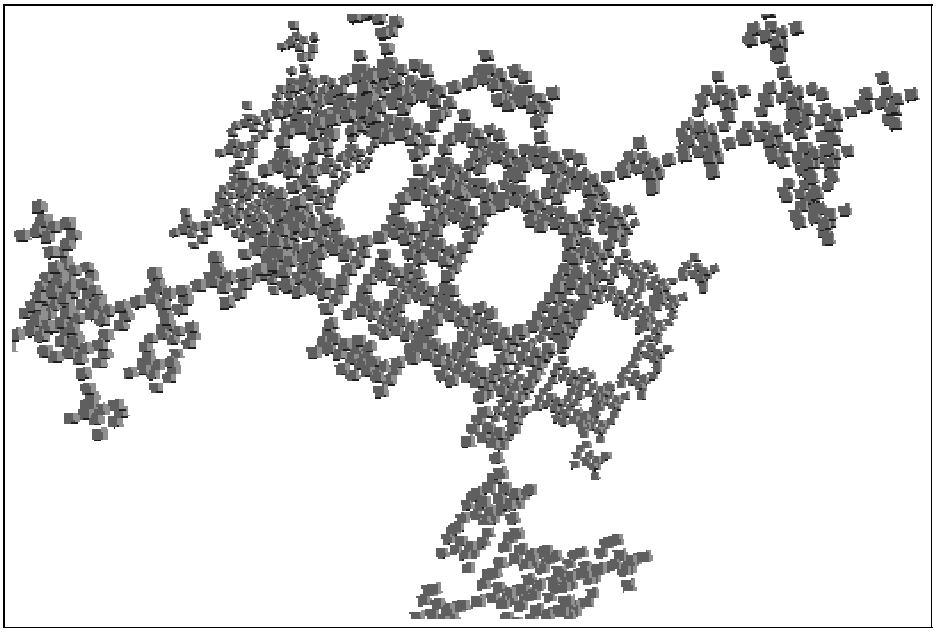
Ce problème est lié à la définition même des IFS. Nous ne pouvons donc espérer y répondre dans le cas général. En revanche, on peut se restreindre à certains type de transformations, afin d'obtenir une unicité de la représentation de l'attracteur.

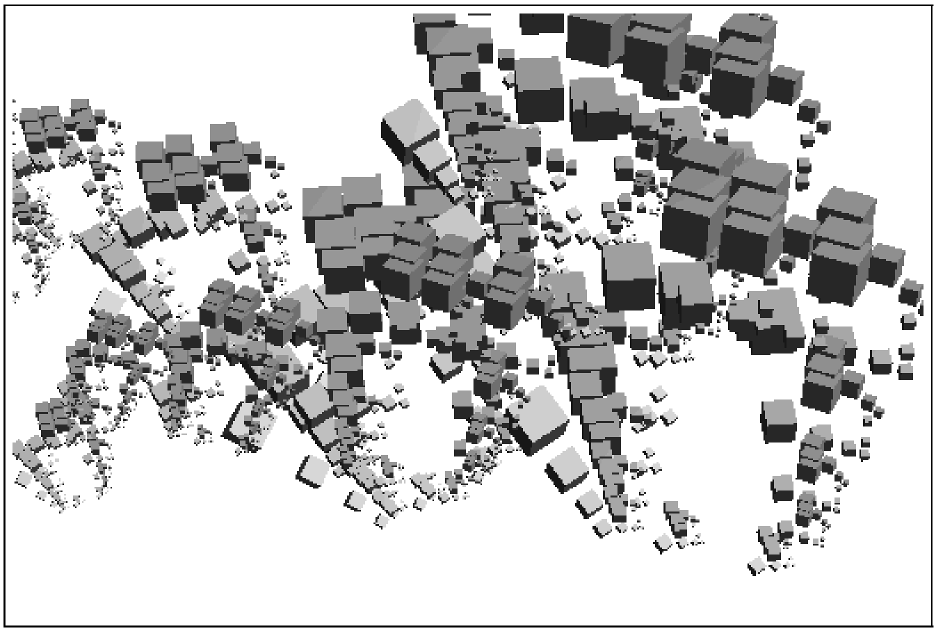
6.3 Quelques images

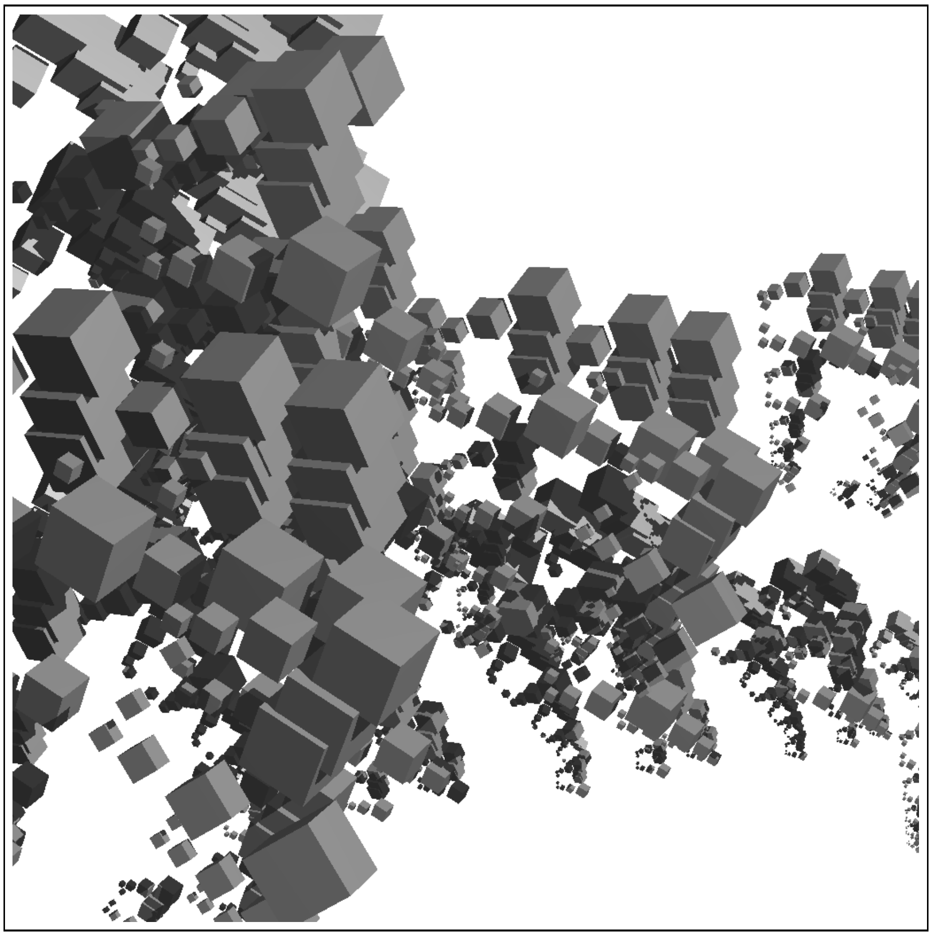
Nous donnons dans cette partie des images composées en collaboration avec Martine Rondet-Mignotte, dans le cadre d'un projet art et science. Ces images utilisent la concaténation et le mélange.











Conclusion générale

Nous avons proposé dans cette thèse deux modèles d'extension d'IFS qui nous permettent de définir des objets fractals de manière constructive. Nous disposons maintenant de toute une gamme d'opérations. Elles ont été ordonnées en terme d'inclusion, afin d'améliorer l'intuition du choix de l'opération correspondant au résultat souhaité. Nous avons vu qu'avec trois de ces opérations nous obtenons un système de géométrie constructive, avec des primitives bien définies.

Ce travail est une étude faisant partie d'un projet de *modeleur fractal*. Un prototype a été programmé par Chems Eddine Zair, Eric Tosan et moi-même. Deux plasticiennes ont travaillé avec nous pour tester ce modeleur. Leur collaboration nous a permis de faire des tests avec des personnes n'ayant pas une formation scientifique. Les résultats obtenus semblent montrer que la démarche constructive se prête bien à une démarche de création.

D'un point de vue théorique, nous avons défini deux modèles permettant de généraliser les modèles liés aux IFS. Le modèle LRIFS va permettre d'atteindre tous les compacts d'un espace métrique complet, donc, en particulier, toutes les figures représentables sur un écran. Le modèle matriciel va nous permettre de sélectionner simplement plusieurs types d'attracteurs.

Plusieurs questions restent ouvertes; en particulier, celle de l'unicité de la représentation. Ce problème est inhérent aux IFS et reste donc ouvert pour les LRIFS. Il serait intéressant de définir des classes de transformations et de langages assurant cette unicité. D'autre part, nous avons montré l'équivalence entre notre modèle matriciel et les modèles antérieurs. Il serait utile de définir les algorithmes permettant de passer d'un modèle à un autre. D'un point de vue théorique, nous devrons maintenant nous attacher à qualifier le résultat des opérations de manière plus précise. Par exemple, en étudiant la dimension fractale ou la connexité des attracteurs résultats par rapport aux attracteurs de départ.

Pour ce qui est du modeleur, un gros travail reste à faire en ce qui concerne l'interface utilisateur. Par exemple, il faut définir la représentation (automate, matrice, système d'équations, ...) qui se prête le mieux à une saisie graphique. Enfin, nous aimerions pouvoir intégrer dans une même scène des objets CSG classiques et des objets fractals, tout en bénéficiant de leurs opérations respectives.

Annexe A

Éléments de Théorie des Langages

Nous donnons ici quelques notions de Théorie des Langages utilisées dans cette thèse. Les notions de langages de mots finis sont tirées de [Har78], et celles de langages de mots infinis de [BN80] et [MS94].

A.1 Langages formels

Les langages formels ont été introduits par Chomsky, en 1956, pour étudier la structure mathématique des langages naturels. Ils sont maintenant largement utilisés dans de nombreux domaines de l'informatique.

A.1.1 Langages de mots finis

La Théorie des Langages est basée sur la notion de lettres et de mots. Nous rappelons donc les différentes définitions liées aux mots :

- Un alphabet Σ est un ensemble fini d'objets, appelés lettres ou caractères.
- Un mot sur un alphabet Σ est une séquence finie de lettres de Σ. On notera Σ^* l'ensemble des mots sur Σ. Le mot vide est noté ϵ.
- La longueur d'un mot non vide w est le nombre de lettres de ce mot; on la note $|w|$.

- La concaténation d'un mot $u = u_1u_2 \dots u_n$ et d'un mot $v = v_1v_2 \dots v_m$ est le mot $u.v = u_1u_2 \dots u_nv_1v_2 \dots v_m$.
- Le mot u est un préfixe du mot w s'il existe un mot v tel que $w = u.v$. On notera
$$\mathrm{Pref}(w) = \{u | \exists v, u.v = w\}.$$

À partir des mots, on peut définir les langages qui seront des ensembles de mots, puis on peut généraliser la notion de concaténation :

- Un langage formel L sur Σ est une partie de Σ^*.
- La concaténation de deux langages est égale à l'ensemble des concaténations de mots des langages :
$$L.L' = \{w \in \Sigma^* / w = u.v \quad u \in L \quad v \in L'\}.$$
On notera $L^i = L.L^{i-1}$.
- On notera L^* le plus petit langage contenant ϵ et L, clos par concaténation,
$$L^* = \bigcup_{i \in \mathbb{N}} L^i,$$
c'est-à-dire l'ensemble des concaténations finies de mots de L.
- L'ensemble des préfixes d'un langage L est égal à l'ensemble des mots qui sont préfixes d'un mot de L. On le notera $\mathrm{Pref}(L)$.

A.1.2 Langages de mots infinis

En Théorie des Langages classique, les langages sont des ensemble infinis de mots finis. On peut introduire des langages infinis de mots infinis. Cette approche permet d'introduire des notions particulières telles que l'entropie d'un langage définie dans [MS94] appelée aussi adhérence dans [BN80].

- Un mot infini sur Σ est une séquence infinie de lettres de Σ.
- L'ensemble des mots infinis sur Σ est noté Σ^ω.
- La concaténation se généralise aux mots infinis. Soient $u \in \Sigma^*$, $\sigma \in \Sigma^\omega$, et $\alpha \in \Sigma^* \cup \Sigma^\omega$; alors,
$$u.\sigma = u_1u_2 \dots u_n\sigma_1\sigma_2 \dots$$

$$\sigma.\alpha = \sigma.$$

Cette définition nous permet de généraliser la notion de préfixes. Pour tout $\alpha, \beta \in \Sigma^* \cup \Sigma^\omega$, α est un préfixe de β si et seulement si il existe $\gamma \in \Sigma^* \cup \Sigma^\omega$ tel que

$$\alpha.\gamma = \beta.$$

- Un langage infini est une partie de Σ^ω.
- Soit $L \subseteq \Sigma^*$, on définit

  $$L^\omega = \{\sigma \in \Sigma^\omega / \sigma = \sigma_1\sigma_2 \ldots \quad \sigma_i \in L - \{\epsilon\}\},$$

  c'est-à-dire l'ensemble des concaténations infinies de mots de L.
- Un mot infini σ, tel que $\mathrm{Pref}(\sigma) \subseteq \mathrm{Pref}(L)$, est appelé une adhérence de L. L'ensemble des adhérences de L est noté $\mathrm{ls}(L)$. On a donc

  $$\begin{aligned} \mathrm{ls}(L) &= \{\sigma \in \Sigma^\omega / \mathrm{Pref}(\sigma) \subseteq \mathrm{Pref}(L) \\ &= \{\sigma \in \Sigma^\omega / \sigma_1\sigma_2 \ldots \quad \sigma_n \in L_n\}, \end{aligned}$$

  en notant L_n l'ensemble des préfixes de longueur n de L.

On peut définir des ultramétriques sur les langages, de telle sorte que $\Sigma^* \cup \Sigma^\omega$ soit un espace métrique complet. Ces ultramétriques nous permettront de faire des calculs de limites sur des suites de langages.

Définition A.1.1 ([Edg90]) *Soient $\alpha, \beta \in \Sigma^* \cup \Sigma^\omega$. Si l'on note γ le plus long préfixe commun entre α et β, on définit alors :*

$$d(\alpha, \beta) \quad = \quad \rho(\gamma),$$

avec ρ une fonction à valeurs dans $[0, 1[$, telle que

$$\begin{cases} \rho(\gamma) = 0 \text{ si } \gamma \in \Sigma^\omega, \\ \rho(\gamma_1) \leq \rho(\gamma_2) \text{ si } \gamma_1 \in \mathrm{Pref}(\gamma_2). \end{cases}$$

On a alors la proposition suivante [Edg90] :

Proposition A.1.1 (Σ^ω, d) *est un espace métrique complet. De plus, les espaces métriques, construits sur les différentes fonctions ρ, sont tous homéomorphes.*

En particulier, la plupart des auteurs ont utilisé la fonction :

$$\rho(\gamma) = r^{|\gamma|},$$

avec $r \in \mathbb{R}$ et $r < 1$.

On peut induire une distance de HAUSDORFF à partir de cette ultramétrique :

Définition A.1.2 *Soient L, L' deux langages compacts, c'est-à-dire $L, L' \in \mathcal{H}(\Sigma^\omega)$ ou $L, L' \in \mathcal{H}(\Sigma^* \cup \Sigma^\omega)$. On définit*

$$d_H(L, L') = \max\{\max_{\alpha \in L} \min_{\beta \in L'} d(\alpha, \beta), \max_{\beta \in L'} \min_{\alpha \in L} d(\alpha, \beta)\}.$$

A.2 Langages rationnels

Les langages rationnels forment une classe de langages facilement implémentables. En effet, il existe des procédés simples pour générer ces langages. Trois représentations ont été étudiées : les expressions rationnelles, les grammaires, et les automates. La classe des langages rationnels correspond à la classe des langages décrits par une expression rationnelle, générés par une grammaire linéaire à droite, ou reconnus par un automate fini. Ces trois modèles étant équivalents, nous utiliserons celui qui nous permettra de faire les démonstrations les plus simples.

A.2.1 Expressions rationnelles

Définition A.2.1 *Les expressions rationnelles sont définies par récurrence :*

1. *L'ensemble vide est une expression rationnelle.*
2. *Si $a \in \Sigma$ alors $\{a\}$ est une expression rationnelle.*
3. *Si e_1 et e_2 sont deux expressions rationnelles, alors $e_1 \cup e_2$, $e_1.e_2$ et e_1^* sont des expressions rationnelles.*

Exemple : $R_{arbre} = \{3, 4\}^*\{1, 2\}^*$ est une expression rationnelle qui décrit le langage L_{arbre}, dont les mots sont tels qu'aucun 1 ou 2 n'est suivi par un 3 ou un 4.

A.2.2 Grammaires

Définition A.2.2 *Une grammaire formelle est un quadruplet* $\mathcal{G} = (N, \Sigma, P, S)$, *avec :*

- N *un ensemble fini de variables;*
- Σ *un alphabet;*
- P *un ensemble de productions ou règles de grammaire,*
$$P = \{(u, v) \in V^* \times V^*\}$$
avec $V = N \cup \Sigma$. *On notera les productions* $u \to v$ *ou* $u \to v|v'|\ldots$ *si* u *est la partie gauche de plusieurs productions;*
- $S \in N$ *l'axiome de* $\mathcal{G}$.

Le langage engendré par $\mathcal{G}$ est
$$L(\mathcal{G}) = \{u \in \Sigma^* / S \Rightarrow^* u\},$$
avec $u \Rightarrow v$ si
$$u = wu'w',$$
$$v = wv'w',$$
$$u' \to v',$$
et $u \Rightarrow^* v$ si $\exists (w_i)_{0 \le i \le n} \in V^*$, telle que
$$w_0 = u,$$
$$w_n = v,$$
$$\forall i \ w_i \Rightarrow w_{i+1} \in P.$$

Exemple : la grammaire d'état fini $\mathcal{G}_{arbre} = (N, \Sigma, P, S)$, avec :
$$N = \{S, X\},$$
$$\Sigma = \{1, 2, 3, 4\},$$
$$P = \{S \to 3S|4S|1X|2X|\epsilon, X \to 1X|2X|\epsilon\},$$
engendre le langage
$$L(\mathcal{G}_{arbre}) = L_{arbre} = \{3, 4\}^*\{1, 2\}^*.$$

A.2.3 Automates finis

Définition A.2.3 *Un automate fini est un quintuplet* $\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$ *avec :*

- Q *un ensemble fini appelé* l'ensemble des états*;*
- Σ *un alphabet;*
- δ *la fonction de transition de l'automate. On dit que l'automate est* déterministe *si* $\delta : Q \times \Sigma \to Q$*, et qu'il est* non déterministe *si* $\delta : Q \times \Sigma \to \mathcal{P}(Q)$*;*
- $Q_I \subseteq Q$ *l'ensemble des états initiaux;*
- $Q_F \subseteq Q$ *l'ensemble des états terminaux.*

Le langage reconnu par $\mathcal{M}$ est dans le cas déterministe

$$L(\mathcal{M}) = \{u \in \Sigma^* / \delta(q, u) = q' \text{ avec } q \in Q_I \text{ et } q' \in Q_F\},$$

avec $\delta(q, a.u) = \delta(\delta(q, a), u)$ si $a \in \Sigma$ et $u \in \Sigma^*$.

Dans le cas non déterministe,

$$L(\mathcal{M}) = \{u \in \Sigma^* / \exists q \in Q_I, \delta(q, u) \cap Q_F \neq \emptyset\}$$

On autorise dans ce cas les ϵ-transitions : une transition étipuettée ϵ qui ne reconnaite aucune lettre mais permet de passer à un autre état.

Exemple : l'automate $\mathcal{M}_{arbre} = (Q, \Sigma, \delta, Q_I, Q_F)$ avec :

$$Q = Q_F = \{q_0, q_1\},$$

$$Q_I = \{q_0\},$$

$$\Sigma = \{1, 2, 3, 4\},$$

$$\delta(q_0, 3) = q_0,$$

$$\delta(q_0, 4) = q_0,$$

$$\delta(q_0, 1) = q_1,$$

$$\delta(q_1, 1) = q_1,$$

$$\delta(q_0, 2) = q_1,$$

$$\delta(q_1, 2) = q_1,$$

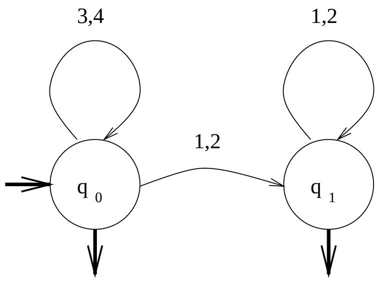


FIG. A.1 - *Exemple de graphe d'automate.*

reconnaît le langage $L(\mathcal{M}_{arbre}) = L_{arbre} = \{3,4\}^*\{1,2\}^* = L(\mathcal{G}_{arbre})$.

Si l'on considère l'automate fini $\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$, on peut lui associer le graphe $G(\mathcal{M}) = (Q, E)$ orienté et étiqueté tel que :

- l'ensemble des sommets du graphe est l'ensemble Q des états de l'automate;
- l'ensemble E des arêtes du graphe représente δ la fonction de transition de l'automate, c'est-à-dire

$$(q, a, q') \in E \Leftrightarrow q' \in \delta(q, a).$$

On notera avec une flèche entrante les états initiaux et avec une flèche sortante les états finaux.

Exemple : Nous avons dessiné Figure A.1 le graphe correspondant à l'automate $\mathcal{M}_{arbre}$.

Proposition A.2.1 *Si $G(\mathcal{M})$ est le graphe associé à l'automate $\mathcal{M}$, alors l'ensemble des mots de longueur n générés par $\mathcal{M}$ est donné par l'ensemble des chemins de longueur n partant d'un état initial et aboutissant à un état final.*

Démonstration : Un mot $u = u_1u_2 \ldots u_m$ est accepté par $\mathcal{M}$ si

$$\exists q_0, q_1, q_2, \ldots q_m \in Q,$$

tels que

$$\forall i \; \delta(q_{i-1}, u_i) = q_i,$$

$$q_0 \in Q_I \text{ et } q_m \in Q_F.$$

Par conséquent, u est accepté par $\mathcal{M}$ si

$$\exists q_0, q_1, q_2, \ldots q_m \in Q,$$

tels que

$$\forall i \ (q_{i-1}, u_i, q_i) \in E,$$

$$q_0 \in Q_I \text{ et } q_m \in Q_F,$$

c'est-à-dire s'il existe un chemin dans $G(\mathcal{M})$, partant d'un état initial, se terminant dans un état final, et étiqueté par les lettres de u. $\square$

A.2.4 Langages infinis rationnels

Définition A.2.4 *Un mot infini est accepté par un automate fini si la fonction de transition passe une infinité de fois par un état final.*

Un langage infini, dont les mots sont reconnus par un automate fini selon la définition précédente, est un *langage infini rationnel.*

Annexe B

Notations

B.1 IFS

$(\mathcal{X}, d)$ un espace métrique complet.
p, p_i des points de $\mathcal{X}$.
T, T_i des transformations.
s_i la constante de contraction associée à T_i.
c le point fixe ou centre d'une transformation.
$\mathcal{H}(\mathcal{X})$ l'ensemble des compacts non vides de $(\mathcal{X}, d)$.
d_H la distance de HAUSDORFF.
K, K' des compacts.
H l'opérateur de HUTCHINSON.
$\mathcal{T} = \{T_1, \ldots, T_N\}$ un IFS.
$\mathcal{A}(\mathcal{T})$ l'attracteur de l'IFS $\mathcal{T}$.

B.2 Langages

Σ un alphabet.
a, b des lettres.
Σ^* L'ensemble des mots finis sur Σ.
ϵ le mot vide.
u, v, w des mots finis.
$u_1, u_2, \ldots u_n$ les lettres du mot u.

L, L' des langages.
$\mathrm{Pref}(L)$ l'ensemble des préfixes d'un langage L.
L_n l'ensemble des préfixes de longueur n de L.
Σ^ω l'ensemble des mots infinis sur Σ.
σ un mot infini.
α, β, γ des mots finis ou infinis.
$\mathrm{ls}(L)$ l'adérence du langage L.
$\mathcal{G} = (N, \Sigma, P, S)$ une grammaire formelle.
$L(\mathcal{G})$ le langage généré par $\mathcal{G}$.
$\mathcal{M} = (Q, \Sigma, \delta, Q_I, Q_F)$ un automate fini.
$L(\mathcal{M})$ le langage reconnu par $\mathcal{M}$.
$G = (V, E)$ un graphe.
$G(\mathcal{M})$ le graphe de l'automate $\mathcal{M}$.

B.3 IFS et langages

h la fonction d'étiquetage.
ϕ la fonction d'adressage.
$\mathcal{I} = (\mathcal{T}, \Sigma, h, L)$ un LRIFS.
$\mathcal{A}(L)$ l'attracteur associé au LRIFS $\mathcal{I}$.
$\mathcal{A}_L$ attracteur selon PRUSINKIEWICZ.

B.4 IFS et matrices

$M_R(\mathcal{H}(\mathcal{X}))$: l'ensemble des matrices de compacts de $\mathcal{X}$.
$\tilde{K}$: une matrice de compacts.
$M_R(\mathcal{P}_+(\mathcal{S}))$, l'ensemble des matrices de HUTCHINSON.
$\tilde{H}$ une matrice de HUTCHINSON.
d_∞ : la distance du max déduite de la distance de HAUSDORFF.
$\Delta(\mathcal{M})$ matrice associée à un automate $\mathcal{M}$.
$\tilde{\mathcal{A}}(\tilde{H})$: la matrice d'attracteurs associée à $\tilde{H}$.
$\mathcal{A}(\tilde{H})$: l'attracteur "scalaire" associé à $\tilde{H}$.
$\mathcal{A}(\mathcal{M})$ attracteur associé à un automate $\mathcal{M}$.
I et F le vecteur initial et le vecteur final.
X, Y des vecteurs de $\mathcal{H}(\mathcal{X})^n$.

Table des figures

Bibliographie

[Ban89] C. Bandt. Topological Markov Chaïne and Mixed Self-Similar Sets. *Math. Nachr.*, 142:107–123, 1989.

[Bar88] M.F. Barnsley. *Fractals Everywhere.* Academic press, INC, 1988.

[BHH89] M.F. Barnsley, J.H. Helton, and D.P Hardin. Recurrent iterated function systems and the global construction of fractals. *Constructive Approximation*, 5:3–31, 1989.

[BJM+88] M.F. Barnsley, A. Jacquin, F. Malassenet, L. Reuter, and A.D. Sloan. Harnessing chaos for image synthesis. *Computer Graphics*, 22(4):131–140, August 1988.

[BM89] J. Berstel and M. Morcrette. Compact representation of patterns by finite automata. In *Proceedings of Pixim*, pages 387–395, 1989.

[BN80] L. Boasson and M. Nivat. Adherances of languages. *Journal of computer and system sciences*, 20:285–309, 1980.

[BNA89] J. Berstel and A. Nait Abdallah. Tétrarbres engendrés par des automates finis. *Langages et algorithmes du graphique, Bigre n° 61-62*, Avril 1989.

[CD92] K. Culik II and S. Dube. L-systems and mutually recusive function systems. To appear, September 1992.

[CD93a] K. Culik II and S. Dube. Balancing order and chaos in image generation. *Computer & Graphics*, 17(4):465–486, 1993.

[CD93b] K. Culik II and S. Dube. Rationnal and affine expressions for image synthesis. *Discrete Appl. Math.*, 41:85–120, 1993.

[CG93] J.D. Corbit and D.J. Garbary. Computer simulation of the morphology and development of several species of seaweed using Lindenmayer systems. *Computer & Graphics*, 17(1):85–88, 1993.

[Dek80] F.M Dekking. Variations on Peano. *Nieuw Archief Voor Wiskunde*, XXVIII:275–281, 1980.

[Dek82] F.M. Dekking. Recurrent sets. *Advances in Mathematics*, 44(1):78–104, April 1982.

[Dek87] F.M. Dekking. Construction de fractals et problèmes de dimension. In *Dimensions Non Entière et Applications*. Masson, 1987.

[DLVM95] K. Daoudi, J. Lévy Véhel, and Y. Meyer. Construction of continuous functions with prescribed local regularity. *To appear in Constructive Approximation*, 1995. Rapport de recherche INRIA *N°* 2763.

[Edg90] G.A. Edgar. *Measure, Topology, and Fractal Geometry*. Springer-Verlag, 1990.

[Gen92] C. Gentil. *Les Fractales en Synthèse d'Images : le Modèle IFS*. PhD thesis, Université Claude Bernard LYON 1, France, 1992.

[GM86] M. Gondran and M. Minoux. *Graphes et algorithmes*. Editions Eyrolles, 1986.

[GV90] C. Gentil and D. Vandorpe. Modélisation de fractales par IFS : Visualisation et approximation. *Journées GROPLAN Saint-Etienne*, 1990.

[Har78] M.A. Harrison. *Introduction to Formal Language Theory*. Addison-Wesley Publishing Company, 1978.

[Har91] J. Hart. unNatural Phenomena. SIGGRAPH Video Review 71, 1991.

[Har92] J. Hart. The object instancing paradigm for linear fractal modeling. In *Proceedings of Graphics Interface'92*, May 1992.

[HPS91] D. Hepting, P. Prusinkiewicz, and D. Saupe. Rendering methods for iterated function systems. In H.-O. Peitgen, J.M. Henriques, and L.F. Penedo, editors, *Fractals in the Fundamental and Applied Sciences*, pages 183–224. IFIP, Elsevier Science Publishers B.V, 1991.

[Hut81] J. Hutchinson. Fractals and self-similarity. *Indiana University Journal of Mathematics*, 30:713–747, 1981.

[Man83] B.B. Mandelbrot. *The Fractal Geometry of Nature*. W.H. Freeman and Co, New York, 1983.

[Mor96] M. Morcrette. Sur l'équivalence de descriptions de figures itérées. À pararaître dans Theoritical Computer Science, 1996.

[MS94] W. Merzenich and L. Staiger. Fractals,dimension, and formal languages. *Theoritical Informatics and Application*, 28(3-4):361–386, 1994.

[MW88] R.D. Mauldin and L.C. Williams. Hausdorff dimension in graph directed constructions. *Trans. Amer. Math. Soc.*, 309:811–829, 1988.

[PH90] P. Prusinkiewicz and J. Hanan. Visualization of botanical structures and processes using parametric l-systems. *Scientific Visualisation and Graphic Simulation*, pages 183–201, 1990.

[PH91] P. Prusinkiewicz and M.S. Hammel. Automata, languages and iterated function systems. In *lecture notes for the SIGGRAPH'91 course : "Fractal modeling in 3D computer graphics and imagery"*, 1991.

[PH92] P. Prusinkiewicz and M.S. Hammel. Escape-time visualization method for language-restricted iterated function systems. In *Proceedings of Graphics Interface'92*, May 1992.

[PJS92] H.O. Peitgen, H. Jűrgens, and D. Saupe. Encoding images by simple transformations. In *Fractals For The Classroom, New York*, 1992.

[PLH88] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. Developmental models of herbaceous plants for computer imagery purposes. *Computer Graphics*, 22(4):141–150, August 1988.

[Pru86] P. Prusinkiewicz. Graphical applications of L-systems. In *Proceedings of Graphics Interface'86 - Vision Interface '86*, pages 243–247, May 1986.

[PS88] P. Prusinkiewicz and G. Sandness. Koch curves as attractors and repellers. *IEEE Computer Graphics & Applications*, pages 26–40, November 1988.

[Req80] A.A.G. Requicha. representations for rigid solids : theory, methods and systems. *ACM computing surveys*, 12(4), December 1980.

[Rot82] Scott D. Roth. Ray casting for modelling solids. *Computer graphics and image processing*, 18:109–144, 1982.

[SS89] J. Shallit and J. Stolfi. Two methods for generating fractals. *Computer & Graphics*, 13(2):185–191, 1989.

[Tho93] J. Thollot. Langages formels et fractales. Master's thesis, Université Claude Bernard LYON 1, France, Octobre 1993. Rapport de DEA "Informatique fondamentale".

[TT93a] J. Thollot and E. Tosan. Construction of fractales using formal languages and matrices of attractors. In Harold P. Santos, editor, *Proceedings of Compugraphics'93*, pages 74–81, Technical University of Lisbon, december 1993.

[TT93b] J. Thollot and E. Tosan. Langages formels et fractales : les LRIFS. In LaBRI, editor, *Premières Journées AFIG-GROPLAN 93*, pages 9–18, Bordeaux, december 1993.

[TT95a] J. Thollot and E. Tosan. Automata and constructive fractal geometry. Research report RR02.95, LIGIM, Université Claude Bernard, Lyon, France, june 1995.

[TT95b] J. Thollot and E. Tosan. Constructive fractal geometry : constructive approach to fractal modeling using languages operations. In *Proceedings of Graphics Interface'95, Quebec, Canada*, pages 196–203, may 1995.

[Vrs91] E.R. Vrscay. Iterated function systems: Theory, applications and the inverse problem. *Fractal geometry and analysis*, pages 405–468, 1991.

[ZT94] C. Zair and E. Tosan. Définition de courbes ou surfaces lisses et fractales à l'aide d'IFS. In *Journées AFIG de Toulouse*, 1994.

[ZT96] C. Zair and E. Tosan. Fractal modeling using free forms techniques. In *Proceedings of Eurographics*, 1996.