Université de Marne-la-Vallée

Automates et mots infinis

Mémoire d'habilitation à diriger des recherches

Présenté par

Olivier Carton

le 14 décembre 2001
devant le jury :

Jean Berstel	Université de Marne-la-Vallée
Christian Choffrut	Université Paris 7
Dominique Perrin	Université de Marne-la-Vallée
Jean-Éric Pin	CNRS
Paul Schupp	University of Illinoy
Géraud Sénizergues	Université de Bordeaux 1
Wolfgang Thomas	RWTH Aachen

Table des matières

Remerciements

J'aimerais remercier toutes les personnes qui m'ont permis de mener à bien mes travaux de recherches et d'aboutir à la présentation de cette Habilitation à Diriger des Recherches. Je tiens tout particulièrement à en remercier quelques unes.

J'aimerais tout d'abord remercier Paul Schupp, Géraud Sénizergues et Wolfgang Thomas d'avoir accepté le rôle au combien ingrat de rapporteur même si ce mémoire a le mérite d'être concis.

J'aimerais ensuite remercier Jean Berstel, Christian Choffrut et Jean-Éric Pin de faire partie de mon jury. J'ai eu souvent l'occasion de discuter et de travailler avec eux. C'est avec beaucoup de plaisir que les vois participer à ce jury.

Il est souvent dit que la recherche en mathématiques est un travail solitaire. C'est souvent le cas et les collaborations n'en ont que plus de valeur. J'aimerais remercier chaleureusement tous ceux et toutes celles avec qui j'ai eu le plaisir de travailler. Ces recherches menées en collaboration eurent des fortunes diverses mais ce sont plus les moments passés à chercher et à sécher ensemble qui importent à mes yeux. Pour citer quelques noms, je mentionnerais Marie-Pierre Béal dont le partage du bureau a conduit à commettre quelques articles sur les transducteurs et Véronique Bruyère que les ordres dispersés mettaient dans tous ses états.

Dominique Perrin a su guider mes premiers pas en recherche et a accompagné de ses encouragements constants ceux qui ont suivi. Il a su me communiquer son enthousiasme pour les automates et les mathématiques discrètes en général. Depuis quelques temps déjà, il m'incitait à rédiger ce mémoire. Après quelques résistances de ma part, ses encouragements ne sont pas restés vains.

L'avancement de mes travaux doit beaucoup à deux années passées en délégation au CNRS. Beaucoup d'articles ont été rédigés et certains résultats ont été trouvés pendant cette période. Cette délégation m'a en outre permis d'effectuer un long séjour à Aix-la-Chapelle. Je voudrais remercier Pascal Weil d'abord pour son soutien dans l'obtention de cette délégation et surtout pour les nombreuses discussions que nous avons eu à propos des semigroupes.

J'aimerais remercier les personnes qui m'ont accueilli dans des universités étrangères où j'ai eu l'occasion de séjourner. À ce titre, j'aimerais en particulier remercier Wolfgang Thomas pour deux séjours fructueux passés à Aix-la-Chapelle et Véronique Bruyère pour son accueil à Mons toujours aussi chaleureux.

J'aimerais finalement adresser mes remerciements sincères et émus à toutes les personnes qui m'ont soutenu pendant les moments difficiles.

1 Introduction

Ce document est une présentation d'une partie des travaux que j'ai effectués depuis mon doctorat en 1993. Il ne s'agit pas d'une énumération systématique et exhaustive. Ce n'est pas que ma production soit si importante, mais j'ai préféré présenter des travaux qui me semblent bien représenter les orientations de mes recherches et ceux auxquels je suis particulièrement attaché pour des raisons esthétiques.

Depuis le début de mon doctorat, le thème central de mes recherches a été l'étude des automates sur des objets infinis tels que les mots infinis, les mots bi-infinis ou même les mots transfinis. Cette présentation est articulée autour de trois thématiques qui sont les automates sur les mots infinis, les transducteurs et la dynamique symbolique et finalement les automates sur des ordres linéaires. Ce regroupement est fait en fonction des thématiques auxquelles se rattachent naturellement mes travaux. Par contre, il ne tient pas compte des méthodes employées qui auraient pu donner lieu à un tout autre regroupement. Certains travaux utilisent des approches de nature algorithmique et combinatoire alors que d'autres sont très algébriques.

La théorie des automates a des liens étroits avec nombre de domaines aussi bien en informatique qu'en mathématiques. Il existe en effet des connexions avec l'algorithmique, la théorie de la complexité, la logique, la combinatoire, la théorie des jeux, l'algèbre, la topologie, la théorie descriptive, les systèmes dynamiques et la théorie des nombres. Cette diversité des interactions est bien reflétée par les différents aspects abordés dans mes recherches. Lors de mon doctorat déjà, j'avais étudié la hiérarchie de Wagner qui établit des liens entre des classes topologiques et des automates. Mes travaux en collaboration avec Marie-Pierre Béal sur les transducteurs et la méthode de calcul de l'indice de Rabin développée avec Ramón Maceiras sont de nature algorithmique. Dans l'étude des automates sur les mots transfinis, nous avons utilisé, avec Nicolas Bedon, des outils algébriques. Finalement, les prédicats morphiques abordés en collaboration avec Wolfgang Thomas concernent des questions de logique.

Les interactions entre ces différents domaines deviennent encore plus concrètes lorsque des méthodes issues d'un domaine donné peuvent être appliquées avec succès à des questions provenant d'autres problématiques. Ainsi les méthodes algébriques ont permis de décomposer les langages cycliques, ce qui a débouché sur des résultats concernant la fonction zêta en dynamique symbolique. De même, la définition des prédicats morphiques provient de la combinatoire des mots qui utilise abondamment les points fixes de morphismes. C'est encore en utilisant des méthodes algébriques que nous avons pu résoudre des questions de décidabilité de certaines logiques liées à ces prédicats.

L'objectif principal de ce document est de donner un aperçu de mes travaux de recherche. L'idée directrice a été de présenter ces travaux de manière simple et concise en essayant d'éviter au maximum la technicité. J'en ai profité pour donner les intuitions, chose qu'on a rarement l'occasion de faire dans les articles. Par contre, cette présentation n'a pas la prétention d'être ni un survol du domaine ni une bibliographie. La liste de références bibliographiques est d'ailleurs limitée au nécessaire pour situer mes résultats.

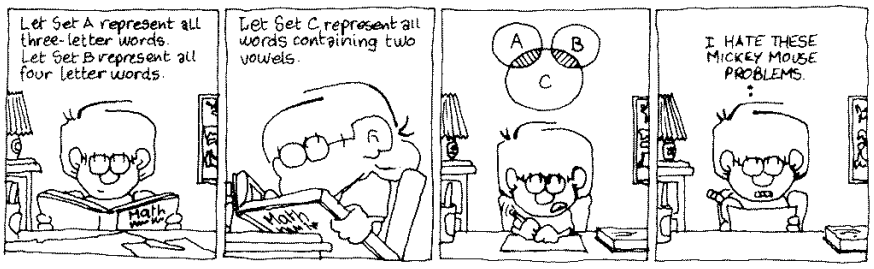


FIG. 1 – Les mammifères à grandes oreilles.

Ce document comporte quatre parties. Au prix de quelques redites, les parties sont indépendantes. La première partie est une introduction aux automates à l'usage des néophytes. Elle est particulièrement destinée aux personnes qui pensent naïvement que les automates sont des petits mammifères sauvages à grandes oreilles. Que cette partie les détrompe en leur montrant qu'on peut les apprivoiser sans difficulté.

La seconde partie est consacrée à des travaux concernant les automates sur les mots infinis. Elle regroupe les résultats de trois collaborations sur des thèmes assez différents avec Ramón Maceiras, Max Michel et Wolfgang Thomas. Les résultats obtenus avec Ramón Maceiras sont de nature algorithmique. Ils concernent le calcul de l'indice de Rabin d'un ensemble de mots infinis donné par un automate à parité. L'objet des travaux avec Max Michel est l'étude d'une classe particulière d'automates de Büchi appelés non-ambigus. Le résultat principal s'apparente à un théorème de McNaughton pour les automates co-déterministes. Finalement, la collaboration avec Wolfgang Thomas a conduit à des résultats de décidabilité de certaines logiques. Il s'agit en quelque sorte d'un retour aux sources puisque ce sont ces mêmes questions de décidabilité qui avaient conduit Büchi à introduire les automates sur les mot infinis.

La troisième partie regroupe des travaux sur la dynamique symbolique et les transducteurs. Le premier résultat a été obtenu en collaboration avec Marie-Pierre Béal et Christophe Reutenauer. Il s'agit d'une décomposition des langages cycliques en langages de stabilisateurs qui conduit à une nouvelle

preuve de la rationalité de la fonction zêta d'un langage cyclique. Les deux autres résultats ont été obtenus en collaboration avec Marie-Pierre Béal. Le premier concerne la synchronisation de transducteurs sur des mots bi-infinis, et le second la déterminisation de transducteurs sur des mots infinis.

La quatrième et dernière partie est consacrée à des extensions de la notion de mots infinis et d'automates. Les résultats obtenus en collaboration avec Nicolas Bedon concernent les mots transfinis et les automates introduits par Büchi. Notre contribution essentielle est d'avoir développé une théorie algébrique de ces mots. Nous avons obtenu un théorème de variétés qui étend celui d'Eilenberg. Nos résultats redonnent en particulier une autre preuve de la déterminisation des automates. Finalement, les travaux avec Véronique Bruyère ont étendu ces automates sur les mots transfinis à des automates sur des ordres linéaires. Le résultat principal est une extension du théorème de Kleene aux ordres dispersés et dénombrables.

2 Petite introduction aux automates

Qu'est-ce qu'un automate ?
C'est une façon concise de représenter ou de décrire un ensemble de mots. Prenons quelques exemples pour éclaircir notre propos. Supposons pour simplifier que les mots soient uniquement écrits avec les lettres a et b. Pour le moment, un mot sera pour nous une suite quelconque de lettres comme aab sans se préoccuper du sens. Pour commencer en douceur, supposons que l'ensemble X contienne le seul mot aab. On peut décrire X par l'automate :

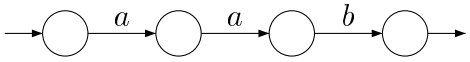


FIG. 2 – Automate pour $X = \{aab\}$.

Le mot aab correspond à la lecture des étiquettes des flèches le long du chemin de la flèche entrante à la flèche sortante. Bien sûr, la description de X par l'automate ne paraît pas très différente de l'écriture $X = \{aab\}$. L'utilité des automates apparaît dès que X contient deux mots. Supposons que X contienne les deux mots aaa et aab. On peut décrire X par l'automate :

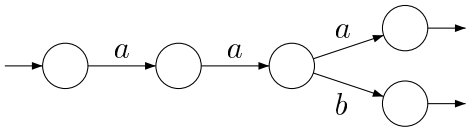


FIG. 3 – Automate pour $X = \{aaa, aab\}$.

Les deux mots aaa et aab correspondent à la lecture des étiquettes des flèches le long des chemins de la flèche entrante à l'une des flèches sortantes. La description de X par automate est déjà plus concise que l'écriture *in extenso* de X. Ceci devient plus marquant si on suppose que X contient tous les mots de longueur 3 pouvant être écrits avec les lettres a et b. On peut décrire X par l'automate :

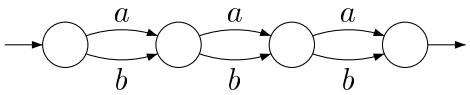


FIG. 4 – Automate pour $X = \{aaa, aab, \ldots, bbb\}$.

Chacun des mots de longueur 3 correspond à la lecture des étiquettes le long d'un des chemins de la flèche entrante à la flèche sortante. L'automate

est beaucoup plus concis qu'une écriture explicite de X. Ce phénomène serait encore plus visible si on avait choisi de représenter les mots de longueur 10. Cette possibilité de représenter un ensemble de mots de façon compacte est d'ailleurs largement exploitée par les linguistes pour la compression des dictionnaires.

On peut aller plus loin en décrivant des ensembles infinis de mots. Supposons que X contienne les mots a, aba, $ababa$, etc., c'est-à-dire tous les mots qu'on peut écrire en alternant un a avec un b et ce jusqu'à un dernier a. Cet ensemble peut être décrit par l'automate :

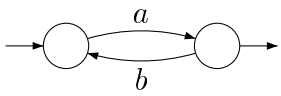


FIG. 5 – Automate pour $X = \{a, aba, ababa, \dots\}$.

On peut aussi représenter l'ensemble X des mots à l'intérieur desquels les blocs de a consécutifs sont de longueur paire comme par exemple $baaaab$ et $bbaabaabbbaaaa$. Ceci est fait par l'automate :

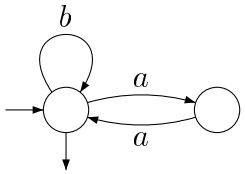


FIG. 6 – Automate pour $X = (b + aa)^*$.

Ce dernier ensemble X est aussi décrit par l'expression rationnelle $(b + aa)^*$ qui exprime que tout mot de X peut être obtenu en mettant bout à bout des b et des blocs aa. De même, l'ensemble précédent est décrit par l'expression rationnelle $a(ba)^*$. En effet, un mot de cet ensemble est obtenu en mettant un a suivi d'un nombre quelconque de blocs ba. Ceci pourrait être fait pour n'importe quel automate. Les initiés au domaine savent bien que les automates sont équivalents aux expressions rationnelles. Ceci signifie que tout ensemble de mots décrit par un automate peut aussi l'être par une expression rationnelle et inversement. Il existe d'autres façons équivalentes de décrire des ensembles de mots comme les formules de logique.

L'intérêt des automates par rapport aux autres méthodes pour décrire des ensemble est qu'ils rendent effectives nombre d'opérations sur ces ensembles. Par exemple, étant données deux descriptions d'ensemble, une question naturelle est de savoir si ces deux descriptions donnent en fait le même ensemble. Si les ensembles sont donnés par deux automates, il est aisé de savoir s'ils

sont égaux. Si au contraire, ils sont donnés par deux expressions rationnelles, c'est plus délicat. Dans ce dernier cas, la bonne méthode consiste en fait à convertir chacune des expressions en un automate équivalent puis à effectuer l'opération sur les automates.

Cette effectivité a bien sûr un prix. Tous les ensembles ne peuvent pas être décrits par un automate. Par exemple, l'ensemble $X = \{a^n b^n \mid n \in \mathbb{N}\}$ ne peut pas l'être à moins d'autoriser des automates eux-mêmes infinis.

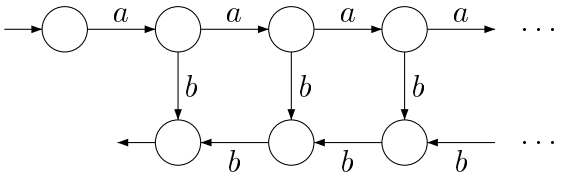


FIG. 7 – Automate pour $X = \{a^n b^n \mid n \in \mathbb{N}\}$.

Dans la suite, on va imposer aux automates d'être finis et on interdira des automates tels que celui de la figure 7. Il s'avère que l'ensemble $X = \{a^n b^n \mid n \in \mathbb{N}\}$ ne peut pas être représenté par un automate fini.

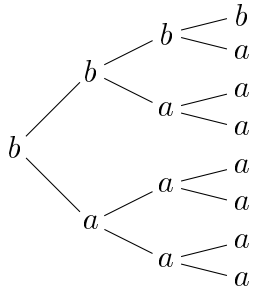


FIG. 8 – Un arbre.

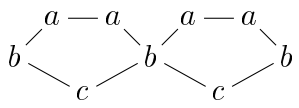


FIG. 9 – Une trace.

Dans les exemples précédents, nous avons utilisé des automates pour décrire ou définir des ensembles de mots. Il est possible de remplacer les mots par d'autres types d'objets tels les arbres (cf. figure 8 et [82]), les traces (cf. figure 9 et [38]) ou des mots bi-dimensionnels (cf. figure 10 et [46]). Le cadre

$a\,a\,b\,a$
$a\,b\,a\,b$
$a\,a\,b\,a$

FIG. 10 – Un mot bi-dimensionnel.

général est le suivant : on dispose d'une classe d'objets fixée *a priori*. Ces objets peuvent être des mots, des arbres, des traces ou des tableaux de caractères. À chaque automate est associée une partie de ces objets qui est donc décrite par l'automate. Un automate peut être vu comme une façon concise de décrire une partie de ces objets qui peut être finie ou infinie.

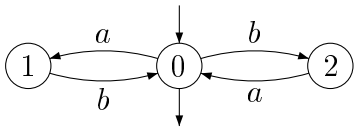


FIG. 11 – Machine à trois états.

Jusqu'ici, un automate a été présenté comme un graphe avec des sommets et des flèches étiquetées avec des lettres. De plus, certains sommets sont désignés comme initiaux par une flèche entrante alors que d'autres sont désignés comme finaux par une flèche sortante. L'automate décrit l'ensemble des mots qui peuvent être lus le long d'un chemin d'un sommet initial à un sommet final. Cette vision d'un automate est complètement statique.

Cette vision peut être rendue dynamique en considérant un automate comme la description d'une machine. Les sommets représentent les différents états possibles et les flèches sont les transitions que la machine est capable d'effectuer. L'automate de la figure 11 représente une machine qui a trois états possibles. Les lettres des transitions doivent être interprétées comme des données ou des événements extérieurs qui dirigent les changements d'états de la machine. La machine peut seulement effectuer une transition d'étiquette a lorsqu'elle reçoit la lettre a en entrée. On dira que la machine *lit* la lettre a. La suite des lettres que lit la machine au cours d'un calcul forme un mot qu'on appellera mot d'entrée. Considérons, par exemple, la machine de la figure 11 et supposons que le mot d'entrée est $x = abba$.

Les différentes étapes du calcul de la machine sont représentées sur la figure 12. À chaque étape, l'état dans lequel se trouve la machine est marqué en noir. Au départ, la machine se trouve dans l'état 0 qui est initial. Ensuite, la machine effectue successivement des transitions d'étiquettes a, b, b et a. À la fin de ce calcul, la machine se trouve dans un état qui est final. Pour cette raison, le calcul est dit acceptant.

Cette façon dynamique de considérer un automate consiste à voir un automate comme une machine de Turing très particulière à une seule bande.

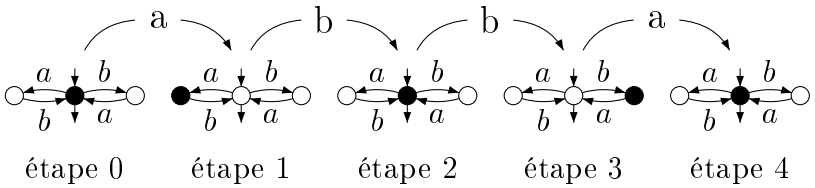


FIG. 12 – Le calcul de l'automate pour $x = abba$.

Un automate est une machine de Turing dont le type de transitions autorisées a été énormément restreint. D'une part, l'automate peut lire la bande sur laquelle a été placé le mot d'entrée, mais il ne peut pas y écrire. D'autre part, à chaque transition, la tête de lecture avance d'une position vers la droite. Les lettres du mots d'entrée sont lues dans l'ordre et chacune est lue exactement une fois. La machine effectue autant de transitions qu'il y a de lettres dans le mot d'entrée.

Jusqu'à maintenant, nous avons utilisé les automates pour décrire des ensembles de mots finis, c'est-à-dire des suites finies de lettres. Nous allons voir qu'un automate peut également décrire des suites infinies de lettres. Par exemple, l'automate de la figure 5 décrit l'ensemble $X = \{a, aba, ababa, \dots\}$. Il décrit aussi le mot infini $x = ababab\dots$ obtenu en répétant infiniment le bloc ab. Plus généralement, un automate décrit un ensemble de mots infinis qui peut contenir un seul mot comme dans l'exemple précédent, plusieurs ou même un nombre infini. Ces mots sont les étiquettes des chemins infinis dans l'automate. Par exemple, l'automate de la figure 6 décrit l'ensemble des mots infinis dont les blocs finis de a consécutifs sont de longueur paire. Autant le rôle joué par les sommets initiaux reste clair dans ce cadre, autant celui joué par les sommets finaux ne l'est plus. Ce point un peu délicat sera développé ultérieurement.

Dès qu'on parle d'infini, surgit le problème de savoir de quel type d'infini il s'agit. Lorsqu'on a écrit le mot $x = ababab\dots = (ab)^\omega$ en répétant infiniment le bloc ab, on a implicitement supposé que l'on écrivait de la gauche vers la droite. Même si c'est moins naturel, on aurait pu écrire de la droite vers la gauche et obtenir le mot $x' = \dots ababab = {}^\omega(ab)$ qui est bien différent du mot x. On aurait aussi pu écrire de gauche à droite en commençant à l'infini à gauche et en terminant à l'infini à droite pour obtenir le mot $\dots ababab\dots = {}^\omega(ab)^\omega$ qui est aussi le résultat de la concaténation de x' et de x. En poursuivant cette idée, on aurait tout aussi bien écrire une première copie de x suivie d'une deuxième pour obtenir le mot $abab\dots \;\; abab\dots$ ou bien écrire une copie de x suivie d'une copie de x' pour obtenir un mot

$abab\ldots \quad \ldots abab = (ab)^{\omega\omega}(ab)$. Dans ce travail, on s'intéressera essentiellement aux types d'infinis définis par l'ordre naturel des entiers et par les ordinaux dénombrables. Nous aborderons également les ordres linéaires qui permettent de prendre en compte les différents infinis que nous venons de voir.

3 Automates et mots infinis

La notion d'automate pour les mots finis est relativement naturelle. Dans un tel automate, un chemin est acceptant s'il commence dans un état initial et s'il se termine dans un état final. Dans un automate sur les mots infinis, un chemin est une suite infinie d'états. Il n'est plus possible d'imposer à un chemin acceptant de se terminer dans un état final puisque ce chemin est infini et qu'il n'a pas de dernier état. Pour les mots infinis, plusieurs variantes d'automates ont été introduites qui correspondent à différentes façons de remplacer cette condition sur la fin des chemins par quelque chose d'adapté aux chemins infinis. Dans toutes ces variantes, un automate possède des états, des transitions entre ces états, des états initiaux et une condition d'acceptation. Un chemin dans un automate est alors naturellement une suite d'états qui est compatible avec ses transitions. Ces variantes d'automates se différencient par leur condition d'acceptation, c'est-à-dire par leur façon de spécifier quand un chemin commençant en un état initial est acceptant. Ces différentes conditions d'acceptation ont en commun qu'elles portent sur les états infiniment répétés le long du chemin. Nous donnons maintenant la définition générique d'un automate sur les mots infinis de façon indépendante de la condition d'acceptation. Nous fixons également quelques notations.

Définition 1 *Un* automate *sur les mots infinis est un automate* (Q, A, E, I, Φ) *où* Q *est un ensemble fini d'états,* A *un alphabet fini,* $E \subset Q \times A \times Q$ *l'ensemble des transitions,* I *est l'ensemble des états initiaux et où* Φ *est la condition d'acceptation.*

Soit $\mathcal{A} = (Q, A, E, I, \Phi)$ un tel automate et soit $x = a_0a_1a_2\ldots$ un mot infini. Un chemin γ d'étiquette x dans $\mathcal{A}$ est une suite d'états $q_0, q_1, q_2, \ldots$ telle que pour tout entier k, $q_k \xrightarrow{a_k} q_{k+1}$ soit une transition de $\mathcal{A}$. Comme pour les automates de mots finis, le chemin γ est *initial* si q_0 est initial. Le chemin γ est *final* s'il vérifie la condition d'acceptation Φ. Il est finalement *acceptant* s'il est à la fois initial et final. Un mot est *accepté* par l'automate s'il est l'étiquette d'un chemin acceptant.

Pour un chemin γ, on définit l'ensemble $\lim \gamma$ des états qui apparaissent infiniment souvent dans γ. Un état q appartient à cet ensemble si pour tout

entier n, il existe un entier k, plus grand que n, tel que $q = q_k$.

$$\lim \gamma = \{q \mid \forall n \geq 0 \ \ \exists k \geq n \ \ q = q_k\}.$$

Comme l'ensemble des états est fini, cet ensemble $\lim \gamma$ ne peut pas être vide.

La condition d'acceptation Φ porte sur cet ensemble $\lim \gamma$ des états qui ont une infinité d'occurrences dans γ. Cet ensemble d'états peut être vu comme un état fictif qui est ajouté à la fin du chemin et qui devient alors le dernier état du chemin. Cet état fictif comble la lacune des chemins infinis qui n'ont pas de dernier état. On verra que cette méthode est utilisée de façon systématique pour les automates sur les mots transfinis.

Büchi [22] a tout d'abord introduit la condition d'acceptation la plus simple. Pour un automate de Büchi, la condition d'acceptation Φ consiste uniquement en un ensemble F d'état finaux. La définition d'un automate de Büchi est formellement identique à celle d'un automate de mots finis. C'est l'utilisation même de l'automate qui est différente puisqu'un automate de Büchi accepte des mots infinis et non pas des mots finis. Avec cette condition d'acceptation, un chemin γ est final s'il passe infiniment souvent par au moins un état final, c'est-à-dire si $\lim \gamma \cap F \neq \varnothing$.

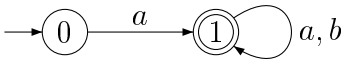


FIG. 13 – Automate de Büchi pour aA^ω.

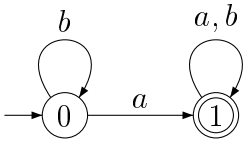


FIG. 14 – Automate de Büchi pour A^*aA^ω.

Exemple 2 Les figures 13, 14, 15 et 16 représentent des automates de Büchi qui acceptent les ensembles de mots aA^ω, A^*aA^ω, A^*b^ω et $(A^*a)^\omega$. Ces ensembles sont respectivement l'ensemble des mots commençant par un a, l'ensemble des mots ayant au moins une occurrence de a, l'ensemble des mots ayant un nombre fini d'occurrences de a et finalement l'ensemble des mots ayant un nombre infini d'occurrences de a. Sur les figures, les états initiaux

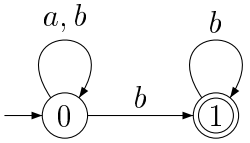


FIG. 15 – Automate de Büchi pour A^*b^ω.

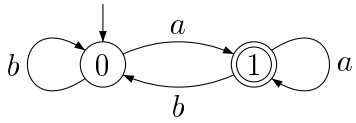


FIG. 16 – Automate de Büchi pour $(A^*a)^\omega$.

sont toujours marqués par une flèche entrante alors que les états finaux sont maintenant marqués par un double cercle.

L'inconvénient de cette condition d'acceptation est qu'elle est trop faible lorsqu'on considère des automates déterministes. L'automate de la figure 15 accepte l'ensemble $X = A^*b^\omega$ des mots qui ont un nombre fini de a. Le complémentaire de X est l'ensemble $(A^*a)^\omega$ qui est accepté par l'automate déterministe de la figure 16. Par contre, il n'est pas très difficile de montrer que l'ensemble X ne peut pas être accepté par un automate de Büchi déterministe. Il est toutefois accepté par un automate de Muller déterministe dont la définition sera donnée à la section suivante. L'approche pour la complémentation via la déterminisation est donc vouée à l'échec si l'on se restreint aux automates de Büchi.

La raison intuitive est que cette condition peut forcer un chemin à passer par certains états infiniment souvent. Par contre elle ne peut pas interdire à un tel chemin de passer infiniment souvent dans d'autres états. Ce manque de symétrie explique pourquoi certains langages rationnels sont acceptés par un automate de Büchi déterministe alors que leur complémentaire ne l'est pas.

3.1 Calcul de l'indice de Rabin

Le travail présenté dans cette partie a été réalisé en collaboration avec Ramón Maceiras dans le cadre de son DEA.

Nous avons décrit la condition d'acceptation dite de Büchi qui est donnée par un ensemble d'états finaux. Celle-ci est simple mais elle n'est pas suffisante pour les automates déterministes. Certains ensembles rationnels ne peuvent pas être acceptés par un automate de Büchi déterministe. Pour

les automates déterministes, il faut considérer d'autres conditions d'acceptation comme celle de Muller ou de Rabin. Comme la condition de Büchi, ces conditions portent sur les états infiniment répétés le long d'un chemin. Le fait d'être acceptant pour un chemin γ ne dépend que de la valeur de l'ensemble $\lim \gamma$. Les valeurs de cet ensemble pour lesquelles le chemin est acceptant sont appelées les ensembles d'états *acceptants*. Un ensemble d'états est, par exemple, acceptant avec une condition de Büchi s'il contient au moins un des états finaux. Les différentes conditions d'acceptation diffèrent uniquement par la façon dont elles spécifient ces ensembles acceptants. Toutes ces conditions ont bien sûr le même pouvoir d'expression puisqu'elles définissent tous les ensembles reconnaissables. Par contre, elles ont des comportements différents d'un point de vue algorithmique. La donnée d'un ensemble reconnaissable sous la forme d'un automate ayant telle ou telle condition d'acceptation peut radicalement changer la complexité de calcul de certains paramètres. Nous allons nous intéresser dans cette partie au calcul de l'indice de Rabin d'un ensemble reconnaissable de mots infinis. Nous commençons par passer rapidement en revue ces différentes conditions d'acceptation.

Muller [59] introduisit la condition qui porte maintenant son nom, mais c'est McNaughton qui montra que tout ensemble rationnel de mots infinis est accepté par un automate déterministe avec cette condition [55]. Cette condition est constituée par une famille d'ensembles d'états appelée *table*. Avec cette condition, un chemin est final si l'ensemble des états infiniment répétés le long du chemin appartient à la table. Cette condition est donc la plus explicite puisqu'elle consiste en une énumération directe des ensembles acceptants.

Rabin [70] introduisit initialement sa condition pour les automates d'arbres, mais elle peut aussi être utilisée pour les automates de mots. Elle consiste en une famille de paires d'ensembles d'états. Un chemin γ est final pour cette condition s'il existe une paire (L_i, U_i) de la famille telle que γ passe infiniment souvent par au moins un état de U_i, mais passe un nombre fini de fois par tous les états de L_i. La condition duale de la condition de Rabin est appelée condition de Streett [80].

La dernière condition que nous considérons est la condition dite *à parité*. Elle fut initialement introduite par Mostowski [57] et elle est aussi appelée condition de Rabin à chaîne. Elle est donnée par une fonction π qui associe un entier $\pi(q)$ à chaque état q de l'automate. Un chemin est alors final si la plus grande valeur qui apparaît infiniment le long du chemin est impaire. Cette condition est une généralisation de la condition de Büchi. En effet, si on associe 1 à tous les états finaux et 0 aux autres, on retrouve la condition de Büchi. La condition à parité est aussi un cas particulier de la condition de Rabin. Si les ensembles L_i et U_i sont définis par $L_i = \{q \mid \pi(q) \geq 2i\}$ et

$U_i = \{q \mid \pi(q) \geq 2i-1\}$, la condition de Rabin $\{(L_1, U_1), \ldots, (L_m, U_m)\}$ où $m = \lfloor (\max_{q \in Q} \pi(q) + 1)/2 \rfloor$ définit les mêmes ensembles acceptants que la condition à parité donnée par la fonction π. On remarque que la condition de Rabin équivalente est particulière puisque les ensembles L_i et U_i forment une suite décroissante. On a en effet la chaîne d'inclusions $U_1 \supseteq L_1 \supseteq U_2 \supseteq \cdots \supseteq U_m \supseteq L_m$. La condition à parité a été utilisée indépendamment en [44] et [58] pour obtenir des stratégies sans mémoire.

Dans la suite de cette partie, nous ne considérons plus que des automates déterministes avec des conditions d'acceptation de Rabin, de Muller ou à parité. La condition de Rabin introduit de manière naturelle une hiérarchie parmi les ensembles reconnaissables de mots infinis. Le nombre de paires utilisées dans une condition de Rabin constitue en quelque sorte une mesure de la complexité de l'automate. Bien entendu, un même ensemble de mots peut être accepté par deux automates ayant des nombres différents de paires dans leurs conditions d'acceptation respectives. Pour un ensemble donné de mots infinis, le nombre minimal de paires nécessaires dans un automate acceptant constitue par contre une mesure intrinsèque de sa complexité. Cet entier est appelé l'*indice de Rabin*.

Les classes de la hiérarchie définie par l'indice de Rabin sont en fait des classes particulières d'une hiérarchie beaucoup plus fine découverte par Wagner [85]. La hiérarchie de Wagner est elle-même la trace sur les ensembles rationnels d'une hiérarchie de théorie descriptive des ensembles introduite par Wadge [84]. Ainsi, les classes induites par l'indice de Rabin ont une interprétation topologique et elles raffinent la hiérarchie de Borel.

Il a été prouvé par Krishnan *et al.* [51] que le calcul de l'indice de Rabin à partir d'un automate de Rabin déterministe reconnaissant l'ensemble est NP-complet. Cette complexité traduit le fait que l'automate de Rabin qui réalise le nombre minimal de paires dans sa condition d'acceptation peut être totalement différent de l'automate donné en entrée. D'un autre côté, la méthode développée par Krishnan *et al.* permet de calculer en temps polynomial l'indice de Rabin à partir d'un automate de Muller déterministe. Un autre algorithme plus efficace a été donné par Wilke et Yoo [89]. Le temps de calcul de leur algorithme est $O(m^2nc)$ où m, n et c sont respectivement le nombre d'états de l'automate, son nombre d'ensembles acceptants et le cardinal de l'alphabet.

La différence entre les deux conditions d'acceptation montre que la condition de Rabin est beaucoup plus compacte. La taille d'un automate de Muller équivalent à un automate de Rabin peut être exponentielle. Ceci traduit le fait que la condition de Muller énumère explicitement tous les ensembles acceptants qui peuvent être très nombreux.

La condition à parité définit également une mesure de complexité d'un

automate. Celle-ci est donnée par la valeur maximale $\max_{q\in Q}\pi(q)$ atteinte par les entiers associés aux états. Cette valeur dépend de l'automate considéré. Pour un ensemble de mots infinis, il faut prendre la valeur minimale de cette mesure parmi les automates le reconnaissant, valeur que nous appellerons *indice de parité*. Il s'avère que l'indice de Rabin et l'indice de parité sont étroitement liés. Si les indices de Rabin et de parité d'un ensemble X sont respectivement notés $\mathrm{ind}(X)$ et $\mathrm{m}(X)$, on a la relation $\mathrm{ind}(X) = \lfloor(\mathrm{m}(X)+1)/2\rfloor$. L'indice de parité détermine complètement l'indice de Rabin.

Nous avons montré [30] que l'indice de parité et, par conséquent, l'indice de Rabin peuvent être calculés en temps polynomial à partir d'un automate à parité déterministe. La complexité de l'algorithme est $O(n^2c)$ où n et c sont respectivement le nombre d'états de l'automate et le cardinal de l'alphabet.

J'avais prouvé dans ma thèse [27] qu'il est possible de minimiser la condition d'acceptation d'un automate à parité. Ceci signifie que pour un automate à parité de fonction de parité π, il est possible de trouver une autre fonction de parité π' telle que π et π' définissent les mêmes ensembles acceptants et telle que $\max_{q\in Q}\pi'(q)$ soit égal à l'indice de parité de l'ensemble accepté par l'automate. Nous avons finalement prouvé le résultat suivant [30].

Théorème 3 *L'indice de Rabin d'un ensemble de mots infinis accepté par un automate à parité déterministe (Q, A, E, q_0, π) et la fonction de parité minimale π' équivalente à π peuvent être calculés en temps $O(n^2c)$.*

3.2 Automates non ambigus

Le travail présenté dans cette partie a été réalisé en collaboration avec Max Michel.

Un problème important sur les automates est le problème de la complémentation. Il consiste, pour un automate $\mathcal{A}$ donné, à trouver un autre automate $\mathcal{A}'$ qui accepte le complémentaire de l'ensemble accepté par $\mathcal{A}$. Ce problème apparaît naturellement lorsque les automates sont utilisés pour des questions de décidabilité de certaines logiques comme dans [22, 24]. Une formule est alors traduite en un automate et les négations présentes dans la formule correspondent à des complémentations des ensembles acceptés par les automates.

Nous allons d'abord exposer la solution classique dans le cas des mots finis. Le problème est trivial si l'automate $\mathcal{A}$ donné est déterministe et complet. Il suffit alors d'interchanger états finaux et états non finaux de l'automate. Si l'automate n'est pas complet, il est aisé de le compléter en ajoutant un nouvel état *puits*. Par contre, si l'automate n'est pas déterministe, il faut trouver

un automate déterministe équivalent. La méthode classique pour obtenir un tel automate est d'appliquer la *construction par sous-ensembles* [1].

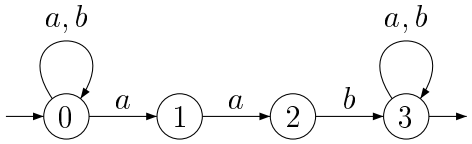
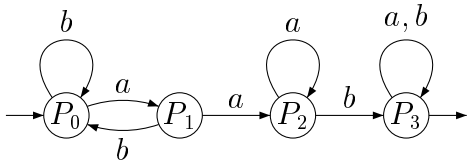


FIG. 17 – Automate non déterministe pour A^*aabA^*.



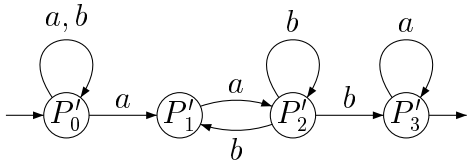
$P_0 = \{0\}$ $P_1 = \{0, 1\}$ $P_2 = \{0, 1, 2\}$ $P_3 = \{0, 3\}$

FIG. 18 – Automate déterministe pour A^*aabA^*.

Considérons l'automate non déterministe de la figure 17 qui accepte l'ensemble $X = A^*aabA^*$ des mots ayant un facteur aab. En appliquant la construction par sous-ensembles à cet automate, on obtient essentiellement l'automate déterministe et complet de la figure 18 qui accepte bien sûr le même ensemble de mots. Pour être précis, on obtient en fait un automate avec deux états en plus qui peuvent être identifiés avec l'état P_3. Un automate pour accepter le complémentaire de X peut facilement être déduit de cet automate déterministe. Il suffit de remplacer l'ensemble $F = \{P_3\}$ des états finaux par le nouvel ensemble $F' = \{P_0, P_1, P_2\}$.

De manière générale, si $\mathcal{A} = (Q, A, E, I, F)$ est un automate déterministe et complet qui accepte un ensemble X de mots finis, l'automate $(Q, A, E, I, Q \setminus F)$ obtenu en échangeant les états finaux accepte le complémentaire $A^* \setminus X$ de X. La propriété essentielle est que dans un automate déterministe et complet, tout mot x étiquette un unique chemin initial $\gamma(x)$. Le mot x est alors accepté si et seulement si ce chemin $\gamma(x)$ est acceptant (c'est-à-dire final puisqu'il est déjà initial). En complémentant F, le chemin $\gamma(x)$ devient final s'il ne l'était pas et inversement.

La notion d'automate déterministe privilégie la lecture des mots de gauche à droite. Puisque gauche et droite jouent des rôles symétriques pour les mots finis, on aurait pu aussi lire les mots de droite à gauche et considérer des automates co-déterministes. Rappelons qu'un automate est *co-déterministe*



$P'_0 = \{0,3\}\ P'_1 = \{1,3\}\ P'_2 = \{2,3\}\ P'_3 = \{3\}$

FIG. 19 – Automate co-déterministe pour A^*aabA^*.

s'il a un unique état final et si pour tout état q et pour toute lettre a, il existe au plus un état p tel que $p \xrightarrow{a} q$ soit une transition. On peut bien sûr appliquer (de façon duale) la construction par sous-ensembles pour obtenir un automate co-déterministe équivalent à un automate donné. Par exemple, si on applique la méthode à l'automate de la figure 17, on obtient essentiellement celui de la figure 19. Une fois obtenu un automate co-déterministe, le problème de la complémentation se résout aisément. Il suffit d'échanger les états initiaux de l'automate pour qu'il accepte le complémentaire. Dans l'automate co-déterministe de la figure 19, seul l'état P'_0 est initial. Si on remplace $\{P'_0\}$ par $\{P'_1, P'_2, P'_3\}$ comme ensemble d'états initiaux, l'automate accepte alors les mots qui ne contiennent pas le facteur aab.

Le fait d'être déterministe pour un automate est une propriété locale. Elle ne dépend que des états initiaux et des transitions sortantes de chaque état. *Mutatis mutandis*, ceci s'applique aussi au co-déterminisme. Le déterminisme implique que chaque mot fini ou même infini est l'étiquette d'un seul chemin initial dans l'automate. Cette dernière propriété est par contre une propriété globale et il y a donc un passage du local au global. Il est encore vrai que tout mot fini est l'étiquette d'un seul chemin final dans un automate co-déterministe, mais ce n'est, par contre, plus vrai pour les mots infinis.

Pour les automates sur les mots infinis, le problème de la complémentation est plus compliqué que pour les automates sur les mots finis. Il existe trois approches classiques pour résoudre ce problème [83]. La première approche due à Büchi permet de passer directement d'un automate de Büchi à un autre automate de Büchi pour le complémentaire. Elle utilise une congruence et est basée sur un argument combinatoire dû à Ramsey [71]. L'approche basée sur des ω-semigroupes peut être considérée comme une variante de cette approche. La seconde approche due initialement à McNaughton [55] puis améliorée par Safra [74] consiste à passer d'un automate de Büchi à un automate de Muller [59] déterministe reconnaissant le même ensemble. La complémentation est alors rendue triviale par le déterminisme de l'automate. Cette approche implique des constructions élaborées sur les automates.

La construction due à Safra est particulièrement astucieuse. La troisième approche est basée sur les automates alternants introduits par Muller et Schupp [60] pour lesquels la complémentation est également triviale. Elle fut ensuite reprise par Kupferman et Vardi [52] puis par Thomas [83].

Pour les mots finis, la construction par sous-ensembles permet de passer d'un automate non déterministe à un automate déterministe. La méthode de McNaughton [55] en est une extension aux mot infinis. Elle permet de passer d'un automate de Büchi non déterministe à un automate de Muller déterministe. Il reste cependant un fossé entre mots finis et mots infinis. D'une part, la déterminisation pour les mots infinis nécessite de passer à une condition d'acceptation plus compliquée. D'autre part, la complexité dans le pire cas est différente puisqu'elle est de l'ordre de 2^n pour les mots finis et de n^n pour les mots infinis.

Pour les mots finis, la symétrie entre la gauche et la droite autorise à utiliser la construction par sous-ensembles de façon duale pour obtenir un automate co-déterministe. Par contre, la méthode de McNaughton ne peut pas être appliquée pour obtenir un automate co-déterministe. En effet, la symétrie pour les mots finis entre la gauche et la droite n'existe plus pour les mots infinis. Ceux-ci sont infinis à droite et non à gauche. Ces mots ont une première lettre mais pas de dernière lorsque l'on les lit de gauche à droite. La question est alors de trouver des automates qui d'une certaine façon liraient les mots infinis de droite à gauche et qui donc commenceraient la lecture par la partie infinie.

Définition 4 *Un automate de Büchi $\mathcal{A}$ est dit* non-ambigu *(respectivement* complet*) si et seulement si tout mot infini x étiquette au plus (respectivement au moins) un chemin final dans $\mathcal{A}$. Si $\mathcal{A}$ est non-ambigu, cet unique chemin est noté $\gamma(x)$.*

De manière générale, la notion de non-ambiguïté pour une machine correspond à l'existence d'au plus un calcul pour chaque entrée possible. Au contraire, la notion de complétude correspond plutôt à l'existence d'au moins un calcul. Il existe de nombreuses variantes pour la définition de la non-ambiguïté. Par, exemple dans [16], un automate de mots finis est dit non-ambigu si pour toute paire (p, q) d'états et tout mot fini w, il existe au plus un chemin de p à q d'étiquette w. La notion introduite ici est plus forte puisque l'unicité du chemin est demandée indépendamment du point de départ et du point d'arrivée du chemin. Il est juste requis l'unicité d'un chemin final.

Il faut remarquer que les propriétés de la définition ci-dessus sont des propriétés globales et non pas locales. Ce sont ces propriétés qui sont réellement requises pour la complémentation car une propriété locale ne suffirait pas à

les garantir pour les automates de mot infinis.

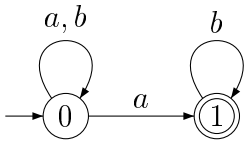


FIG. 20 – Automate co-déterministe mais non complet.

Il n'est pas très difficile de montrer qu'un automate non-ambigu est nécessairement co-déterministe. Si, de plus, l'automate est complet, il existe, pour chaque état q et chaque lettre a, un unique état p tel que $p \xrightarrow{a} q$ soit une transition. Par contre, la réciproque est fausse comme le montre l'automate de la figure 20. Cet automate n'est pas complet puisque le mot a^ω n'étiquette aucun chemin final. Ceci contraste avec la situation pour les mots finis. Chaque mot fini étiquette exactement un chemin dans un automate qui a un seul état final et où il existe une unique transition $p \xrightarrow{a} q$ pour chaque paire (a, q).

Si l'automate $\mathcal{A}$ est non-ambigu et complet, l'unique chemin final étiqueté par ax est obtenu en prolongeant à gauche d'une transition d'étiquette a le chemin final étiqueté par x. Pour cette raison, on peut considérer que ces automates lisent les mots infinis de la droite vers la gauche.

L'intérêt principal des automates non-ambigus et complets est qu'ils permettent la complémentation de façon immédiate. Il suffit en effet d'échanger les états initiaux pour complémenter. On peut remarquer que les propriétés d'ambiguïté et de complétude dépendent uniquement des transitions et des états finaux de l'automate. Elle ne dépendent pas de ses états initiaux. Si l'automate non-ambigu et complet $\mathcal{A} = (Q, A, E, I, F)$ accepte l'ensemble X, l'automate $(Q, A, E, Q \setminus I, F)$ est également non-ambigu et complet. Il accepte en outre le complémentaire de X.

Puisque ces automates sont co-déterministes, il constituent l'analogue des automates de Muller déterministes. Par contre, ils utilisent la condition d'acceptation de Büchi qui est beaucoup plus simple que la condition de Muller. Le théorème de McNaughton établit que tout ensemble reconnaissable de mots infinis peut être accepté par un automate de Muller déterministe. Le théorème que nous avons obtenu [31] donne la même chose pour les automates de Büchi non-ambigus et complets.

Théorème 5 *Tout ensemble reconnaissable de mots infinis est accepté par un automate de Büchi non-ambigu et complet.*

Deux preuves de ce théorème sont données en [31]. La première preuve utilise les graphes et les stratégies alors que la seconde utilise les semigroupes. Les deux preuves sont effectives dans le sens où elles donnent deux algorithmes calculant un automate de Büchi non-ambigu et complet reconnaissant un ensemble X de mots infinis. Les deux algorithmes ne partent pas de la même entrée. Le premier algorithme suppose que l'ensemble X est donné sous la forme d'un automate de Büchi qui le reconnaît alors que le second suppose qu'il est donné par un morphisme de A^+ dans un semigroupe fini qui le reconnaît.

Les deux preuves utilisent une première réduction du problème en introduisant des automates de Büchi généralisés où la condition d'acceptation est donnée par une famille d'ensembles d'états finaux au lieu d'un seul ensemble dans les automates de Büchi usuels. Un chemin est alors final si au moins un état de chacun de ces ensembles apparaît infiniment souvent le long du chemin. Ces automates de Büchi généralisés se révèlent plus pratiques pour construire des automates non-ambigus. En outre, il est facile de passer d'un automate de Büchi généralisé non-ambigu et complet à un automate de Büchi non-ambigu et complet qui lui est équivalent.

Les trois automates de Büchi des figures 13, 14 et 16 sont ambigus. Pour chacun d'eux, le mot infini a^ω est l'étiquette d'au moins deux chemins finaux, l'un partant de l'état 0 et un autre de l'état 1. Nous allons donner trois automates de Büchi non-ambigus et complets qui sont équivalents aux automates des figures 13, 14 et 16. Il faut remarquer que ces automates sont relativement différents de ceux des figures 13, 14 et 16 et qu'ils ne correspondent pas aux automates que l'on construirait de manière intuitive.

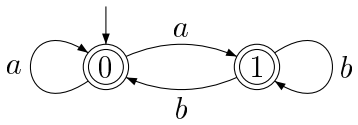


FIG. 21 – Automate de Büchi non-ambigu et complet pour aA^ω.

Pour l'automate de la figure 21, il est aisé de vérifier directement qu'il est non-ambigu et complet. Cela peut s'avérer plus délicat pour des automates plus compliqués. La proposition suivante donne une caractérisation des automates non-ambigus et complets. Soit $\mathcal{A}$ l'automate de Büchi (Q, A, E, I, F) et soit q un de ses états. On note $\mathcal{A}_q$ l'automate $(Q, A, E, \{q\}, F)$ obtenu en prenant le singleton $\{q\}$ comme ensemble d'états initiaux à la place de I. L'ensemble X_q accepté par $\mathcal{A}_q$ est l'ensemble des mots qui étiquettent un chemin final de $\mathcal{A}$ partant de l'état q. Il s'avère que l'automate $\mathcal{A}$ est non-ambigu et complet si et seulement si la famille d'ensembles $(X_q)_{q\in Q}$ forme une par-

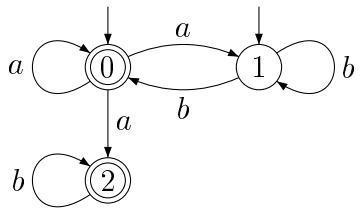


FIG. 22 – Automate de Büchi non-ambigu et complet pour A^*aA^ω.

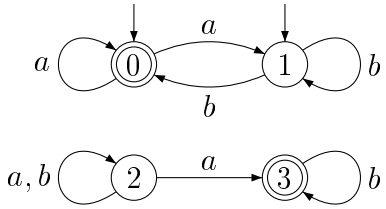


FIG. 23 – Automate de Büchi non-ambigu et complet pour $(A^*a)^\omega$.

tition de A^ω. Ceci permet de décider si un automate donné est non-ambigu et complet.

Cette remarque permet de voir d'une façon différente que si on échange les états initiaux dans un automate de Büchi non-ambigu et complet, alors, l'automate accepte le complémentaire. En effet, l'automate accepte par définition $X = \bigcup_{q \in I} X_q$. Si la famille $(X_q)_{q \in Q}$ est une partition, alors l'union $\bigcup_{q \notin I} X_q$ est effectivement le complémentaire de X.

La remarque précédente permet de vérifier sans difficulté que les deux automates des figures 22 et 23 sont non-ambigus et complets.

3.3 Prédicats morphiques

Le travail présenté dans cette partie a été réalisé en collaboration avec Wolfgang Thomas lors d'un séjour à Aix-la-Chapelle.

Les automates sur les mots infinis furent introduits par Büchi [22] pour résoudre des problèmes de décision de certaines logiques. Büchi s'intéressait en particulier à la logique monadique du second ordre de la structure $\langle \mathbb{N}, < \rangle$ des entiers munis de l'ordre. Nous allons revenir dans cette partie à ce problème initial en montrant que la structure $\langle \mathbb{N}, < \rangle$ peut être enrichie de certains prédicats unaires appelés morphiques tout en préservant la décidabilité de la logique monadique du second ordre.

La méthode due à Büchi consiste à traduire une formule close ϕ de la

logique monadique du second ordre de $\langle \mathbb{N}, < \rangle$ en un automate de Büchi $\mathcal{A}$ tel que ϕ soit valide dans $\langle \mathbb{N}, < \rangle$ si et seulement si $\mathcal{A}$ possède au moins un chemin acceptant. Cette dernière propriété peut bien sûr être testée facilement.

Il fut rapidement observé que cette méthode est applicable dans la situation plus générale où la structure $\langle \mathbb{N}, < \rangle$ est étendue en la structure $\langle \mathbb{N}, <, P \rangle$ où $P \subseteq \mathbb{N}$ est un prédicat unaire fixé. On considère une formule $\phi(X)$ ayant une variable libre X du second ordre. En appliquant la méthode de Büchi, cette formule est traduite en un automate de Büchi $\mathcal{A}$ sur l'alphabet d'entrée $\{0, 1\}$. La formule $\phi(X)$ est alors valide dans la structure $\langle \mathbb{N}, < \rangle$ en prenant P comme interprétation de la variable X si et seulement si $\mathcal{A}$ accepte le mot caractéristique x_P du prédicat P. Rappelons que le mot caractéristique x_P de P est défini de la façon suivante. La k-ième lettre de x_P est 1 si k appartient à P et 0 sinon. La théorie monadique du second ordre de la structure $\langle \mathbb{N}, <, P \rangle$ est alors décidable si l'on peut décider si le mot caractéristique x_P est accepté par un automate de Büchi donné. Elgot et Rabin [43] donnèrent plusieurs prédicats se prêtant à cet approche parmi lesquels l'ensemble des nombres factoriels $n!$, l'ensemble des puissances k-ièmes n^k pour un entier k fixé et les puissances k^n d'un entier k fixé.

Les remarques précédentes invitent à considérer le problème de décision suivant pour un mot infini x fixé.

(Acc_x) Est-ce qu'un automate de Büchi donné accepte le mot x ?

Si le problème (Acc_x) est décidable, cela signifie intuitivement que le mot infini x peut être utilisé comme un oracle externe dans les systèmes à états finis non-terminants sans perdre les résultats de décidabilité.

Pour résoudre le problème (Acc_{x_P}), Elgot et Rabin et ensuite Siefkes [77] utilisèrent une approche de la théorie des automates appelée *méthode de contraction*. Elle consiste à réduire le problème au cas d'un mot ultimement périodique pour lequel le problème (Acc_x) est facile. Pour les prédicats mentionnés ci-dessus, Elgot et Rabin montrèrent que pour tout automate de Büchi $\mathcal{A}$, les blocs de 0 consécutifs de x_P peuvent être contractés de manière à obtenir un mot ultimement périodique x' tel que $\mathcal{A}$ accepte x_P si et seulement si $\mathcal{A}$ accepte x'. Le problème de décider si $\mathcal{A}$ accepte x' se résout facilement.

Récemment, Maes [53] étudia des *prédicats morphiques* qui peuvent être obtenus par application itérée d'un morphisme sur les mots. Il a prouvé que pour un prédicat morphique P, la théorie du *premier ordre* de la structure $\langle \mathbb{N}, <, P \rangle$ est décidable. Les prédicats introduits par Maes sont des prédicats d'arité quelconque. Ils sont obtenus par itération de morphismes de mots à plusieurs dimensions. De manière technique, ces morphismes doivent vérifier une propriété de symétrie qui assure la compatibilité des blocs lors de

l'itération dans le cas d'une dimension strictement supérieure à 1.

Le résultat principal que nous avons obtenu est qu'il est possible d'étendre le résultat de Maes en passant du premier ordre au second ordre monadique. Il faut cependant se restreindre aux prédicats unaires. En effet, l'addition est un prédicat morphique et il est bien connu que la théorie du second ordre monadique de la structure $\langle \mathbb{N}, + \rangle$ est indécidable.

Nous donnons maintenant la définition précise d'un prédicat morphique. Puisque dans la suite, nous utilisons uniquement des prédicats unaires, nous nous contentons de la définition dans ce cas qui a l'avantage d'être plus simple. Rappelons qu'un *morphisme* τ de A^* dans lui-même est une application telle que l'image d'un mot $u = a_1 \dots a_n$ est égale à la concaténation $\tau(a_1) \dots \tau(a_n)$ des images de ses lettres. Le morphisme est donc complètement déterminé par les images des lettres. On définit la suite $(x_n)_{n \geq 0}$ de mots finis par $x_n = \tau^n(a)$ où a est une lettre fixée. Si l'image $\tau(a)$ de a commence par a, chaque mot x_n est un préfixe du mot suivant x_{n+1}. Si, de plus, la suite des longueurs $|x_n|$ n'est pas bornée, la suite $(x_n)_{n \geq 0}$ converge vers un mot infini que nous notons $\tau^\omega(a)$. Un mot infini x sur un alphabet B est dit *morphique* s'il existe un morphisme τ de A^* dans lui-même et un autre morphisme σ de A^* dans B^* tel que $x = \sigma(\tau^\omega(a))$. Par extension, un prédicat P est aussi dit *morphique* si son mot caractéristique x_P est morphique. Nous pouvons maintenant énoncer notre résultat [32].

Théorème 6 *Soit $x = \sigma(\tau^\omega(a))$ un mot morphique où τ et σ sont des morphismes. On peut décider si un automate de Büchi donné accepte le mot x.*

Grâce au résultat de Büchi [22], le théorème a pour conséquence immédiate le résultat suivant.

Corollaire 7 *Pour tout prédicat morphique, la théorie du second ordre monadique de la structure $\langle \mathbb{N}, <, P \rangle$ est décidable.*

Ce dernier résultat est intéressant car la classe des prédicats morphiques est relativement riche. Elle contient nombre de prédicats étudiés auparavant. Presque tous les prédicats considérés par Elgot et Rabin [43] font en particulier partie de cette classe. Notre résultat fournit une approche unifiée aux résultats précédents de Elgot et Rabin ainsi que ceux de Maes tout en les généralisant. La preuve que nous donnons a également l'avantage d'être courte tout en restant élémentaire.

Une généralisation du théorème précédent serait d'obtenir le même résultat en remplaçant le point fixe du morphisme par le point fixe d'une fonction séquentielle. Si la transition qui est étiquetée par une lettre a et qui part de l'état initial d'un transducteur a une sortie qui commence par a, les images

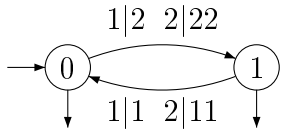


FIG. 24 – Fonction séquentielle de point fixe la suite de Kolakovski.

successives $\sigma(a), \sigma^2(a), \sigma^3(a), \ldots$ de a par la fonction séquentielle σ réalisée par le transducteur sont préfixes les unes des autres. Si de plus la suite des longueurs de ces mots n'est pas bornée, la suite de ces mots converge vers un mot infini qui est point fixe de la fonction séquentielle. Cette généralisation semble toutefois difficile car certaines suites compliquées sont point fixe d'une fonction séquentielle. La suite de Kolakovski est en effet point fixe de la fonction séquentielle représentée à la figure 24.

Pour montrer que la classe des prédicats morphiques contient des prédicats intéressants, nous avons établi le résultat suivant qui s'applique à beaucoup de prédicats. Les preuves que des prédicats donnés sont effectivement morphiques sont souvent excessivement techniques. La proposition suivante permet de simplifier les arguments dans un grand nombre de cas.

Proposition 8 *Soit $(u_n)_{n\geq 0}$ une suite d'entiers positifs ou nuls et soit $d_n = u_{n+1} - u_n$ la suite des différences. S'il existe un entier ℓ tel que $d_n \geq 1$ pour $n \geq \ell$ et tel que la suite $(d_n)_{n\geq \ell}$ soit $\mathbb{N}$-rationnelle, alors le prédicat $P = \{u_n \mid n \geq 0\}$ est morphique.*

Le corollaire suivant montre que la plupart des prédicats considérés par Elgot et Rabin sont morphiques.

Corollaire 9 *Soit Q un polynôme de coefficient dominant entier et positif tel que $Q(n)$ soit entier pour tout entier n et soit k un entier positif. Le prédicat $P = \{Q(n)k^n \mid n \geq 0 \text{ et } Q(n) \geq 0\}$ est morphique.*

En prenant Q égal à n^ℓ pour un entier ℓ et k égal à 1, on retrouve le prédicat $\{n^\ell \mid n \geq 0\}$ et en prenant Q égal à 1, on retrouve le prédicat $\{k^n \mid n \geq 0\}$. Le seul prédicat considéré par Elgot et Rabin qui échappe au corollaire précédent est le prédicat factoriel $\{n! \mid n \geq 0\}$. Par un argument de croissance, on peut facilement montrer que ce prédicat n'est pas morphique.

Afin de prendre en compte le prédicat factoriel, nous avons généralisé la méthode utilisée pour les prédicats morphiques. Nous avons introduit une classe plus large de prédicats pour lesquels la théorie du second ordre monadique de la structure $\langle \mathbb{N}, <, P\rangle$ reste décidable. Cette généralisation est basée sur la notion de suite profiniment ultimement périodique que nous définissons maintenant.

Définition 10 *Une suite $(u_n)_{n\geq 0}$ de mots sur un alphabet A est dite* profiniment ultimement périodique *si pour tout morphisme μ de A^+ dans un semigroupe fini S, la suite $\mu(u_n)$ est ultimement périodique.*

Les suites de mots qui sont profiniment ultimement constantes ont beaucoup été étudiées. Elles sont appelées *opérations implicites* dans la littérature [2]. L'exemple canonique de telles suites est la suite $u_n = a^{n!}$.

Un mot infini x sur l'alphabet $\{0,1\}$ se factorise de manière unique $x = u_0u_1u_2\ldots$ en une suite de mots u_n appartenant à 0^*1. Le mot x est alors dit profiniment ultimement périodique si la suite $(u_n)_{n\geq 0}$ des mots de la factorisation est profiniment ultimement périodique. Par extension, un prédicat est aussi dit profiniment ultimement périodique si son mot caractéristique est profiniment ultimement périodique. La suite des mots $\tau^n(a)$ obtenus en itérant un morphisme τ est profiniment ultimement périodique. On retrouve ainsi les prédicats morphiques dans la classe $\mathcal{K}$ des prédicats profiniment ultimement périodiques. De plus, cette classe $\mathcal{K}$ jouit de nombreuses propriétés de clôture (somme, produit, etc.) qui la rend particulièrement riche. Elle contient en particulier le prédicat factoriel. Nous avons montré en outre que tout prédicat P de $\mathcal{K}$ conduit à une théorie monadique de la structure $\langle \mathbb{N}, <, P\rangle$ qui est décidable [32]. La preuve de ce résultat est en fait une extension directe de celle pour les prédicats morphiques.

3.4 Produit en couronne

Dans cette partie, nous nous intéressons aux liens entre les fonctions séquentielles d'une part et le produit en couronne d'autre part. Pour les mots finis, ces liens sont bien connus et ils sont appelés *principe du produit en couronne* dans la littérature [79, 66]. Nous allons étendre ce principe aux mots infinis puis l'utiliser pour obtenir des résultats de décomposition pour les variétés de mots infinis.

Les fonctions séquentielles sont les fonctions réalisées par les transducteurs déterministes en entrée. Elles peuvent être vues comme une généralisation simple des morphismes. L'image par un morphisme τ d'un mot $u = a_1\ldots a_n$ est le mot $\tau(a_1)\ldots\tau(a_n)$ obtenu en substituant chaque lettre a_i par $\tau(a_i)$. Dans le cas d'un morphisme, ce par quoi est substituée une lettre ne dépend que de cette lettre. Dans le cas d'une fonction séquentielle (gauche), ceci dépend de la lettre a_i mais aussi du préfixe $a_1\ldots a_{i-1}$ du mot. En réalité, ceci dépend de manière très simple de ce préfixe puisque c'est entièrement déterminé par l'état d'arrivée du chemin initial étiqueté par ce préfixe dans le transducteur et par la lettre a_i. La lettre a_i est en effet remplacée par la sortie de la transition étiquetée en entrée par a_i et issue de l'état atteint par

le préfixe $a_1 \dots a_{i-1}$ à partir de l'état initial.

Le théorème d'Eilenberg établit qu'il y a une correspondance bijective entre les (pseudo-)variétés de semigroupes finis et les variétés de langages reconnaissables. Une question naturelle est de connaître pour chaque opération sur les langages l'opération correspondante au niveau des semigroupes. Soit une opération $L \mapsto \hat{L}$ qui associe un langage reconnaissable $\hat{L}$ à chaque langage reconnaissable L de mots finis. L'opération correspondante se décrit au niveau des variétés de semigroupes. Elle associe à chaque variété $\mathbf{V}$ de semigroupes finis une variété $\hat{\mathbf{V}}$ qui est engendrée par les semigroupes syntaxiques des langages $\hat{L}$ pour tous les langages L dont le semigroupe syntaxique appartient à $\mathbf{V}$.

Il s'avère que nombre d'opérations naturelles sur les langages peuvent être décrites en utilisant des images inverses de fonctions séquentielles [68]. Lorsque c'est le cas, le principe du produit en couronne permet de décrire l'opération $\mathbf{V} \mapsto \hat{\mathbf{V}}$.

Si le langage L de mots finis est reconnu par le semigroupe S et si σ est une fonction séquentielle, alors l'image inverse $\sigma^{-1}(L)$ de L par σ est reconnue par le produit en couronne $S \circ S(\sigma)$ de S et du semigroupe syntaxique $S(\sigma)$ de σ. Rappelons que toute fonction séquentielle est réalisée par un transducteur minimal [33]. Par définition, le *semigroupe syntaxique* de σ est le semigroupe de transitions de ce transducteur minimal. Le monoïde $S(\sigma)$ est en quelque sorte minimal également puisqu'il divise le semigroupe de transitions de tout autre transducteur séquentiel qui réalise σ.

Inversement, le principe du produit en couronne décrit les langages qui sont reconnus par un produit en couronne $S \circ T$ de deux semigroupes S et T. Il établit que tout langage reconnu par $S \circ T$ est égal à une combinaison booléenne de langages reconnus par S et de langages de la forme $\sigma^{-1}(L)$ où σ est une fonction séquentielle et où L est reconnu par T. Les fonctions séquentielles σ qui apparaissent dans cet énoncé correspondent aux différents morphismes de A^+ dans T et décrivent d'une certaine manière le produit interne de T.

Nous avons étendu ces deux résultats aux mot infinis [29]. La première étape est de définir le produit en couronne pour les ω-semigroupes. En effet, la bonne structure algébrique pour reconnaître les ensembles de mots infinis est la structure de ω-semigroupe qui est un semigroupe muni en plus d'un produit infini [87, 88, 65]. La définition de ce produit en couronne passe par la définition préalable du produit semi-direct. L'extension aux mots infinis du principe du produit en couronne se fait alors sans difficulté majeure.

Nous avons ensuite utilisé ces résultats pour obtenir des résultats de décomposition de variétés de langages de mots infinis. L'énoncé de ces résultats nécessite quelques précisions sur les ω-semigroupes. Rappelons qu'un

ω-semigroupe est une paire (S_+, S_ω) où S_+ est un semigroupe muni de deux opérations supplémentaires. La première opération est une action à gauche de S_+ sur S_ω. La seconde est un produit infini qui associe un élément de S_ω à toute suite de longueur ω d'éléments de S_+. Ce produit infini vérifie une propriété d'associativité et il est compatible avec l'action à gauche.

Un ω-semigroupe (S_+, S_ω) définit naturellement deux semigroupes. Le premier est bien entendu le semigroupe S_+. Le second est le quotient de S_+ par la congruence $\sim$ définie de la façon suivante. Deux éléments s et s' de S_+ vérifient $s \sim s'$ si et seulement si $su = s'u$ pour tout u de S_ω. Chaque classe de cette congruence est constituée des éléments ayant même action à gauche sur S_ω. Pour cette raison, le semigroupe quotient $S_+/\sim$ est noté $\mathrm{act}(S)$. Pour chaque variété $\mathbf{V}$ de semigroupes, on peut définir deux variétés $\mathbf{V}_+$ et $\mathbf{V}_\omega$ de ω-semigroupes. La variété $\mathbf{V}_+$ est la variété des ω-semigroupes (S_+, S_ω) tels que S_+ appartient à $\mathbf{V}$. La variété $\mathbf{V}_\omega$ est la variété des ω-semigroupes S tels que $\mathrm{act}(S)$ appartient à $\mathbf{V}$. Puisque $\mathrm{act}(S)$ est un quotient de S_+, on a l'inclusion $\mathbf{V}_+ \subseteq \mathbf{V}_\omega$.

On note $\mathbf{I}$ la variété de semigroupes qui ne contient que le semigroupe trivial à un seul élément. Nous avons montré le résultat suivant qui relie les deux variétés $\mathbf{V}_+$ et $\mathbf{V}_\omega$ [29].

Théorème 11 *Pour toute variété* $\mathbf{V}$ *de semigroupes, on a l'égalité*

$$\mathbf{V}_\omega = \mathbf{I}_\omega \circ \mathbf{V}_+.$$

Par définition, un ω-semigroupe appartient à $\mathbf{I}_\omega$ si son action est triviale. Un ensemble X de mots infinis est reconnu par un tel ω-semigroupe si pour toute lettre a et tout mot infini x, ax appartient à X si et seulement si x appartient à X. Ceci signifie de manière intuitive que l'appartenance d'un mot x à X ne dépend pas des préfixes de x, mais uniquement de sa partie ultime. Un exemple typique d'un tel ensemble X est l'ensemble $(A^*a)^\omega$ des mots ayant un nombre infini de a.

En vertu du principe du produit en couronne, l'énoncé du théorème précédent s'interprète de la façon suivante. Un ensemble a son action dans $\mathbf{V}$ s'il est l'image inverse d'un ensemble d'action triviale par une fonction séquentielle dont le semigroupes syntaxique appartient à la variété $\mathbf{V}$.

Dans le cas de variétés apériodiques, le résultat du théorème précédent peut être amélioré. Il est en effet possible de montrer le résultat suivant [29].

Théorème 12 *Pour toutes variétés* $\mathbf{V}$ *et* $\mathbf{W}$ *de semigroupes telles que* $\mathbf{V}$ *soit apériodique, c'est-à-dire vérifiant l'inclusion* $\mathbf{V} \subseteq \mathbf{A}$*, on a l'égalité*

$$\mathbf{V}_\omega \cap \mathbf{W}_+ = (\mathbf{I}_\omega \cap \mathbf{W}_+) \vee (\mathbf{V}_+ \cap \mathbf{W}_+).$$

En prenant la variété $\mathbf{W}$ égale à la variété $\mathbf{S}$ de tous les semigroupes finis, on obtient le corollaire suivant qui précise le résultat du théorème précédent dans le cas de variétés apériodiques.

Corollaire 13 *Pour toute variété apériodique* $\mathbf{V}$ *de semigroupes, on a l'égalité*

$$\mathbf{V}_\omega = \mathbf{I}_\omega \vee \mathbf{V}_+.$$

Ce dernier résultat signifie de manière intuitive qu'un ensemble X dont l'action est dans $\mathbf{V}$ peut s'écrire comme une combinaison booléenne d'ensembles d'action triviale et d'ensembles reconnus par la variété $\mathbf{V}$. L'hypothèse que $\mathbf{V}$ soit apériodique est indispensable. Il est en effet possible de montrer que pour la variété $\mathbf{G}$ de tous les groupes finis, les inclusions $\mathbf{G}_+ \subsetneq \mathbf{I}_\omega \subsetneq \mathbf{G}_\omega$ sont strictes.

4 Transducteurs et dynamique symbolique

La dynamique symbolique étudie des systèmes dynamiques qui sont décrits par des ensembles de suites bi-infinies de symboles pris dans un alphabet. La classification de ces systèmes passe par l'étude des transformations entre ces systèmes. Ces transformations entre systèmes permettent de modéliser les problèmes de codage et ont ainsi de nombreuses applications très pratiques. Parmi ces applications, on peut mentionner le codage par canaux contraints [4]. Lorsque des données sont écrites sur un disque dur, des contraintes d'ordre physique imposent qu'un nombre limité de 0 ou de 1 consécutifs puissent être écrits. Il faut donc encoder une suite quelconque de 0 et de 1 en une suite qui vérifie ces contraintes. Ceci peut être réalisé par un transducteur.

Les transformations entre systèmes dynamiques symboliques sont souvent réalisées par des transducteurs finis. La dynamique symbolique se trouve ainsi intimement liée à l'étude des transducteurs. Cette partie regroupe des résultats sur la dynamique symbolique et les transducteurs. Le premier résultat concerne la fonction zêta qui est un invariant des systèmes. La synchronisation des transducteurs lorsqu'ils réalisent des transformations entre systèmes est l'objet du second résultat. Le dernier est la déterminisation de transducteurs réalisant des relations de mots infinis.

4.1 Fonction Zêta et langages cycliques

Les résultats présentés dans cette partie ont été obtenus en collaboration avec Marie-Pierre Béal et Christophe Reutenauer.

De manière générale, un système dynamique est un espace topologique muni d'une transformation. En dynamique symbolique, un système est toujours un sous-système d'un système $A^{\mathbb{Z}}$ des mots bi-infinis sur un alphabet A fixé. La topologie de $A^{\mathbb{Z}}$ est la topologie produit qui peut aussi être définie par une métrique. La transformation canonique de $A^{\mathbb{Z}}$ est le *décalage* (*shift* en anglais) σ qui envoie une suite $(a_n)_{n\in\mathbb{Z}}$ sur la suite décalée $(a_{n+1})_{n\in\mathbb{Z}}$. Un système est alors un ensemble de mots bi-infinis qui est clos pour le décalage et fermé pour la topologie. Un exemple important de système est l'ensemble des mots qui sont l'étiquette d'un chemin bi-infini dans un automate fini. Un tel système est appelé *sofique*. Un autre exemple est l'ensemble des mots bi-infinis où n'apparaissent pas comme facteur les mots d'un ensemble fini et fixé. Un tel système est dit de *type fini*. Un système de type fini est un cas particulier de système sofique puisqu'on peut prendre l'automate de De Bruijn pour une longueur de mots assez grande. Un système de type fini correspond au cas où l'automate peut être choisi local.

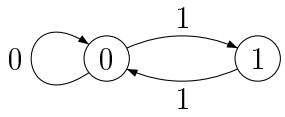


FIG. 25 – Un système sofique.

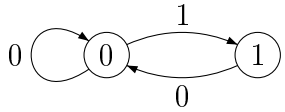


FIG. 26 – Un système de type fini.

Une application *locale* entre deux systèmes est une application continue qui commute avec le décalage. La terminologie *locale* provient du fait que l'image d'un mot par une telle application peut être calculée en faisant glisser une fenêtre le long du mot. La lettre de l'image à la position i est entièrement déterminée par les lettres du mot $a_{i-m}\dots a_{i+k}$ pour des entiers k et m fixés. La taille de la fenêtre est alors $m+k+1$. Pour cette raison, une application locale est aussi appelée application *à fenêtre glissante*. Une application locale peut être réalisée par un transducteur dont l'automate des entrées est un automate de De Bruijn.

Deux systèmes sont dits *conjugués* s'il existe une application locale bijective entre les deux systèmes. On vérifie que la réciproque est aussi locale même

si la taille minimale de sa fenêtre peut être beaucoup plus grande. La décidabilité de la conjugaison pour les systèmes sofiques est encore actuellement un problème ouvert. Un certain nombre d'invariants répondent partiellement à cette question en donnant des conditions nécessaires à la conjugaison. Le principal invariant est l'*entropie* qui mesure la croissance du nombre de facteurs dans le système. Un autre invariant peut être obtenu en considérant les facteurs interdits minimaux à la place des facteurs qui apparaissent [13]. La fonction *zêta* d'un système est un invariant plus fin que l'entropie. Elle compte le nombre d'orbites périodiques du système. Formellement, elle est définie par $\zeta(S) = \exp \sum_{n\geq 1} a_n \frac{z^n}{n}$ où a_n est le nombre de mots bi-infinis de période n (dont la période divise n) du système. Par exemple, la fonction zêta du système sofique de la figure 25 est $(1+z)/(1-z-z^2)$ alors que celle du système de type fini de la figure 26 est $1/(1-z-z^2)$.

La $\mathbb{Z}$-rationalité de la fonction zêta d'un système de type fini a d'abord été prouvée par Bowen et Lanford [19] puis a été étendue aux systèmes sofiques par Manning [54]. Bowen [18] redonne une autre expression rationnelle de la fonction zêta d'un système sofique qui conduit à un meilleur algorithme de calcul de celle-ci. La validité de cette expression n'est pas établie dans l'article de Bowen, mais une preuve de celle-ci peut être trouvée en [5]. La $\mathbb{N}$-rationalité de la fonction zêta d'un système sofique a été finalement prouvée par Reutenauer [72], mais cette preuve est très compliquée et une autre preuve plus simple reste encore à trouver.

Berstel et Reutenauer ont introduit en [17] la notion de fonction *zêta généralisée* d'un langage de mots finis. Pour un langage L de mots finis, celle-ci est définie par

$$\zeta(L) = \exp \sum_{n\geq 1} \frac{\varphi(L \cap A^n)}{n}$$

où $\varphi(L \cap A^n)$ est l'image commutative des mots de longueur n de L. On retrouve la fonction zêta classique en projetant toutes les lettres de l'alphabet sur la variable z. Il est montré en [17] que la fonction zêta généralisée d'un langage cyclique est $\mathbb{Z}$-rationnelle. Nous avons redonné en [12] une nouvelle preuve de ce résultat en décomposant un langage cyclique en langages fortement cycliques. Notre preuve s'appuie alors sur la preuve de la $\mathbb{Z}$-rationalité de la fonction zêta d'un langage fortement cyclique donnée en [5]. Cette preuve fournit un nouvel algorithme de calcul pour cette fonction zêta.

Un ensemble L de mots finis sur un alphabet A est dit *cyclique* s'il vérifie

$$\begin{array}{ll} \forall u \in A^* \ \ \forall n > 0 & u \in L \iff u^n \in L \\ \forall u, v \in A^* & uv \in L \iff vu \in L. \end{array}$$

La première propriété signifie qu'un mot appartient à un langage cyclique L si et seulement si sa racine primitive appartient à L. La seconde signifie qu'un langage cyclique est clos par conjugaison. Un langage cyclique est donc entièrement déterminé par les mots de Lyndon qu'il contient. L'ensemble $L = A^*aA^*$ des mots ayant au moins un a est cyclique. L'ensemble

$$\begin{aligned} L =&\{a^p b^{n_1} a^{n_2} b^{n_2} \dots a^{n_k} b^{n_k} a^q \mid n_i \geq 0 \text{ et } p+q = n_1\} \cup \\ &\{b^p a^{n_1} b^{n_1} a^{n_2} b^{n_2} \dots a^{n_k} b^q \mid n_i \geq 0 \text{ et } p+q = n_k\} \end{aligned}$$

est aussi cyclique mais il n'est pas rationnel.

Soit $\mathcal{A} = (Q, A, E)$ un automate déterministe sans états initiaux et sans états finaux. Pour un état q et un mot fini w, on note $q \cdot w$, l'unique état p, s'il existe, tel qu'il y ait un chemin de q à p étiqueté par w. On dit qu'un mot fini w *stabilise* un sous-ensemble $P \subseteq Q$ d'états si $P \cdot w = P$, ce qui signifie

$$\begin{aligned} &\forall p \in P && p \cdot w \in P \\ &\forall p' \in P \; \exists p \in P && p \cdot w = p'. \end{aligned}$$

On note $\mathrm{stab}(\mathcal{A})$ l'ensemble des mots finis qui stabilisent au moins un sous-ensemble non vide de Q. Un ensemble de mots finis est dit *fortement cyclique* s'il est égal à $\mathrm{stab}(\mathcal{A})$ pour un automate déterministe $\mathcal{A}$. Les langages fortement cycliques associés aux automates des figures 25 et 26 sont respectivement $a^* + (aa + b)^* + a(aa + b)^*a$ et $(b + ab)^*(\varepsilon + a)$. Les fonctions zêta généralisées de ces langages sont respectivement $(1 + a)/(1 - b - a^2)$ et $1/(1 - b - ab)$. Il faut remarquer que le mot vide stabilise n'importe quel sous-ensemble d'états et qu'il appartient par conséquent à tout langage fortement cyclique. La terminologie est justifiée par le fait qu'on vérifie sans difficulté qu'un langage fortement cyclique est effectivement cyclique. La remarque précédente montre que le langage cyclique $L = A^*aA^*$ n'est pas fortement cyclique puisqu'il ne contient pas le mot vide. Par contre ce langage s'écrit comme la différence $A^* - b^*$ et les langages A^* et b^* sont fortement cycliques. Le résultat que nous avons montré établit que le cas général se comporte comme cet exemple. Tout langage cyclique peut être décomposé à l'aide d'une chaîne de langages fortement cycliques.

Théorème 14 *Soit L un langage rationnel cyclique. Il existe alors une suite décroissante $L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n$ de langages fortement cycliques rationnels tels que $L = L_1 - L_2 + \cdots \pm L_n$.*

La preuve de ce résultat s'obtient en travaillant sur les monoïdes finis. Une première étape consiste à obtenir des caractérisations en terme de monoïdes syntaxiques des langages cycliques et fortement cycliques. Ensuite, le langage

fortement cyclique L_1 est choisi à partir du monoïde syntaxique de L de telle sorte que $L - L_1$ soit reconnu par un monoïde strictement plus petit. La preuve se termine alors par une récurrence sur la taille du monoïde syntaxique de L.

Une question naturelle est se savoir pour un langage cyclique L donné, quelle est la longueur minimale d'une chaîne de langages fortement cycliques permettant d'exprimer L. Cette question a été résolue dans l'article [28]. J'ai effectivement montré que la hiérarchie induite par cette longueur minimale de chaîne correspondait à des suites d'idempotents décroissantes pour l'ordre $\mathcal{H}$ dans le monoïde syntaxique et qu'elle était donc décidable.

Le théorème précédent permet de prouver que la fonction zêta généralisée d'un langage cyclique est $\mathbb{Z}$-rationnelle. En effet, la fonction zêta généralisée d'un langage L égal à $L_1 - L_2 + \cdots \pm L_n$ est donnée par la formule

$$\zeta(L) = \prod_{i=1}^{n} \zeta(L_i)^{(-1)^i}.$$

D'après le théorème précédent, la $\mathbb{Z}$-rationalité de la fonction zêta d'un langage cyclique se ramène à la $\mathbb{Z}$-rationalité de la fonction zêta d'un langage fortement cyclique qui a été prouvée en [5]. Il faut remarquer qu'un mot fini w non vide stabilise un sous-ensemble d'états d'un automate $\mathcal{A}$ si et seulement si le mot bi-infini périodique ${}^{\omega}w^{\omega}$ est l'étiquette d'un chemin dans $\mathcal{A}$. La fonction zêta de $\mathrm{stab}(\mathcal{A})$ est donc égale à la fonction zêta du système sofique défini par $\mathcal{A}$.

4.2 Synchronisation de transducteurs

Ce travail a été réalisé en collaboration avec Marie-Pierre Béal.

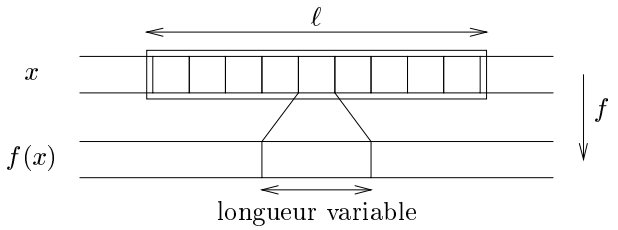


FIG. 27 – Une application à fenêtre glissante.

Une application locale de $A^{\mathbb{Z}}$ dans $B^{\mathbb{Z}}$ est une application continue qui commute avec le décalage. L'image d'un mot par une telle application est obtenue en faisant glisser une fenêtre d'une longueur fixée le long du mot. Pour

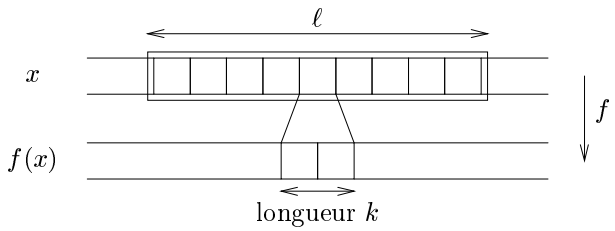


FIG. 28 – Une application à fenêtre glissante synchrone.

chaque position de la fenêtre, le contenu de cette fenêtre détermine un mot fini. L'image globale est obtenue en concaténant tous ces mots finis donnés par les différentes positions de la fenêtre. Si la longueur de la fenêtre est ℓ, l'application locale f est définie par une fonction $\bar{f}$ de A^ℓ dans B^k qui associe un mot à chaque contenu de fenêtre. Pour les applications locales habituellement considérées en dynamique symbolique, la taille de ce mot est fixe. Pour cette raison, ces applications locales sont dites *synchrones*. Nous avons étendu cette définition en autorisant cette longueur à être variable. La fonction $\bar{f}$ devient alors une fonction de A^ℓ dans B^+ et l'application locale associée est dite *asynchrone*. Nous avons étudié le problème de la synchronisation de ces applications locales. Il s'agit de trouver des conditions naturelles qui garantissent qu'une application locale définie par une application $\bar{f}$ de A^ℓ dans B^+ peut aussi être définie par une fonction $\bar{f}'$ de $A^{\ell'}$ dans B^k qui la rend synchrone. Ces conditions apparaissent naturellement lorsque l'application est réalisée par un transducteur.

Un transducteur est un automate dont les étiquettes des transitions sont des paires de mot et non pas des lettres comme pour les automates usuels. En dynamique, on considère des chemin bi-infinis et les transducteurs n'ont ni états initiaux ni états finaux. Un transducteur est juste un graphe étiqueté par des paires de mots. Le taux de transmission d'une transition étiquetée par une paire (u, v) est le rapport $|v|/|u|$ entre les longueurs de v et de u. Le transducteur est dit *synchrone* si ce taux de transmission est constant pour toutes les transitions. Dans ce cas, il est appelé taux de transmission du transducteur.

Pour chaque application locale, on peut trouver un transducteur qui la réalise. Si de plus l'application est synchrone, le transducteur peut être aussi choisi synchrone. Supposons que l'application locale f soit définie par une fonction $\bar{f}$ de A^ℓ dans B^+ et choisissons un entier n compris entre 1 et ℓ. L'ensemble des états du transducteur est l'ensemble $A^{\ell-1}$ des mots de lon-

gueur $\ell - 1$ et les transitions ont la forme

$$a_1 \dots a_{\ell-1} \xrightarrow{a_n | \bar{f}(a_1 \dots a_\ell)} a_2 \dots a_\ell.$$

Ce transducteur est co-déterministe si $n = 1$ et déterministe si $n = \ell$. Si la fonction $\bar{f}$ est de A^ℓ dans B^k, le transducteur ainsi défini est synchrone et son taux de transmission est k. Il faut remarquer que chaque transition détermine une des lettres qui forment les mots identifiant les états. Il s'ensuit que tout chemin de longueur $\ell - 1$ dans ce transducteur détermine complètement un des états. Dans un chemin

$$q_1 \xrightarrow{a_1|u_1} q_2 \xrightarrow{a_2|u_2} \cdots \xrightarrow{a_{\ell-2}|u_{\ell-2}} q_{\ell-1} \xrightarrow{a_{\ell-1}|u_{\ell-1}} q_\ell$$

l'état q_n est nécessairement égal à $a_1 \dots a_{\ell-1}$. Un transducteur vérifiant cette propriété est dit $(n-1, \ell-n)$-*local* ou simplement local. On vérifie qu'un automate est local si et seulement si chaque mot infini est l'étiquette d'un seul chemin. Ceci garantit en particulier que la relation réalisée par le transducteur est en fait une fonction. La terminologie est justifiée par le fait que tout application réalisée par un transducteur local est également locale. Synchroniser une application locale revient alors à synchroniser un transducteur, c'est-à-dire à trouver un transducteur synchrone réalisant la même relation.

Le problème de la synchronisation est apparu dès l'introduction des relations rationnelles et des transducteurs par Elgot et Mezei [42] avec le résultat d'Eilenberg et Schützenberger [39]. Ce dernier établit qu'une relation rationnelle de $A^* \times B^*$ qui préserve la longueur est en fait une relation rationnelle de $(A \times B)^*$. En terme de transducteurs, ce résultat signifie qu'une relation rationnelle qui préserve la longueur peut être réalisée par un transducteur synchrone de taux de transmission égal à 1. La preuve originale de ce théorème se faisait directement sur les expressions rationnelle. En [45], Frougny et Sakarovitch ont étendu ce résultat aux relations rationnelles à différence de longueurs bornée. Leur preuve opère sur les transducteurs et fonctionne aussi bien pour les relations de mots finis que pour les relations de mots infinis.

Nous considérons ici des relations de mots bi-infinis. Un transducteur est synchronisable s'il a un taux de transmission constant sur tous ses cycles. Nous avons montré en [8] que l'algorithme de [45] peut être réalisé en utilisant des éclatements d'états. Il est d'ailleurs possible d'utiliser uniquement des éclatements entrants ou uniquement des éclatements sortants. Ceci permet de préserver éventuellement le déterminisme ou le co-déterminisme du transducteur. L'avantage de cette approche est que l'éclatement d'états est un isomorphisme entre les systèmes dynamiques. Il préserve en particulier la

propriété d'être local du transducteur, ce qui est essentiel dans notre cas. Un éclatement d'état consiste à remplacer un état q par plusieurs états $q_1, \ldots, q_n$. Si l'on réalise un éclatement entrant, les transitions entrantes de q sont réparties entres $q_1, \ldots, q_n$ alors que les transitions sortantes sont dupliquées pour chacun des états $q_1, \ldots, q_n$. L'algorithme de synchronisation est implanté par une succession d'éclatements d'états et d'étapes où certaines lettres de sortie migrent pour équilibrer les taux de transmission. Les éclatements d'états sont en fait réalisés de façon à rendre possible la migration des lettres nécessaire à l'équilibrage.

La complexité de l'algorithme ainsi implanté est exponentielle en le nombre d'états. Nous avons montré que cette explosion du nombre d'états est intrinsèque au problème. Nous avons donné en [8] des exemples de transducteurs pour lesquels tout transducteur équivalent qui est synchrone et local a un nombre exponentiel d'états. Par contre, la taille de la fenêtre qui est un paramètre important en dynamique symbolique croît seulement de manière linéaire. Ceci peut être obtenu en réalisant simultanément plusieurs éclatements d'états.

4.3 Déterminisation de transducteurs

Les résultats présentés dans cette partie ont été obtenus en collaboration avec Marie-Pierre Béal. Dans ce travail, nous nous sommes intéressés aux relations de mot infinis réalisées par des transducteurs.

Un transducteur est un automate dont les transitions sont étiquetées par des paires de mots. La première composante est l'entrée et la seconde est la sortie. Les transducteurs que nous considérons ont des états initiaux ainsi qu'une condition d'acceptation adaptée aux mots infinis de type Muller ou Büchi. La relation réalisée est l'ensemble des paires de mots infinis qui sont acceptées par le transducteur. Nous supposons toujours que les relations réalisées par les transducteurs sont des fonctions. Il s'agit d'une propriété décidable [48].

Parmi les transducteurs, les transducteurs qui sont déterministes en entrée jouent un rôle particulier. Ils permettent un calcul séquentiel de la sortie à partir de l'entrée et sont appelés *séquentiels* pour cette raison. La déterminisation d'un transducteur est la construction d'un transducteur séquentiel qui réalise la même fonction. Cette construction n'est pas toujours possible en général. Les fonctions qui peuvent être réalisées par un transducteur séquentiel sont elles-mêmes appelées *séquentielles*.

Pour les mots finis, Schützenberger [76] a introduit une variante des transducteurs séquentiels qui sont appelés *sous-séquentiels*. Un tel transducteur est autorisé à ajouter à la sortie un mot déterminé par l'état final lorsque celui-ci

a été atteint. Les fonctions réalisées par ces transducteurs sont en définitive plus naturelles que celles réalisées par les transducteurs purement séquentiels, mais cette distinction n'est pas significative pour les mots infinis.

Pour les mots finis, la caractérisation des fonctions séquentielles et sous-séquentielles est due à Choffrut [34, 35]. Celle-ci comporte d'une part une caractérisation intrinsèque de ces fonctions en terme de distances sur les mots et d'autre part une caractérisation des transducteurs réalisant une fonction séquentielle. Cette caractérisation des transducteurs est basée sur une propriété de jumelage des chemins. Lorsque la fonction est séquentielle, Choffrut a donné une façon de construire un transducteur séquentiel équivalent. Cette construction est une généralisation de la déterminisation d'un automate par la méthode des sous-ensembles [1]. Au lieu d'être simplement des ensembles d'états, les états du transducteur séquentiel sont des ensembles de paires formées d'un état du transducteur initial et d'un mot fini. C'est la propriété de jumelage qui garantit que la taille des mots apparaissant dans ces paires est bornée et que le nombre d'états est fini.

La décidabilité de la sous-séquentialité et de la séquentialité découle de la caractérisation donnée par Choffrut [34, 35]. Des algorithmes polynomiaux ont ensuite été donnés par Weber et Klemm [86]. Nous avons également fourni un autre algorithme polynomial en [11, 10].

La déterminisation de transducteurs inclut la déterminisation d'automates puisqu'un transducteur séquentiel fournit immédiatement un automate déterministe pour le domaine de la fonction. Pour les mots infinis, cette déterminisation d'automates est beaucoup plus délicate et nécessite des constructions assez élaborées comme celle due à Safra [74]. Cette difficulté se retrouve bien sûr au niveau des transducteurs.

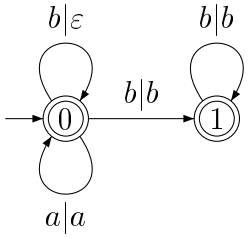


FIG. 29 – Un transducteur.

Pour contourner cette difficulté, nous nous sommes d'abord intéressés aux transducteurs dont tous les états sont finaux. La présence de boucles dont la sortie est vide rend ces transducteurs plus compliqués qu'on ne pourrait le penser. La fonction réalisée par le transducteur de la figure 29 n'est pas

continue. En effet, l'image d'un mot x est égale à a^ω si x a une infinité d'occurrences de a et elle est égale à $a^n b^\omega$ si x possède n occurrences de a. En [9, 7], nous avons caractérisé parmi ces transducteurs ceux qui réalisent une fonction séquentielle et nous avons donné un algorithme de déterminisation. Nous avons en particulier montré que si une fonction séquentielle est réalisée par un transducteur dont tous les états sont finaux, elle peut être réalisée par un transducteur séquentiel dont tous les états sont aussi finaux. Nous avons ensuite résolu le cas général dans [6] où nous considérons des fonctions réalisées par des transducteurs avec une condition de Büchi. Nous avons caractérisé les transducteurs réalisant des fonctions qui sont Büchi séquentielles et Muller séquentielles. Puisque les automates de Büchi déterministes ne reconnaissent pas tous les ensembles rationnels de mots infinis, nous avons en effet distingué des transducteurs séquentiels avec une condition de Büchi et des transducteurs séquentiels avec une condition de Muller.

Ces différentes caractérisations utilisent des variantes de la propriété de jumelage introduite par Choffrut ainsi que la notion d'état constant. Nous avons appelé *constant* un état tel que tous les chemins finaux qui en sont issus aient la même étiquette de sortie. Par exemple, l'état 1 du transducteur de la figure 29 est constant alors que l'état 0 de ce même transducteur ne l'est pas. La terminologie est justifiée par le fait que la fonction réalisée par le transducteur obtenu en prenant cet état comme état initial est constante. Un transducteur satisfait la propriété de *jumelage faible* si toute paire de chemins

$$\begin{aligned} i &\xrightarrow{t|u} q \xrightarrow{v|w} q \\ i' &\xrightarrow{t|u'} q' \xrightarrow{v|w'} q', \end{aligned}$$

où i et i' sont initiaux vérifient les conditions suivantes :

- si q et q' ne sont pas constants, alors $|w| = |w'|$ et si de plus $w \neq \varepsilon$, alors $uw^\omega = u'w'^\omega$,
- si q n'est pas constant, si q' est constant et si $w \neq \varepsilon$ alors $uw^\omega = u'y_{q'}$ où $y_{q'}$ est la sortie constante des chemins finaux issus de q'.

La première condition exprime que les états non constants vérifient la propriété de jumelage telle que l'a définie Choffrut. La deuxième condition est une adaptation de la propriété de jumelage lorsque l'un des deux états est constant. Si le mot w' est non vide, le mot $y_{q'}$ est en effet égal à w^ω même si la boucle de sortie w' n'est pas acceptant. Le transducteur de la figure 29 ne vérifie pas la propriété de jumelage, mais il vérifie la propriété de jumelage faible. Les deux caractérisations que nous avons obtenues sont les suivantes.

Théorème 15 *Soit f une fonction de mots infinis réalisée par un transduc-*

teur $\mathcal{T}$. La fonction f est Muller séquentielle si et seulement si les conditions suivantes sont satisfaites :

- *la fonction f est continue,*
- *le transducteur $\mathcal{T}$ a la propriété de jumelage faible.*

Bien que le transducteur de la figure 29 vérifie la propriété de jumelage faible, la fonction réalisée par celui-ci n'est pas séquentielle parce qu'elle n'est pas continue.

Théorème 16 *Soit f une fonction de mots infinis réalisée par un transducteur $\mathcal{T}$. La fonction f est Büchi séquentielle si et seulement si les conditions suivantes sont satisfaites :*

- *la fonction f est Muller séquentielle,*
- *le domaine de f est accepté par un automate de Büchi déterministe.*

Il faut remarquer que la différence entre Büchi séquentielle et Muller séquentielle est uniquement due au domaine de la fonction f.

L'algorithme de déterminisation se déroule en deux étapes. La première étape est la construction d'un transducteur séquentiel qui réalise une extension de la fonction à un domaine plus large. Dans une deuxième étape, ce transducteur séquentiel est combiné avec un automate déterministe qui accepte le domaine de la fonction pour obtenir un transducteur qui réalise la fonction f.

La construction du transducteur séquentiel de la première étape est une généralisation de la méthode utilisée pour les mots finis. Les états de ce transducteur séquentiel sont encore des ensembles de paires formées d'un état et d'un mot. Par contre, le mot d'une paire est un mot infini ultimement périodique lorsque l'état de cette paire est un état constant. La preuve que le transducteur construit réalise bien une extension de la fonction f est relativement technique et nécessite plusieurs lemmes.

Il est possible de décider en temps polynomial la propriété de jumelage faible. Prieur [69] a montré que la continuité d'une fonction réalisée par un transducteur est aussi décidable en temps polynomial. Il est ainsi possible de décider en temps polynomial si une fonction est Muller séquentiel. Par contre, tester si l'ensemble des mots acceptés par un automate de Büchi peut être accepté par un automate de Büchi déterministe est un problème NP-difficile. A fortiori, tester si le domaine d'une fonction réalisée par un transducteur peut être accepté par un automate de Büchi déterministe est aussi un problème NP-difficile.

5 Automates et ordres linéaires

Dans les deux parties précédentes ont été considérés des automates et des transducteurs qui acceptent des mots infinis ou bi-infinis. Dans cette partie, on considère des généralisations de ces mots et des automates correspondants. Une première partie est consacrée aux mots transfinis et aux automates introduits par Büchi. Notre principale contribution est d'avoir développé une théorie algébrique de ces mots et d'avoir montré que des résultats importants comme le théorème des variétés pouvaient être étendus à ces mots. Une seconde partie est finalement consacrée aux mots sur des ordres linéaires. Nous avons introduit des automates pour ces mots et des expressions rationnelles. Nous avons étendu le théorème de Kleene aux mots sur les ordres dispersés et dénombrables.

5.1 Automates sur les ordinaux

Les travaux présentés dans cette partie ont été effectués en collaboration avec Nicolas Bedon dans le cadre de sa thèse de doctorat.

Les automates sur les ordinaux furent introduits par Büchi [23, 24, 25]. Büchi a considéré deux variantes essentielles d'automates sur les ordinaux. Dans les deux variantes, l'état à une position limite dépend des états qui sont apparus avant.

Dans la première variante, la longueur des mots acceptés est inférieure à ω^n pour un entier n fixé. Les ordinaux limites sont alors de la forme $\beta + \omega^k m$ pour k et m strictement positifs. L'état à la position indexée par $\beta + \omega^k m$ est égal à l'ensemble des états qui apparaissent infiniment souvent le long de la suite $\beta + \omega^k(m-1) + \omega^{k-1} i$ quand i parcourt les entiers. Cet état est donc un élément de $\mathcal{P}^k(Q)$ où $\mathcal{P}^k(Q)$ est défini par

$$\mathcal{P}^0(Q) = Q \quad \text{et} \quad \mathcal{P}^{k+1}(Q) = \mathcal{P}(\mathcal{P}^k(Q)).$$

Choueka fit une étude systématique de ces automates dans [36] où il montre en particulier une extension du théorème de Kleene pour ces automates. Ces automates sont d'ailleurs souvent appelés *automates de Choueka* bien qu'ils aient été introduits par Büchi.

Dans la seconde variante, l'état à une position limite dépend de tous les états qui apparaissent avant cette position de manière cofinale. La longueur des mots acceptés par des automates sur ce principe n'est plus limitée et elle peut être égale à n'importe quel ordinal. On peut montrer que les automates de la première variante correspondent à une classe particulière d'automates de la seconde variante qu'il est possible de caractériser [14]. Pour ces automates de la seconde variante, il existe à nouveau deux approches pour les

chemins. Une première façon est de considérer que l'état en une position limite β est l'ensemble cofinal du chemin en β qui est une partie de Q. C'est alors la transition suivante qui refait passer à un état q de Q. L'inconvénient de cette approche est qu'un chemin n'est pas vraiment une suite d'éléments de Q puisque les positions limites sont étiquetées par des parties de Q. Il faut en particulier adapter légèrement la définition de l'ensemble cofinal pour ne prendre en compte que les ordinaux qui sont successeurs. Une seconde approche consiste à considérer que l'automate effectue des transitions silencieuses au moment des passages de limites. L'automate a des transitions limites de la forme $P \to q$ sans étiquette. Si l'ensemble cofinal en β est égal à P, l'état en position β est alors l'état q. Au passage de la limite, l'automate passe directement de l'ensemble cofinal P à l'état q. L'avantage de cette approche est qu'un chemin devient une véritable suite d'éléments de Q. Il n'est pas difficile de voir que les deux approches sont équivalentes dans le sens où elle définissent les mêmes ensembles de mots. C'est cette seconde variante que nous choisirons dans la suite.

En 1962, Büchi [22] prouva la décidabilité de la logique monadique du second ordre de la structure $\langle \mathbb{N}, < \rangle$ des entiers munis de l'ordre. De manière globale, la méthode consiste à montrer une correspondance entre les formules de cette logique et les automates sur les mots infinis. La difficulté essentielle réside dans la preuve que la classe des ensembles acceptés par automate est fermée par complémentation. Büchi introduisit les automates sur les ordinaux afin d'étendre ce résultat de décidabilité à des structures de la forme $\langle \alpha, < \rangle$ pour un ordinal α donné [23, 24]. Le résultat principal qu'il a obtenu dans cette direction est pour α égal à ω_1, le premier ordinal non dénombrable. Comme les ordinaux inférieurs à ω^ω peuvent être définis dans cette logique, on a aussi la décidabilité si α est inférieur à ω^ω. Ce résultat est une extension naturelle de la décidabilité de la même logique pour la structure $\langle \mathbb{N}, < \rangle$. La méthode utilisée est d'ailleurs assez similaire. Elle consiste à construire pour chaque formule un automate qui accepte les modèles de la formule. La satisfiabilité de la formule se ramène au problème du vide pour l'automate. La construction de l'automate est faite par récurrence sur la taille de la formule. La quantification se traduit sur les automates par une projection de l'alphabet. Les opérateurs $\wedge$ et $\vee$ sont traduits directement sur les automates et la négation correspond à une complémentation. Cette dernière opération est de loin la plus délicate.

Nous renvoyons à [73] pour une introduction complète aux ordinaux. On va essentiellement utiliser les ordinaux pour indexer des suites. Comme c'est l'usage, on identifie un ordinal avec l'ensemble des ordinaux plus petits. Pour un ordinal α, un mot de longueur α est une suite de lettres indexées par les ordinaux inférieurs à α. Si α est un ordinal fini, on retrouve la définition

usuelle d'un mot comme une suite finie de lettres et si $\alpha = \omega$, on retrouve la définition d'un mot infini. La concaténation d'une suite $(x_\gamma)_{\gamma<\beta}$ de mots est le mot de longueur $\sum_{\gamma<\beta} |x_\gamma|$ obtenu en mettant bout à bout les mots x_γ.

5.1.1 Automates

Rappelons que le plus petit ordinal non dénombrable est appelé ω_1. Dans [15], on s'est principalement intéressé aux mots de longueur inférieure à ω_1, c'est-à-dire aux ordinaux dénombrables. Si l'on dépasse l'ordinal ω_1, quelques difficultés apparaissent et certains résultats vrais pour les mots finis ou infinis ne peuvent pas être étendus aux mots transfinis. En particulier, la famille des ensembles qui peuvent être acceptés par des automates n'est plus close par complément. Ce genre de problèmes voue à l'échec les approches algébriques. En revanche, si l'on se restreint à une classe plus petite d'ordinaux, la famille de mots que l'on considère n'est pas close par concaténation. Dans le cas de ω_1, tout produit d'une suite de longueur inférieure à ω_1 de mots eux-mêmes de longueur inférieure à ω_1 est encore de longueur inférieure à ω_1. Cette propriété est due au fait que ω_1 est un cardinal.

Définition 17 *Un automate sur les mots transfinis (appelé* automate transfini*) est un automate (Q, A, E, I, F) où Q est un ensemble fini d'états, A un alphabet fini, $E \subset (Q \times A \times Q) \cup (\mathcal{P}(Q) \times Q)$ l'ensemble des transitions et où I et F les ensembles d'états initiaux et finaux.*

Comme pour un automate usuel sur les mots finis, un automate transfini a un ensemble fini d'états parmi lesquels certains sont initiaux et/ou finaux. La seule différence est qu'un automate transfini possède deux types de transitions. D'une part, il possède des transitions de la forme (p, a, q) où p et q sont des états et a est une lettre. Ces transitions sont appelées *transitions successeurs* et sont notées $p \xrightarrow{a} q$. Elles correspondent aux transitions usuelles d'un automate sur les mots finis. D'autre part, il possède des transitions de la forme (P, q) où P est sous-ensemble d'états et q est un état. Ces transitions sont appelées *transitions limites* et sont notées $P \to q$. Un automate sur les mots finis peut être vu comme un automate transfini qui n'aurait pas de transitions limites.

L'automate de la Figure 30 possède les 4 transitions successeurs $0 \xrightarrow{a} 0$, $0 \xrightarrow{b} 0$, $1 \xrightarrow{a} 0$ et $1 \xrightarrow{b} 0$ qui sont représentées comme pour un automate usuel. Il possède également les deux transitions limites $\{0\} \to 1$ et $\{0, 1\} \to 1$.

Pour définir la notion de chemin dans un automate transfini, il est nécessaire de définir la notion d'ensemble cofinal. L'ensemble *cofinal* d'une suite

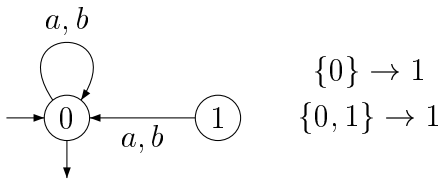


FIG. 30 – Automate transfini.

$c = (q_\gamma)_{\gamma<\alpha}$ en une position limite β est défini par

$$\lim_\beta c = \{q \mid \forall \eta < \beta \ \ \exists \lambda \quad \eta < \lambda < \beta \text{ et } q = q_\lambda\}.$$

L'ensemble cofinal $\lim_\beta q_\gamma$ est en quelque sorte l'ensemble des valeurs d'adhérence de la suite $(q_\gamma)_{\gamma<\alpha}$ quand γ tend vers β. Si β est dénombrable, un état q appartient à $\lim_\beta q_\gamma$ s'il existe une suite $(\lambda_n)_{n<\omega}$ d'ordinaux qui converge vers β telle que $q = q_{\lambda_n}$ pour tout n.

$$\begin{array}{rccccccc}
 & \vdots & \vdots & \vdots & \vdots & & \\
 & 0 & 0 & 0 & 0 & & \\
 & 0 & 0 & 0 & 0 & & \\
 & 0 & 0 & 0 & 0 & & \\
x = & 0 & 1 & 1 & 1 & \dots & 1 \\
\beta = & & \omega & \omega{\cdot}2 & \omega{\cdot}3 & \omega{\cdot}4 & \omega^2 & \\
\lim_\beta c = & & \{0\} & \{0\} & \{0\} & \{0\} & \{0,1\} &
\end{array}$$

FIG. 31 – Ensembles cofinaux de $c = 0^\omega(10^\omega)^\omega 1$.

La notion d'ensemble cofinal est illustrée par l'exemple suivant.

Exemple 18 Soit c la suite $0^\omega(10^\omega)^\omega 1$ de longueur $\omega^2 + 1$. On vérifie que $\lim_\omega c = \{0\}$, $\lim_{\omega\cdot 2} c = \{0\}$ et même $\lim_{\omega\cdot k} c = \{0\}$ pour $k \geq 1$ ainsi que $\lim_{\omega^2} c = \{0, 1\}$. Ces ensembles cofinaux sont représentés à la figure 31.

On peut maintenant donner la définition d'un chemin.

Définition 19 *Soit $\mathcal{A}$ un automate transfini et soit $x = (x_\gamma)_{\gamma<\alpha}$ un mot de longueur α. Un chemin c d'étiquette x est une $(\alpha+1)$-suite d'états $(q_\gamma)_{\gamma\leq\alpha}$ telle que*

– *pour tout $\beta < \alpha$, $q_\beta \xrightarrow{a_\beta} q_{\beta+1}$ est une transition successeur de $\mathcal{A}$;*

– *pour ordinal limite* $\beta \leq \alpha$, $\lim_\beta c \to q_\beta$ *est une transition limite de* $\mathcal{A}$.

Un fait important est qu'un chemin étiqueté par un mot de longueur α est une suite de longueur $\alpha + 1$ et non pas α. La longueur d'un chemin est donc un ordinal successeur. Un chemin a donc toujours un premier état q_0 et un dernier état q_α. Le chemin c est dit *initial* si q_0 est initial et *final* si q_α est final. Il est dit *acceptant* s'il est initial et final.

La suite $c = 0^\omega(10^\omega)^\omega 1$ de longueur $\alpha = \omega^2 + 1$ est bien un chemin de l'automate transfini de la figure 30. En effet, si c est égal à $(q_\gamma)_{\gamma<\alpha}$, l'état q_γ est égal à 1 si γ est limite et à 0 sinon. Comme l'automate possède les transitions $\{0\} \to 1$ et $\{0,1\} \to 1$, la suite c satisfait bien la définition donnée ci-dessus.

Comme dans les automates de mots finis, un chemin est une suite d'états qui vérifie des propriétés locales de compatibilité données par les transitions de l'automate. Lorsque deux états sont consécutifs, c'est-à-dire lorsque ce sont des états de la forme q_β et $q_{\beta+1}$ pour un ordinal $\beta < \alpha$, $q_\beta \xrightarrow{a_\beta} q_{\beta+1}$ doit être une transition de l'automate. Ainsi, chaque état q_β est relié à l'état qui le suit $q_{\beta+1}$ et chaque état est relié à son prédécesseur lorsque sa position β est ordinal successeur. Lorsque β est un ordinal limite, il n'y a pas d'état qui précède l'état q_β. L'état est donc relié à la partie du chemin qui le précède par la relation $\lim_\beta c \to q_\beta$. L'ensemble $\lim_\beta c$ joue donc le rôle d'un état fictif qui précéderait l'état q_β. Il faut noter que la transition $\lim_\beta c \to q_\beta$ est silencieuse puisqu'elle n'a pas d'étiquette. L'idée intuitive est qu'en arrivant au point β, l'automate effectue de façon spontanée cette transition qui ne dépend que de l'historique du chemin et non pas de la lecture d'une lettre.

Si x est le mot fini $a_0 \dots a_{n-1}$ de longueur n, un chemin étiqueté par x dans un automate transfini est une suite $q_0, \dots, q_n$ de $n+1$ états tels que $q_k \xrightarrow{a_k} q_{k+1}$ soit une transition pour tout $k < n$. La définition coïncide avec la notion de chemin dans un automate de mots finis. Les automates transfinis généralisent donc les automates de mots finis.

Les automates transfinis généralisent également les automates de Muller. En effet, un automate de Muller peut être vu comme un automate transfini en ajoutant un nouvel état final f ainsi que toutes les transitions limites $P \to f$ pour chacune des parties P de la table de l'automate de Muller.

5.1.2 ω_1-semigroupes

Pour les mots finis, il est bien connu que les automates sont équivalents aux semigroupes finis. A tout ensemble de mots reconnu par un automate fini est associé un semigroupe fini canonique appelé *semigroupe syntaxique*. Cette correspondance établit un pont entre d'un côté la théorie des automates et de l'autre côté l'algèbre.

Le premier résultat utilisant cette correspondance est le très beau résultat dû à Schützenberger [75] qui caractérise les langages sans étoile. Rappelons qu'un ensemble de mots est dit *sans étoile* s'il peut être engendré en partant des lettres et en utilisant les opérations booléennes et le produit de concaténation. Le théorème de Schützenberger établit qu'un langage est sans étoile si et seulement si son semigroupe syntaxique ne contient que des sous-groupes triviaux. D'autres résultats ont alors suivi comme la caractérisation des langages testables par morceaux par Simon [78] ou la caractérisation par Brzozowski, Simon [21] et McNaughton [56] des langages localement testables.

L'idée d'utiliser les propriétés algébriques des semigroupes syntaxiques fut développée de façon systématique par Eilenberg et Schützenberger [40, 41]. Le théorème des variétés établit une correspondance bijective entre d'une part certaines classes de langages reconnaissables, les *variétés de langages*, et d'autre part certaines classes de semigroupes finis, les *variétés de semigroupes*. Les variétés de langages sont des classes de langages reconnaissables fermées pour les opérations booléennes et quelques autres opérations. Les variétés de semigroupes finis sont des classes de semigroupes finis fermées par produit fini, passage au sous-semigroupe et passage au quotient. Elles peuvent être définies par des équations algébriques.

L'étude des propriétés algébriques des ensembles de mots infinis fut initiée par Pécuchet [63, 62]. Une approche plus satisfaisante fut ensuite proposée par Wilke [87, 88], Perrin et Pin [65]. L'idée principale est d'utiliser des semigroupes munis d'un produit infini qui donne une valeur à une suite de longueur ω d'éléments du semigroupe. Lorsque le semigroupe est fini, un tel produit peut toujours être défini en utilisant un résultat de Ramsey [71]. Il est même possible de décrire ce produit de façon finie.

Nous avons étendu cette théorie algébrique aux mots indexés par les ordinaux dénombrables [15]. Nous avons généralisé la notion de ω-semigroupe en introduisant des ω_1-semigroupes où le produit d'une suite d'éléments indexés par n'importe quel ordinal dénombrable est défini. Nous avons montré que ces objets algébriques sont équivalents aux automates de Büchi. La preuve de ce résultat donne un nouvel algorithme pour la déterminisation d'un automate. Büchi avait montré [23] que tout automate est équivalent pour les ordinaux dénombrables à un automate déterministe. Nous avons montré que tout automate est équivalent à un ω_1-semigroupe qui peut être effectivement calculé. Réciproquement, nous avons aussi montré que tout ω_1-semigroupe est équivalent à un automate déterministe. Cette réciproque est d'ailleurs la partie délicate de la preuve. Elle implique la construction non triviale d'un automate déterministe à partir d'un ω_1-semigroupe.

Nous avons également montré qu'à tout ensemble X rationnel de mots

indexés par des ordinaux dénombrables peut être associé un ω_1-*semigroupe syntaxique* qui est le plus petit ω_1-semigroupe reconnaissant l'ensemble X. Ce résultat nous a permis d'étendre la correspondance d'Eilenberg aux ensembles reconnaissables de mots transfinis et aux variétés de ω_1-semigroupes.

Dans le cas des mots infinis, un ω-semigroupe est une paire (S_+, S_ω) où S_+ est un semigroupe qui agit à gauche sur l'ensemble S_ω. Cet ω-semigroupe est en outre muni d'un ω-produit π qui associe un élément de S_ω à toute suite de longueur ω d'éléments de S_+. Dans un ω_1-semigroupe, les deux parties S_+ et S_ω d'un ω-semigroupe se retrouvent confondues. Un ω_1-semigroupe est en effet un ensemble S muni d'un ω_1-produit φ. Cet ω_1-produit associe un élément de S à toute suite d'éléments de S dont la longueur est un ordinal dénombrable. Il vérifie une généralisation de l'associativité qui signifie intuitivement qu'on peut regrouper arbitrairement les éléments de la suite sans changer la valeur du produit. Le ω_1-semigroupe S se trouve naturellement muni d'une structure de semigroupe. Lorsque le ω_1-produit est appliqué aux suites de longueur 2, il définit un produit associatif et l'action à gauche d'un ω-semigroupe est aussi définie par ce produit. La définition d'un ω_1-semigroupe est finalement plus simple que celle d'un ω-semigroupe puisque le produit associatif, l'action à gauche et le ω-produit sont regroupés en une seule et même opération. Ceci provient du fait que la concaténation des mots infinis est une opération partielle. Il est possible de concaténer un mot fini avec un mot infini, mais l'inverse est impossible car cela donnerait un mot de longueur strictement supérieure à ω. Dans le cas des mots de longueurs dénombrables, il n'y a aucune restriction sur la concaténation. Parce que ω_1 est aussi un cardinal, on peut sans problème concaténer une suite dénombrable de mots de longueur dénombrable.

En [15], nous avons montré que tout ω_1-langage rationnel admet un ω_1-semigroupe syntaxique. Cet ω_1-semigroupe divise tout autre ω_1-semigroupe reconnaissant le même ω_1-langage. Il peut être obtenu en quotientant l'ensemble de tous les mots transfinis par une congruence qui est la généralisation de la congruence d'Arnold [3] aux mots transfinis.

Nous avons également étendu le théorème d'Eilenberg aux mots transfinis. Les notions de variétés de semigroupes finis et de variétés de langages s'étendent aux ω_1-semigroupes finis et aux ω_1-langages. La preuve de la correspondance s'établit alors sans difficulté. Les ω_1-semigroupes que nous avons définis ne sont pas munis d'un ordre. Par contre, il est facile d'introduire des ω_1-semigroupes ordonnés et d'obtenir un théorème de correspondance pour les variétés positives comme cela est possible pour les mots infinis [67].

Dans l'article [26], nous avons donné trois exemples de correspondances entre des variétés de ω_1-semigroupes finis et des variétés de ω_1-langages. Nous avons caractérisé d'abord la classe des langages qui sont reconnus par des

automates dans lesquels les transitions limites qui s'intersectent se terminent dans le même état. Il s'avère que la variété correspondante de ω_1-semigroupes est définie par une équation qui a une interprétation topologique dans le cas des mots infinis. Elle caractérise les langages de mots infinis de la classe $\Delta_2 = \Pi_2 \cap \Sigma_2$ de la hiérarchie de Borel. Ce résultat est utilisé pour prouver qu'un ω_1-langage est reconnu par un automate extensif si et seulement si son ω_1-semigroupe syntaxique est $\mathcal{R}$-trivial et satisfait l'équation Δ_2. Ce résultat étend celui d'Eilenberg à propos des semigroupes $\mathcal{R}$-triviaux et des automates extensifs. Nous avons finalement caractérisé les ω_1-langages reconnus par des automates extensifs dont les transitions limites sont triviales.

5.2 Automates sur les ordres linéaires

Le travail présenté dans cette partie a été réalisé en collaboration avec Véronique Bruyère.

Dans cette partie, on définit des automates sur les ordres linéaires qui sont une généralisation naturelle des automates sur les ordinaux introduits par Büchi [24]. On a vu que les automates sur les ordinaux sont des automates usuels auxquels ont été ajoutées des transitions limites de la forme $P \to q$. Les automates introduits dans cette section possèdent des transitions limites de la forme $P \to q$ mais aussi des transitions limites de la forme $q \to P$. Ils rétablissent une symétrie entre la gauche et la droite que n'ont pas les automates sur les ordinaux. Le résultat principal que nous avons prouvé est qu'il est possible d'étendre le théorème de Kleene aux ordres dispersés et dénombrables [20].

On considère généralement que le premier résultat de la théorie des automates est le théorème de Kleene [50]. Ce théorème établit l'équivalence entre les automates et les expressions rationnelles. Un ensemble de mot est reconnu par un automate fini s'il peut être engendré à partir des lettres de l'alphabet en utilisant l'union, le produit de concaténation et l'étoile. Ce résultat est remarquable car il relie des propriétés de nature tout à fait différente : être reconnu par un automate d'une part et une propriété combinatoire d'autre part. Depuis lors, différentes extensions de ce théorème ont été prouvées pour différentes structures comme les arbres ou les traces.

Nous nous intéressons ici aux objets qui peuvent être linéairement ordonnés. Ceci comprend en particulier les mots finis, les mots infinis et bi-infinis ainsi que les mots sur les ordinaux. Pour chacune de ces classes d'objets, des automates appropriés ont été définis et une extension du théorème de Kleene a été prouvée. Büchi introduisit les automates sur les mots infinis [22] puis ensuite les automates sur les ordinaux en [24] où les expressions ω-rationnelles sont déjà définies. Un théorème de Kleene pour les mots ordinaux

de longueur inférieure à ω^ω a été prouvé par Choueka [36]. Wojciechowski l'a ensuite étendu à tous les ordinaux dénombrables [90]. Le cas des mots bi-infinis est finalement traité en [61, 47].

Ces différents résultats se présentent de façon non uniforme. À chacune des classes d'objets correspond un type particulier d'automates avec un fonctionnement adapté à ces objets. Dans un automate sur les mots finis, un chemin acceptant doit se terminer en un état final alors que dans un automate de Büchi, un chemin acceptant doit passer infiniment souvent par un état final. De même, les chemins dans les automates sur les ordinaux initialement introduits par Büchi sont formés d'états et d'ensembles d'états. Nous allons montrer qu'il est possible d'avoir une approche unifiée en considérant des mots indexés par des ordres linéaires dénombrables. Les mots finis, les mots infinis et bi-infinis et les mots sur les ordinaux vont alors correspondre à des ordres linéaires particuliers. Nous allons introduire des automates sur ces ordres linéaires qui seront simples et naturels tout en incluant les automates introduits précédemment. Les automates sur les mots finis, les mots infinis et bi-infinis et les mots ordinaux seront retrouvés en imposant des restrictions très naturelles à nos automates. Nous allons également définir des expressions rationnelles et prouver un théorème de Kleene associé.

Les mots indexés par des ordres linéaires dénombrables ont déjà été considérés par Courcelle [37]. Ils sont alors vus comme les frontières des arbres binaires étiquetés qui sont lues de gauche à droite. Des expressions rationnelles pour ces mots sont déjà définies en [37, 49, 81] pour décrire les solutions de systèmes d'équations. Ces opérations rationnelles ont la bonne propriété d'être des applications. Il n'y a aucune restriction sur leur utilisation. Par exemple, une puissance ω peut être concaténée avec un mot fini et le résultat peut à nouveau être itéré en utilisant une puissance ω inversée. Ces opérations permettent la caractérisation des frontières des arbres réguliers.

Dans un automate, un chemin étiqueté par un mot w est une suite d'états telle que cette suite d'états et la suite des lettres de w soient compatibles avec les transitions de l'automate. Si w est par exemple le mot fini $a_1 \dots a_n$, un chemin d'étiquette w est une suite d'états $q_0, \dots, q_n$ de longueur $n+1$ telle que chaque triplet (q_i, a_{i+1}, q_{i+1}) soit une transition de l'automate. Ces contraintes sont dites locales car elle portent sur les facteurs de longueur 3 de la suite

$$q_0 a_1 q_1 a_2 q_2 \dots q_{n-1} a_n q_n$$

obtenue en intercalant les états du chemin et les lettres du mot. Un chemin d'étiquette w peut aussi être vu comme une façon de décorer le mot w en insérant des états entre les lettres tout en respectant les contraintes locales données par les transitions comme c'est représenté à la figure 32.

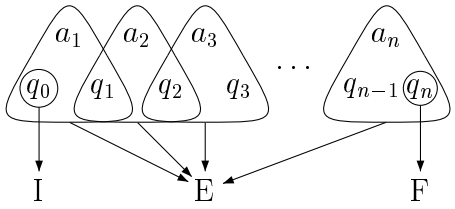


FIG. 32 – Les contraintes locales.

Ce que nous venons de dire pour les automates sur les mots finis s'applique aussi aux automates sur les mots infinis. Un chemin étiqueté par un mot infini x (de longueur ω) est une suite d'états de longueur ω qui respecte les contraintes locales dictées par les transitions de l'automate. La condition d'acceptation détermine si le chemin est acceptant. Que la condition d'acceptation soit de type Büchi, Muller ou Rabin, cette condition porte sur les états infiniment répétés le long du chemin. Cet ensemble d'états infiniment répétés peut être vu comme le voisinage de l'infini. La condition d'acceptation peut être alors considérée comme une contrainte locale sur le chemin en l'infini. Cette remarque est également vraie pour les chemins dans les automates sur les ordinaux. Les transitions limites de même que les transitions successeurs expriment des contraintes locales que doit respecter un chemin.

Pour définir la notion de chemin pour des mots qui sont indexés par des ordres linéaires quelconques, nous allons généraliser cette idée d'intercaler des états entre les lettres du mot tout en respectant des contraintes locales. Lorsque le mot est fini ou même de longueur ω, les positions auxquelles doivent être insérés les états sont évidentes, mais elles le sont moins dès que l'ordre sous-jacent au mot devient plus compliqué. Lorsque deux lettres du mot sont consécutives, il est clair qu'il faut insérer un état entre elles deux. De même, il semble naturel que chaque lettre soit précédée et suivie d'un état. Lorsque le mot est fini ou même ordinal, c'est suffisant mais il faut insérer plus d'états pour des mots plus compliqué. La notion qui va déterminer les positions des états est celle de *coupure* introduite par Dedekind pour la construction des réels.

5.2.1 Ordres linéaires

Nous commençons par quelques rappels sur les ordres linéaires. Un *ordre linéaire* J est un ensemble muni d'un ordre $<$ qui est total. Nous renvoyons à [73] pour une introduction très complète aux ordres linéaires. Dans la suite, les ordres linéaires sont utilisés pour indexer des suites et ils sont donc considérés à isomorphisme près. Par ordre linéaire, on entend type d'ordres li-

néaires.

Pour un ordre linéaire J, nous notons $-J$ l'ordre linéaire inverse obtenu en renversant l'ordre. L'ordre $<^*$ sur $-J$ est défini par $i <^* j$ si et seulement $j < i$ dans J.

Deux éléments j et k d'un ordre linéaire sont *consécutifs* si $j < k$ et s'il n'existe pas d'élément i tel que $j < i < k$. L'élément j est alors appelé le *prédécesseur* de k et k est appelé le *successeur* de j.

Un *intervalle* K d'un ordre linéaire J est un sous-ensemble de J tel que si j et k appartiennent à K et $j < i < k$ alors i appartient aussi à K.

Voici quelques exemples d'ordres linéaires. Nous commençons par quelques exemples simples comme les ordres finis et les ordinaux

Exemple 20 Un ordinal α est un ordre linéaire qu'on identifie généralement avec l'ensemble $\{\beta \mid \beta < \alpha\}$ des ordinaux inférieurs à α.

Exemple 21 Soit ζ^2 l'ensemble des paires (k, j) d'entiers relatifs. On définit la relation $<$ sur ζ^2 par $(k_1, j_1) < (k_2, j_2)$ si et seulement si $j_1 < j_2$ ou $j_1 = j_2$ et $k_1 < k_2$. La relation $<$ munit ζ^2 d'un ordre linéaire.

Exemple 22 Soit ζ^ω l'ensemble des suites $(j_n)_{n\geq 0}$ d'entiers relatifs qui sont presque nulles, c'est-à-dire telles qu'il y ait un nombre fini d'éléments j_n non nuls. On définit la relation $<$ sur ζ^ω par $(j_n)_{n\geq 0} < (k_n)_{n\geq 0}$ si et seulement si $j_m < k_m$ où m est le plus grand entier tel que $j_m \neq k_m$. La relation $<$ munit ζ^ω d'un ordre linéaire.

Nous introduisons maintenant la notion de coupure. Celle-ci est essentielle puisqu'elle permet de définir la notion de chemin dans un automate. Une *coupure* d'un ordre linéaire J est une paire (K, L) d'intervalles tels que $J = K \cup L$ et tels que pour tout $k \in K$ et tout $l \in L$, $k < l$. Les deux intervalles K et L sont donc disjoints et il forment une partition de J. L'ensemble des coupures de J est noté $\hat{J}$. Parmi les coupures, il en existe deux particulières qui sont les paires $(\varnothing, J)$ et $(J, \varnothing)$.

L'ensemble $\hat{J}$ est naturellement muni d'un ordre linéaire. Pour deux coupures $c_1 = (K_1, L_1)$ et $c_2 = (K_2, L_2)$, on définit la relation $c_1 < c_2$ si et seulement si K_1 est strictement inclus dans K_2 ($K_1 \subsetneq K_2$). Cette inclusion implique l'inclusion stricte de L_2 dans L_1 ($L_1 \supsetneq L_2$) et la définition est donc symétrique. Pour cet ordre, les coupures $(\varnothing, J)$ et $(J, \varnothing)$ sont le plus petit et le plus grand éléments de $\hat{J}$.

Nous illustrons la notion de coupure en décrivant les ordres $\hat{J}$ pour quelques ordres J vus précédemment.

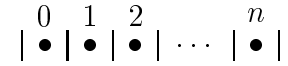


FIG. 33 – Les $n+1$ coupures de l'ordre à n éléments.

Exemple 23 Si J est un ordre linéaire fini à n éléments, par exemple $J = \{1, \ldots, n\}$, alors $\hat{J}$ est un ordre linéaire à $n+1$ éléments. En effet, l'ordre $\hat{J}$ contient les coupures $(\{1, \ldots, k\}, \{k+1, \ldots, n\})$ pour k égal à $0, \ldots, n$.

Exemple 24 Plus généralement, si J est un ordinal α, alors $\hat{J}$ est l'ordinal $\alpha + 1$. En effet, pour toute coupure (K, L) telle que $L \neq \varnothing$, l'intervalle L a un plus petit élément j. De plus, l'application qui associe j à (K, L) est bijective.

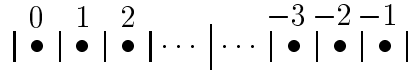


FIG. 34 – Les coupures de l'ordre $\omega + (-\omega)$.

À la vue des deux exemples précédents, il semble que chaque coupure est contiguë à un des éléments de l'ordre. Ce n'est pas toujours le cas comme le montre l'exemple suivant.

Exemple 25 Soit $J = \omega + (-\omega)$ l'ordre formé d'une copie des entiers naturels suivie d'une copie des entiers négatifs. On a $J = \mathbb{Z}$ et l'ordre $<_J$ est défini de la manière suivante. Quand m et n sont tous les deux positifs ou tous les deux strictement négatifs, $m <_J n$ est vrai si et seulement si $m < n$ est vrai. Quand m et n ne sont pas de même signe, $m <_J n$ est vrai si et seulement si $m \geq 0$ et $n < 0$. L'ordre J admet la coupure (K, L) où $K = \omega$ et $L = -\omega$ qui n'est attachée à aucun élément de J.

Les coupures consécutives jouent un rôle particulier dans la définition des chemins et méritent quelques remarques. À tout élément j de J, on peut associer deux coupures consécutives c_j^- et c_j^+ qui sont définies par $c_j^- = (K, \{j\} \cup L)$ et $c_j^+ = (K \cup \{j\}, L)$ où K et L sont donnés par $K = \{k \mid k < j\}$ and $L = \{k \mid j < k\}$. Réciproquement, à toutes coupures consécutives c^- et c^+, on peut associer un unique élément j tel que $c^- = c_j^-$ et $c^+ = c_j^+$.

Nous définissons maintenant la notion de mot sur un ordre linéaire qui généralise la notion de mot transfini. Soit A un alphabet et soit J un ordre linéaire. Un *mot* de longueur J sur A est une fonction de J dans A qui associe

à tout élément j une lettre a_j. Le mot est noté comme une suite en écrivant $(a_j)_{j\in J}$.

La définition précédente généralise bien la notion de mot fini, de mot infini ou même de mot transfini. En effet, si J est un ordre linéaire à n éléments, un mot de longueur J est une suite $a_1, \ldots, a_n$ comme on a l'habitude de le définir. Si J est égal à ω, on retrouve la notion de mot infini et si J est un ordinal on retrouve la notion de mot transfini. Par analogie, l'ordre linéaire J qui sert de support au mot est appelé la *longueur* du mot.

Exemple 26 Le mot $(b^{-\omega}ab^{\omega})^2$ est le mot de longueur $\zeta+\zeta$ où l'ordre $\zeta+\zeta$ est égal à $\{(k,j) \mid k \in \mathbb{Z} \text{ et } j \in \{1,2\}\}$. Le mot x est alors défini par

$$x_{k,j} = \begin{cases} a & \text{si } k = 0 \\ b & \text{sinon.} \end{cases}$$

5.2.2 Automates

Nous introduisons maintenant la définition d'un automate sur les ordres linéaires.

Définition 27 *Un automate est un quintuplet (Q, A, E, I, F) où Q est un ensemble fini d'états, A un alphabet fini, $E \subset (Q \times A \times Q) \cup (\mathcal{P}(Q) \times Q) \cup (Q \times \mathcal{P}(Q))$ l'ensemble des transitions et où I et F les ensembles d'états initiaux et finaux.*

Comme pour un automate usuel, un automate sur les ordres linéaires a un ensemble fini d'états parmi lesquels certains sont initiaux et/ou finaux. La différence est qu'un tel automate possède trois types de transitions. D'une part il possède des transitions de la forme (p, a, q) où p et q sont des états et a est une lettre. Ces transitions sont appelées *transitions successeurs* et notées $p \xrightarrow{a} q$. D'autre part, il possède des transitions de la forme (P, q) ou de la forme (q, P) où P est sous-ensemble d'états et q est un état. Les transitions (P, q) sont notées $P \to q$ et appelées *transitions limites gauches*. Les transitions (q, P) sont notées $q \to P$ et sont appelées *transitions limites droites*. Un automate transfini peut être vu comme un automate sur les ordres linéaires qui n'aurait pas de transitions limites droites.

Exemple 28 L'automate de la figure 35 possède les trois transitions successeurs $1 \xrightarrow{b} 1$, $1 \xrightarrow{a} 2$ et $2 \xrightarrow{b} 2$ ainsi que la transition limite gauche $\{2\} \to 0$ et la transition limite droite $0 \to \{1\}$.

Afin de définir un chemin, il est nécessaire d'introduire la notion de limite. On la définit pour un ordre linéaire quelconque J bien qu'elle soit uniquement

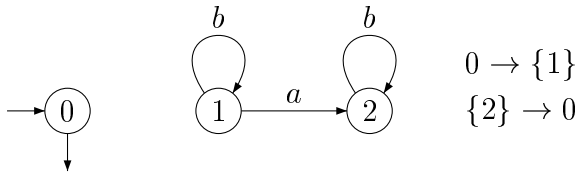


FIG. 35 – Automate sur les ordres linéaires.

utilisée pour l'ordre $\hat{J}$ des coupures. Soit Q un ensemble fini, J un ordre linéaire et soit $\gamma = (q_j)_{j\in J}$ un mot sur Q de longueur J. Soit j un élément fixé de J. Les limites gauche et droite de γ en j sont les deux sous-ensembles $\lim_{j^-}\gamma$ et $\lim_{j^+}\gamma$ de Q définis par

$$\lim_{j^-}\gamma = \{q \in Q \mid \forall k < j\ \exists i \quad k < i < j \text{ et } q = q_i\},$$
$$\lim_{j^+}\gamma = \{q \in Q \mid \forall j < k\ \exists i \quad j < i < k \text{ et } q = q_i\}.$$

Si l'élément j a un prédécesseur, l'ensemble $\lim_{j^-}\gamma$ est vide. Au contraire, si j n'est pas le plus petit élément de J et si j n'a pas de prédécesseur, alors $\lim_{j^-}\gamma$ est non vide puisque Q est fini.

Définition 29 *Soit $\mathcal{A}$ un automate et soit $x = (a_j)_{j\in J}$ un mot de longueur J. Un* chemin *γ d'étiquette x est une suite d'états $\gamma = (q_c)_{c\in\hat{J}}$ de longueur $\hat{J}$ telle que*

- *Pour toutes coupures consécutives c_j^- et c_j^+, $q_{c_j^-} \xrightarrow{a_j} q_{c_j^+}$ est une transition successeur de $\mathcal{A}$.*
- *Pour toute coupure c qui n'est pas la première coupure et qui n'a pas de prédécesseur, alors $\lim_{c^-}\gamma \to q_c$ est une transition limite gauche.*
- *Pour toute coupure c qui n'est pas la dernière coupure et qui n'a pas de successeur, alors $q_c \to \lim_{c^+}\gamma$ est une transition limite droite.*

Il faut remarquer que la longueur d'un chemin étiqueté par un mot x de longueur J est l'ordre $\hat{J}$ des coupures de l'ordre J. Cette différence existe déjà pour les mots finis puisqu'un chemin étiqueté par un mot de n lettres possède $n+1$ états. Cette différence s'accentue pour les mots plus compliqués.

D'après la définition précédente, il y a une transition qui arrive en l'état $\gamma(c)$ pour toute coupure c qui n'est pas la première coupure de $\hat{J}$. Cette transition est successeur si la coupure a un prédécesseur et c'est une transition limite gauche sinon. De façon duale, il y a une transition qui quitte $\gamma(c)$ pour tout coupure c qui n'est pas la dernière. Cette transition est successeur si c a un successeur et elle est une transition limite droite sinon.

Comme l'ordre $\hat{J}$ possède toujours un premier et un dernier élément, un chemin a toujours un premier et un dernier état. Un chemin est alors dit *acceptant* si ce premier état est initial et si ce dernier état est final.

L'intérêt de cette définition d'un chemin est qu'elle est suffisamment générale pour définir la notion de chemin pour n'importe quel ordre, mais qu'elle généralise de façon très naturelle les notions de chemins pour les automates de mots finis, de mots infinis et de mots transfinis. Soit x un mot fini $a_1 \dots a_n$ de longueur n. L'ordre sous-jacent est l'ordre $\{1 < \dots < n\}$ des entiers de 1 à n. Cet ordre possède $n+1$ coupures qui sont les paires $(\{1, \dots, k\}, \{k+1, \dots, n\})$ pour $0 \leq k \leq n$. Ces coupures peuvent être identifiées avec les entiers $\{0, \dots, n\}$. Un chemin étiqueté par le mot x est alors une suite $q_0, \dots, q_n$ d'états. Cette suite doit satisfaire que $q_{i-1} \xrightarrow{a_i} q_i$ est une transition successeur de l'automate puisque toutes les coupures ont un prédécesseur et un successeur. On retrouve bien la définition usuelle d'un chemin dans un automate.

Soit $x = a_0 a_1 a_2 \dots$ un mot infini. L'ordre J sous-jacent à x est l'ordinal ω. L'ensemble des coupures de J est donc l'ordinal $\omega + 1$ et les paires de coupures consécutives sont les paires $(j, j+1)$ pour $j < \omega$. Un chemin étiqueté par x est une suite d'états $q_0, q_1, q_2, \dots, q_\omega$ de longueur $\omega + 1$. D'après notre définition, la suite d'états doit satisfaire que $q_j \xrightarrow{a_j} q_{j+1}$ est une transition successeur de l'automate et que $\lim_{\omega^-} \gamma \to q_\omega$ est une transition limite gauche de l'automate. Il faut remarquer que l'ensemble limite $\lim_{\omega^-} \gamma$ est justement l'ensemble des états qui sont infiniment répétés le long du chemin γ. Toujours d'après notre définition, le chemin est acceptant si q_0 est initial et si le dernier état q_ω est final. On définit alors la table $\mathcal{T}$ de parties de Q par

$$\mathcal{T} = \{P \mid \exists q \in F \text{ tel que } P \to q \in E\}.$$

Un chemin est alors acceptant si q_0 est initial et si l'ensemble des états infiniment répétés appartient à la table $\mathcal{T}$. On retrouve ainsi la définition d'un chemin acceptant dans un automate de Muller dont la condition d'acceptation serait la table $\mathcal{T}$. Notre définition des chemins pour les mots infinis coïncide avec la définition usuelle, à l'exception que nous ajoutons un dernier état q_ω. La condition d'acceptation se trouve transposée dans les transitions et c'est ce dernier état qui détermine avec l'état initial si le chemin est acceptant ou non. Cela semble un peu artificiel si l'on s'intéresse uniquement aux mots infinis, mais c'est déjà nécessaire dès qu'on aborde les mots transfinis.

L'ordre des coupures d'un ordinal α est l'ordinal $\alpha + 1$. Notre définition des chemins coïncide exactement pour les mots transfinis avec la définition des chemins dans les automates sur ces mots transfinis.

Exemple 30 Considérons l'automate de la figure 35 et soit x le mot $(b^{-\omega}ab^{\omega})^2$

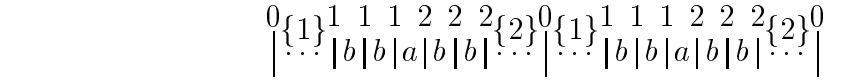


FIG. 36 – Un chemin étiqueté par le mot $(b^{-\omega}ab^{\omega})^2$.

de longueur $\zeta + \zeta$ déjà vu à l'exemple 26. Un chemin acceptant pour ce mot est le chemin représenté à la figure 36. Ce chemin est constitué de deux copies successives du chemin $01^{-\omega}2^{\omega}0$.

Il faut remarquer que la définition d'un chemin que nous avons donnée est très générale. Elle ne présuppose aucune hypothèse sur les ordres sous-jacents aux mots. Par contre, l'extension du théorème de Kleene que nous avons prouvée ne concerne que les ordres qui sont dispersés et dénombrables. Nous commençons par rappeler la définition des ordres dispersés et leur caractérisation due à Hausdorff.

Un ordre linéaire J est dit *dense* si pour tous $i < k$ dans J, il existe j dans J tel que $i < j < k$. Un ordre est dit *dispersé* (*scattered* en anglais) s'il ne contient pas de sous-ordre qui soit dense. L'idée intuitive est d'éviter les ordres qui sont trop éloignés du monde discret en interdisant les ordres denses comme $\mathbb{Q}$ ou $\mathbb{R}$. De même, nous nous restreignons aux ordres qui sont dénombrables. La caractérisation suivante des ordres dispersés et dénombrables est due à Hausdorff. Rappelons que $\mathcal{N}$ et $\mathcal{O}$ dénotent respectivement la classe des ordres linéaires finis et la classe des ordinaux dénombrables.

Théorème 31 (Hausdorff) *Un ordre linéaire dénombrable est dispersé si et seulement s'il appartient à $\bigcup_{\alpha\in\mathcal{O}} V_\alpha$ où les classes V_α sont définies par récurrence sur α de la façon suivante.*

1. $V_0 = \{0, 1\}$;
2. $V_\alpha = \{\sum_{j\in J} K_j \mid J \in \mathcal{N} \cup \{\omega, -\omega, \zeta\}$ *et* $K_j \in \bigcup_{\beta<\alpha} V_\beta\}$.

Il faut remarquer que si dans le point *2* de la définition des classes V_α, on impose à l'ordre J d'appartenir à $\mathcal{N} \cup \{\omega\}$ au lieu de $\mathcal{N} \cup \{\omega, -\omega, \zeta\}$, on retrouve une définition de tous les ordinaux dénombrables. Les ordres ζ^2 et ζ^ω considérés aux exemples 21 et 22 sont dispersés. Ils appartiennent respectivement aux classes V_2 et V_ω du théorème.

Le théorème précédent implique que les ordres linéaires J qui sont dispersés et dénombrables sont exactement ceux pour lesquels l'ordre $\hat{J}$ des coupures est dénombrable. Puisque un chemin étiqueté par un mot de longueur J est une suite d'états de longueur $\hat{J}$, cette restriction sur les ordres est assez naturelle. De plus, si l'ordre J est dispersé, alors l'ordre $\hat{J}$ est également dispersé.

5.2.3 Expressions rationnelles

Dans la suite de cette partie, nous supposons que tous les ordres considérés sont dispersés et dénombrables. Afin d'énoncer une extension du théorème de Kleene, nous donnons les opérations rationnelles qui permettent de définir les ensembles rationnels. Ces opérations comprennent bien sûr les opérations de Kleene usuelles pour les mots finis qui sont l'union, la concaténation et l'itération finie appelée étoile. Elles comprennent aussi l'itération oméga des mots infinis ainsi que l'itération ordinale introduite par Wojciechowski [90]. Trois nouvelles opérations sont encore nécessaires. Il s'agit de l'itération oméga inverse, de l'itération ordinale inverse et d'une dernière opération binaire qui est une sorte d'itération sur tous les ordres.

L'union, la concaténation et l'itération sont comme d'habitude dénotées par les symboles $+$, $\cdot$ et $*$. Les itérations oméga et ordinale sont dénotées par ω et $\sharp$ alors que les itérations oméga et ordinale inverses sont dénotées par $-\omega$ et $-\sharp$. L'iteration sur tous les ordres est dénotée par le symbole $\diamond$.

Nous commençons par rappeler les définitions des différentes itérations de façon unifiée. Pour une classe $\mathcal{J}$ d'ordres linéaires, l'itération générale $X^{\mathcal{J}}$ est définie par

$$X^{\mathcal{J}} = \{\prod_{j \in J} x_j \mid J \in \mathcal{J} \text{ et } x_j \in X\}.$$

Les ensembles X^*, X^ω, $X^{-\omega}$, $X^\sharp$ et $X^{-\sharp}$ sont respectivement égaux à $X^{\mathcal{J}}$ pour $\mathcal{J}$ égale à la classe $\mathcal{N}$ des ordres finis, la classe $\{\omega\}$ ne contenant que ω, la classe $\{-\omega\}$ ne contenant que $-\omega$, la classe $\mathcal{O}$ des ordinaux dénombrables et la classe $-\mathcal{O} = \{-\alpha \mid \alpha \in \mathcal{O}\}$ des ordinaux dénombrables inverses.

Avant de définir l'opération $\diamond$, nous introduisons d'abord une notation. Pour tout ordre J, l'ensemble $J \cup \hat{J}$ peut être muni de manière canonique d'un ordre qui étend ceux de J et de $\hat{J}$. Si pour tout élément j de J et toute coupure $c = (K, L)$ de J, on ajoute aux ordres de J et de $\hat{J}$ les relations $j < c$ si $j \in K$ ou $c < j$ si $j \in L$, on obtient un ordre total sur $J \cup \hat{J}$. Nous notons $J \cup \hat{J}^*$ l'ordre obtenu en supprimant de $J \cup \hat{J}$ les deux coupures particulières $(\varnothing, J)$ et $(J, \varnothing)$. Dans la définition qui suit, nous supposons que $J \cup \hat{J}^*$ est muni de cet ordre. Nous notons également $\mathcal{S}$ la classe de tous les ordres dispersés et dénombrables. Pour deux ensembles X et Y de mots, l'ensemble $X \diamond Y$ est égal à

$$X \diamond Y = \{ \prod_{j \in J \cup \hat{J}^*} z_j \mid J \in \mathcal{S} \text{ et } z_j \in X \text{ si } j \in J \text{ et } z_j \in Y \text{ si } j \in \hat{J}^*\}.$$

De manière intuitive, un mot de $X \diamond Y$ est obtenu en choisissant d'abord un ordre J qui est dispersé et dénombrable, puis en concaténant des mots

indexés par l'ordre $J \cup \hat{J}^*$, pris dans X si l'indice est dans J et dans Y si l'indice est une coupure de J.

Si l'ensemble Y ne contient que le mode vide ($Y = \{\varepsilon\}$), l'ensemble $X \diamond Y$ est égal $X^{\mathcal{S}}$ qui est l'itération sur tous les ordres de $\mathcal{S}$. L'opération $\diamond$ que nous avons définie est plus générale que cette simple itération puisqu'elle permet d'intercaler des mots de Y entre ceux de X. Dans la preuve de l'extension du théorème de Kleene, nous utilisons pleinement cette opération $\diamond$ avec Y différent de $\{\varepsilon\}$. Il est en fait possible de montrer que la simple itération ne suffit pas pour définir tous les ensembles rationnels.

Comme d'habitude, une expression rationnelle est un terme bien formé de l'algèbre libre sur l'ensemble $A \cup \{\varepsilon\}$ avec les symboles des opérations rationnelles comme symboles de fonctions. Chaque expression représente alors un ensemble de mots qui est défini par récurrence sur l'expression en utilisant les définitions des opérations rationnelles. Un ensemble de mots est dit *rationnel* s'il est représenté par une expression rationnelle. Nous pouvons maintenant énoncer notre résultat principal qui étend le théorème de Kleene pour les mots sur les ordres dispersés et dénombrables [20].

Théorème 32 *Un ensemble de mots indexés par des ordres dispersés et dénombrables est rationnel si et seulement s'il est reconnaissable par un automate.*

La preuve qu'un ensemble rationnel est accepté par un automate se fait par récurrence sur l'expression rationnelle. Pour chaque opération rationnelle, nous décrivons une construction correspondante sur les automates. Les constructions pour l'union, la concaténation et l'itération finie sont très semblables aux constructions classiques pour les automates sur les mots finis [64, p. 15].

La preuve que tout ensemble accepté par un automate est rationnel se fait par récurrence sur le nombre d'états de l'automate. C'est une généralisation de l'algorithme classique dû à McNaughton et Yamada. C'est en fait la partie difficile de la preuve.

6 Perspectives

Pour conclure, nous donnons quelques perspectives de recherche.

Les résultats de certains travaux suggèrent des prolongements. Dans les recherches effectuées en collaboration avec Wolfgang Thomas, nous avons montré que la logique du second ordre monadique de la structure $\langle \mathbb{N}, <, P \rangle$ est décidable lorsque P est un prédicat morphique [32]. Une extension naturelle est d'obtenir un résultat analogue pour la structure $\langle \{0,1\}^*, s_0, s_1 \rangle$ à deux

successeurs dont la décidabilité a été établie par Rabin [70]. La première difficulté est de trouver l'analogue des prédicats morphiques dans ce cadre. La seconde est que l'approche algébrique que nous avons utilisée semble mal se prêter aux structures arborescentes.

Les recherches en collaboration avec Véronique Bruyère nous ont permis d'introduire des automates acceptant des mots sur les ordres linéaires et de définir des expressions rationnelles pour ces mots [20]. Le premier résultat a été de prouver un théorème de Kleene lorsqu'on se restreint aux ordres dénombrables et dispersés. Il suscite plusieurs prolongements et de nombreuses questions. Les extensions les plus naturelles sont de s'affranchir de certaines hypothèses sur les ordres. Il semble en particulier que l'opération de mélange (*shuffle* en anglais) introduite par Heilbrunner [49] permette de considérer des ordres qui ne sont pas dispersés.

Une question très naturelle soulevée par notre résultat est le problème de la complémentation, c'est-à-dire de savoir si le complémentaire d'un ensemble rationnel est encore rationnel. Une réponse positive généraliserait le même résultat pour les mots finis, les mots infinis et les mots transfinis. Pour les mots finis, le problème est relativement simple mais il devient nettement plus difficile pour les mots infinis. Il est encore plus délicat pour les mots transfinis. Pour ces derniers, il existe essentiellement deux approches pour l'aborder. La première due à Büchi [23] consiste à prouver un résultat de déterminisation des automates, c'est-à-dire que tout automate est équivalent à un automate déterministe. Il semble cependant que cette propriété n'existe plus pour les automates sur les ordres linéaires. Tout automate sur les ordinaux n'est pas nécessairement équivalent à un automate co-déterministe. La seconde approche que nous avons développée avec Nicolas Bedon est une approche algébrique [15]. Elle consiste à définir l'analogue des semigroupes pour les mots transfinis et à montrer que ces objets algébriques sont équivalents aux automates. Cette approche semble pouvoir s'appliquer lorsqu'on se restreint aux ordres dénombrables et dispersés.

Comme l'avait déjà remarqué Courcelle [37], les mots sur les ordres linéaires dénombrables peuvent être vus comme les frontières des arbres binaires. Ce point de vue établit un lien entre les automates d'arbres et les automates sur les ordres. Tout ensemble de mots linéaires définit l'ensemble des arbres dont la frontière appartient à cet ensemble de mots. Une question assez naturelle est de savoir si ce passage des mots aux arbres préserve la rationalité et dans ce cas de trouver une correspondance bijective.

Références

[1] A. Aho, R. Sethi, and J. Ullman. *Compilers*. Addison-Wesley, 1986.

[2] J. Almeida. *Finite Semigroups and Universal Algebra*. World Scientific, 1994.

[3] A. Arnold. A syntactic congruence for rational ω-languages. *Theoret. Comput. Sci.*, 39 :333–335, 1985.

[4] M.-P. Béal. *Codage Symbolique*. Masson, 1993.

[5] M.-P. Béal. Puissance extérieure d'un automate déterministe, application au calcul de la fonction zêta d'un système sofique. *R.A.I.R.O.-Informatique Théorique et Applications*, 29(2) :85–103, 1995.

[6] M.-P. Béal and O. Carton. Determinization of transducers infinite words II. Soumis à TOCS.

[7] M.-P. Béal and O. Carton. Determinization of transducers over finite and infinite words. Technical Report 99–12, Institut Gaspard Monge, 1999. À paraître.

[8] M.-P. Béal and O. Carton. Asynchronous sliding block maps. *RAIRO – Theoretical Informatics and Applications*, 34 :139–156, 2000.

[9] M.-P. Béal and O. Carton. Determinization of transducers over infinite words. In U. Montanari et al., editors, *ICALP'2000*, volume 1853 of *Lect. Notes in Comput. Sci.*, pages 561–570, 2000.

[10] M.-P. Béal, O. Carton, Ch. Prieur, and J. Sakarovitch. Squaring transducers : An efficient procedure for deciding functionality and sequentiality. *Theoret. Comput. Sci.*, 200 ? À paraître.

[11] M.-P. Béal, O. Carton, Ch. Prieur, and J. Sakarovitch. Squaring transducers : An efficient procedure for deciding functionality and sequentiality. In G. Gonnet, D. Panario, and A. Viola, editors, *LATIN'2000*, volume 1776 of *Lect. Notes in Comput. Sci.*, pages 397–406, 2000.

[12] M.-P. Béal, O. Carton, and Ch. Reutenauer. Cyclic languages and strongly cyclic languages. In *STACS'96*, volume 1046 of *Lect. Notes in Comput. Sci.*, pages 49–59, 1996.

[13] M.-P. Béal, F. Mignosi, A. Restivo, and M. Sciortino. Forbidden words in symbolic dynamics. *Adv in Appl. Math.*, 25 :163–193, 2000.

[14] N. Bedon. *Langages reconnaissables de mots indexés par des ordinaux*. Thèse de doctorat, Université de Marne-la-Vallée, 1998.

[15] N. Bedon and O. Carton. An Eilenberg theorem for words on countable ordinals. In C. L. Lucchesi and A. V. Moura, editors, *Latin'98 : Theoretical Informatics*, volume 1380 of *Lect. Notes in Comput. Sci.*, pages 53–64. Springer-Verlag, 1998.

[16] J. Berstel and D. Perrin. *Theory of Codes.* Academic Press, 1984.

[17] J. Berstel and Ch. Reutenauer. Zeta functions of formal languages. *Trans. Amer. Math. Soc.*, 321 :533–546, 1990.

[18] R. Bowen. Symbolic dynamics. In *On axiom A diffeomorphism*, number 35 in CBMS Reg. Conf. AMS, 1978.

[19] R. Bowen and O. E. Lanford. Zeta functions of restrictions of the shift transformation. In *Proc. Symp. Pure Math. AMS*, volume 14, pages 43–50, 1970.

[20] V. Bruyère and O. Carton. Automata on linear orderings. In J. Sgall, A. Pultr, and P. Kolman, editors, *MFCS'2001*, volume 2136 of *Lect. Notes in Comput. Sci.*, pages 236–247, 2001. Rapport IGM 2000–12.

[21] J. A. Brzozowski and I. Simon. Characterizations of locally testable languages. *Discrete Math.*, 4 :243–271, 1973.

[22] J. R. Büchi. On a decision method in the restricted second-order arithmetic. In *Proc. Int. Congress Logic, Methodology and Philosophy of science, Berkeley 1960*, pages 1–11. Stanford University Press, 1962.

[23] J. R. Büchi. Transfinite automata recursions and weak second order theory of ordinals. In *Proc. Int. Congress Logic, Methodology, and Philosophy of Science, Jerusalem 1964*, pages 2–23. North Holland, 1964.

[24] J. R. Büchi. Decision methods in the theory of ordinals. *Bull. Am. Math. Soc.*, 71 :767–770, 1965.

[25] J. R. Büchi and D. Siefkes. *The Monadic Second Order Theory of All Countable Ordinals*, volume 328 of *Lectures Notes In Mathematics.* Springer-Verlag, Berlin, 1973.

[26] O. Carton. $\mathcal{R}$-trivial languages of words on countable ordinals. À paraître dans TCS.

[27] O. Carton. *Mots infinis, ω-semigroupes et topologie.* Thèse de doctorat, Université Paris 7, 1993.

[28] O. Carton. A hierarchy of cyclic languages. *RAIRO – Theoretical Informatics and Applications*, 31(4) :355–369, 1997.

[29] O. Carton. Wreath product and infinite words. *J. Pure and Applied Algebra*, 153 :129–150, 2000.

[30] O. Carton and R. Maceiras. Computing the Rabin index of a parity automaton. *RAIRO – Theoretical Informatics and Applications*, 33 :495–505, 1999.

[31] O. Carton and M. Michel. Unambiguous Büchi automata. *Theoret. Comput. Sci.*, 2001. A paraître.

[32] O. Carton and W. Thomas. The monadic theory of morphic infinite words and generalizations. In M. Nielsen and B. Rovan, editors, *MFCS'2000*, number 1893 in Lect. Notes in Comput. Sci., pages 275–284, 2000.

[33] Ch. Choffrut. Minimizing subsequential transducers : a survey. A paraître.

[34] Ch. Choffrut. Une caractérisation des fonctions séquentielles et des fonctions sous-séquentielles en tant que relations rationnelles. *Theoret. Comput. Sci.*, 5 :325–338, 1977.

[35] Ch. Choffrut. *Contribution à l'étude de quelques familles remarquables de fonctions rationnelles.* Thèse d'État, Université Paris VII, 1978.

[36] Y. Choueka. Finite automata, definable sets, and regular expressions over ω^n-tapes. *J. Comput. System Sci.*, 17 :81–97, 1978.

[37] B. Courcelle. Frontiers of infinite trees. *RAIRO – Theoretical Informatics and Applications*, 12(4) :319–337, 1978.

[38] V. Diekert and G. Rozenberg. *The Book of Traces.* World Scientific, 1995.

[39] S. Eilenberg. *Automata, Languages and Machines*, volume A. Academic Press, New York, 1972.

[40] S. Eilenberg. *Automata, Languages and Machines*, volume B. Academic Press, New York, 1976.

[41] S. Eilenberg and M.-P. Schützenberger. On pseudovarieties. *Advances in Mathematics*, 19 :413–418, 1976.

[42] C. C. Elgot and J. E. Mezei. On relations defined by generalized finite automata. *IBM Journal Res. and Dev.*, 9 :47–68, 1965.

[43] C. C. Elgot and M. O. Rabin. Decidability and undecidibility of extensions of second (first) order theory of (generalized) successor. *J. Symb. Logic*, 31(2) :169–181, 1966.

[44] E. A. Emerson and C. S. Jutla. Tree automata, Mu-calculus and determinacy. In *Proc. 32th Symp. on Foundations of Computer Science*, pages 368–377, 1991.

[45] Ch. Frougny and J. Sakarovitch. Synchronized relations of finite words. *Theoret. Comput. Sci.*, 108 :45–82, 1993.

[46] D. Giammarresi and A. Restivo. Two-dimensional languages. In G. Rosenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 215–267. Springer-Verlag, 1997.

[47] D. Girault-Beauquier. Bilimites de langages reconnaissables. *Theoret. Comput. Sci.*, 33(2–3) :335–342, 1984.

[48] F. Gire. Two decidability problems for infinite words. *Inform. Proc. Letters*, 22 :135–140, 1986.

[49] S. Heilbrunner. An algorithm for the solution of fixed-point equations for infinite words. *RAIRO – Theoretical Informatics and Applications*, 14(2) :131–141, 1980.

[50] S. C. Kleene. Representation of events in nerve nets and finite automata. In C. E. Shannon, editor, *Automata studies*, pages 3–41. Princeton university Press, Princeton, 1956.

[51] S. C. Krishnan, A. Puri, and R. K. Brayton. Structural complexity of ω-languages. In *STACS'95*, volume 900 of *Lect. Notes in Comput. Sci.*, pages 143–156, Berlin, 1995. Springer-Verlag.

[52] O. Kupferman and M. Y. Vardi. Weak alternating automata are not so weak. In *Proc. 5th Israeli Symp. on Theory of computing and Systems*, pages 147–158. IEEE Computer Soc. Press, 1997.

[53] A. Maes. An automata theoretic decidability proof for the first-order theory of $\langle \mathbb{N}, <, P \rangle$ with morphic predicate P. *Journal of Automata, Languages and Combinatorics*, 4 :229–245, 1999.

[54] A. Manning. Axiom A diffeomorphisms have rational zeta fonctions. *Bull. London Math. Soc.*, 3 :215–220, 1971.

[55] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Inform. Control*, 9 :521–530, 1966.

[56] R. McNaughton. Algebraic decision procedures for local testability. *Math. Systems Theory*, 8 :60–76, 1974.

[57] A. W. Mostowski. Regular expressions for infinite trees and a standard form for automata. In A. Skowron, editor, *Computation theory*, volume 208 of *Lect. Notes in Comput. Sci.*, pages 157–168. Springer-Verlag, Berlin, 1984.

[58] A. W. Mostowski. Hierarchies of weak automata and weak monadic formulas. *Theoret. Comput. Sci.*, 83 :323–335, 1991.

[59] D. Muller. Infinite sequences and finite machines. In *Switching Theory and Logical Design, Proc. Fourth Annual Symp. IEEE*, pages 3–16. IEEE, 1963.

[60] D. Muller and P. Schupp. Alternating automata on infinite objects, determinacy and Rabin's theorem. In M. Nivat and D. Perrin, editors, *Automata on Infinite Words*, volume 192 of *Lect. Notes in Comput. Sci.*, pages 100–107. Springer-Verlag, 1985.

[61] M. Nivat and D. Perrin. Ensembles reconnaissables de mots bi-infinis. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 47–59, 1982.

[62] J.-P. Pécuchet. Étude syntaxique des parties reconnaisables de mots infinis. In *ICALP'86*, volume 226 of *Lect. Notes in Comput. Sci.*, pages 294–303, 1986.

[63] J.-P. Pécuchet. Variétés de semigroupes et mots infinis. In B. Monien and G. Vidal-Naquet, editors, *STACS'86*, volume 210 of *Lect. Notes in Comput. Sci.*, pages 180–191, 1986.

[64] D. Perrin. Finite automata. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 1, pages 1–57. Elsevier, 1990.

[65] D. Perrin and J.-É. Pin. Semigroups and automata on infinite words. In J. Fountain and V. A. R. Gould, editors, *NATO Advanced Study Institute Semigroups, Formal Languages and Groups*, pages 49–72. Kluwer academic publishers, 1995.

[66] J.-É. Pin. *Varieties of formal languages*. North Oxford, London and Plenum, New-York, 1986. (Traduction de *Variétés de langages formels*, Masson, 1984).

[67] J.-É. Pin. Positive varieties and infinite words. In C. L. Lucchesi and A. V. Moura, editors, *Latin'98 : Theoretical Informatics*, volume 1380 of *Lect. Notes in Comput. Sci.*, pages 76–87. Springer-Verlag, 1998.

[68] J.-É. Pin and J. Sakarovitch. Une application de la représentation matricielle des transductions. *Theoret. Comput. Sci.*, 35 :271–293, 1985.

[69] Ch. Prieur. How to decide continuity of rational functions on infinite words. *Theoret. Comput. Sci.*, 250 :71–82, 2001.

[70] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141 :1–35, 1969.

[71] F. P. Ramsey. On a problem on formal logic. *Proc. London Math. Soc.*, 30(2) :264–286, 1929.

[72] Ch. Reutenauer. $\mathbb{N}$-rationality of zeta functions. *Adv in Appl. Math.*, 29 :1–17, 1997.

[73] J. G. Rosenstein. *Linear Orderings*. Academic Press, 1982.

[74] S. Safra. On the complexity of ω-automata. In *29th Annual Symposium on Foundations of computer sciences*, pages 24–29, 1988.

[75] M.-P. Schützenberger. On finite monoids having only trivial subgroups. *Inform. Control*, 8 :190–194, 1965.

[76] M.-P. Schützenberger. Sur une variante des fonctions séquentielles. *Theoret. Comput. Sci.*, 11 :47–57, 1977.

[77] D. Siefkes. Decidable extensions of monadic second order successor arithmetic. In J. Doerr and G. Hotz, editors, *Automatentheorie und Formale Sprachen*, pages 441–472, Mannheim, 1970. B.I. Hochschultaschenbücher.

[78] I. Simon. Piecewise testable events. In *2nd GI Conf.*, volume 33 of *Lect. Notes in Comput. Sci.*, pages 214–222, Berlin, 1975. Springer-Verlag.

[79] H. Straubing. The wreath product and its applications. In J.-É. Pin, editor, *Proc. of the sixteenth Spring School of Theoretical Computer Science*, volume 386 of *Lect. Notes in Comput. Sci.*, pages 15–24. Springer-Verlag, 1989.

[80] R. S. Streett. Propositional dynamic logic of looping and converse is elementary decidable. *Inform. Control*, 54 :121–141, 1982.

[81] W. Thomas. On frontiers of regular sets. *RAIRO – Theoretical Informatics and Applications*, 20 :371–381, 1986.

[82] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 4, pages 133–191. Elsevier, 1990.

[83] W. Thomas. Complementation of Büchi automata revisited. In J. Karhumäki et al., editors, *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 109–122. Springer-Verlag, 1999.

[84] W. Wadge. *Handwritten Notes*. PhD thesis, U.C. Berkeley, 1976.

[85] K. Wagner. On ω-regular sets. *Inform. Control*, 43 :123–177, 1979.

[86] A. Weber and R. Klemm. Economy of description for single-valued transducers. *Inform. Comput.*, 118 :327–340, 1995.

[87] T. Wilke. An Eilenberg theorem for ∞-languages. In *ICALP'91*, volume 510 of *Lect. Notes in Comput. Sci.*, pages 588–599, Berlin, 1991. Springer-Verlag.

[88] T. Wilke. An algebraic theory for regular languages of finite and infinite words. *Int. J. Alg. Comput.*, 3(4) :447–489, 1993.

[89] T. Wilke and H. Yoo. Computing the Rabin index of a regular language of infinite words. *Inform. Comput.*, 130(1) :61–70, 1996.

[90] J. Wojciechowski. Finite automata on transfinite sequences and regular expressions. *Fundamenta informaticæ*, 8(3-4) :379–396, 1985.

7 Publications

Articles en revues

[1] M.-P. Béal and O. Carton. Determinization of transducers over finite and infinite words. À paraître dans Theoret. Comput. Sci.

[2] O. Carton and W. Thomas. The monadic theory of morphic infinite words and generalizations. À paraître dans Inform. Comput.

[3] M.-P. Béal, O. Carton, Ch. Prieur, and J. Sakarovitch. Squaring transducers : An efficient procedure for deciding functionality and sequentiality. À paraître dans Theoret. Comput. Sci.

[4] O. Carton and M. Michel. Unambiguous Büchi automata. Technical Report 2001–06, Institut Gaspard Monge, 2001. À paraître dans Theoret. Comput. Sci.

[5] O. Carton. $\mathcal{R}$-trivial languages of words on countable ordinals. Technical Report 2001–05, Institut Gaspard Monge, 2001. À paraître dans Theoret. Comput. Sci.

[6] M.-P. Béal and O. Carton. Computing the prefix of an automaton. *RAIRO-Informatique Théorique et Applications*, 34(6) :503–514, 2000.

[7] M.-P. Béal and O. Carton. Asynchronous sliding block maps. *RAIRO-Informatique Théorique et Applications*, 34 :139–156, 2000.

[8] O. Carton. Wreath product and infinite words. *J. Pure and Applied Algebra*, 153 :129–150, 2000.

[9] O. Carton and R. Maceiras. Computing the Rabin index of a parity automaton. *RAIRO-Informatique Théorique et Applications*, 33 :495–505, 1999.

[10] O. Carton and D. Perrin. The Wagner hierarchy of ω-rational sets. *Int. J. Alg. Comput.*, 9(5) :597–620, 1999.

[11] O. Carton. A hierarchy of cyclic languages. *RAIRO-Informatique Théorique et Applications*, 31(4) :355–369, 1997.

[12] O. Carton and D. Perrin. Chains and superchains for ω-rational sets, automata and semigroups. *Int. J. Alg. Comput.*, 7(7) :673–695, 1997.

[13] O. Carton. Chain automata. *Theoret. Comput. Sci.*, 161 :191–203, 1996.

Actes de conférences

[1] V. Bruyère and O. Carton. Automata on linear orderings. In J. Sgall, A. Pultr, and P. Kolman, editors, *MFCS'2001*, volume 2136 of *Lect. Notes in Comput. Sci.*, pages 236–247, 2001.

[2] O. Carton and W. Thomas. The monadic theory of morphic infinite words and generalizations. In M. Nielsen and B. Rovan, editors, *MFCS'2000*, number 1893 in Lect. Notes in Comput. Sci., pages 275–284, 2000.

[3] M.-P. Béal and O. Carton. Determinization of transducers over infinite words. In U. Montanari et al., editors, *ICALP'2000*, volume 1853 of *Lect. Notes in Comput. Sci.*, pages 561–570, 2000.

[4] M.-P. Béal, O. Carton, Ch. Prieur, and J. Sakarovitch. Squaring transducers : An efficient procedure for deciding functionality and sequentiality. In G. Gonnet, D. Panario, and A. Viola, editors, *LATIN'2000*, volume 1776 of *Lect. Notes in Comput. Sci.*, pages 397–406, 2000.

[5] O. Carton and M. Michel. Unambiguous Büchi automata. In G. Gonnet, D. Panario, and A. Viola, editors, *LATIN'2000*, volume 1776 of *Lect. Notes in Comput. Sci.*, pages 407–416, 2000.

[6] M.-P. Béal and O. Carton. Asynchronous sliding block maps. In G. Rozenberg and W. Thomas, editors, *Developments in Language Theory*, pages 47–59. World Scientific, 1999.

[7] N. Bedon and O. Carton. An Eilenberg theorem for words on countable ordinals. In Cláudio L. Lucchesi and Arnaldo V. Moura, editors, *Latin'98 : Theoretical Informatics*, volume 1380 of *Lect. Notes in Comput. Sci.*, pages 53–64. Springer-Verlag, 1998.

[8] O. Carton and D. Perrin. The Wadge-Wagner hierarchy of ω-rational sets. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Automata, Languages and Programming*, volume 1256 of *Lect. Notes in Comput. Sci.*, pages 17–35. Springer-Verlag, 1997.

[9] O. Carton and D. Perrin. Chains and superchains in ω-semigroups. In J. Almeida, G. Gomes, and P. Silva, editors, *Semigroups, Automata and Languages*, pages 17–28. World Scientific, 1994.

[10] M.-P. Béal, O. Carton, and Ch. Reutenauer. Cyclic languages and strongly cyclic languages. In *STACS'96*, volume 1046 of *Lect. Notes in Comput. Sci.*, pages 49–59, 1996.

[11] O. Carton. Chain automata. In *IFIP World Computer Congress '94*, pages 451–458, Hamburg, 1994. Elsevier (North-Holland).

En préparation

[1] O. Carton and Ch. Choffrut. Periodicity and roots of transfinite strings. Soumis à TCS.

[2] V. Bruyère and O. Carton. Automata on linear orderings. Technical Report 2000–12, Institut Gaspard Monge, 2000. Soumis à JCSS.

[3] M.-P. Béal and O. Carton. Determinization of transducers over infinite words II. Technical Report 2001–07, Institut Gaspard Monge, 2001. Soumis à TOCS.

Mémoires

[1] O. Carton. *Mots infinis, ω-semigroupes et topologie*. Thèse de doctorat, Université Paris 7, 1993. Rapport LITP-TH 93-08.

[2] O. Carton. Étude de la thèse de Hing Leung. Rapport de DEA, Université Paris 7, 1988. Rapport LITP 88.