

The structure of orders in the pushdown hierarchy

Laurent Braud

▶ **To cite this version:**

Laurent Braud. The structure of orders in the pushdown hierarchy. Modeling and Simulation. Université Paris-Est, 2010. English. NNT : 2010PEST1009 . tel-00587409

HAL Id: tel-00587409
https://pastel.hal.science/tel-00587409

Submitted on 20 Apr 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ PARIS-EST

THÈSE

pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ PARIS-EST

The structure of orders in the pushdown hierarchy
Les structures d'ordre dans la hiérarchie à pile

Spécialité **informatique**
École doctorale MSTIC
Soutenue publiquement par **Laurent Braud**
le **10 décembre 2010**

JURY :

Zoltán	Ésik,	rapporteur,
Wolfgang	Thomas,	rapporteur,
Arnaud	Carayol,	examinateur,
Damian	Niwiński,	examinateur,
Dominique	Perrin,	examinateur,
Didier	Caucal,	directeur de thèse.

Contents

1 Introduction **7**

1.1 (en français) . . . 7

1.2 (in English) . . . 13

2 Preliminaries **19**

2.1 Notations and first structures . . . 19

2.1.1 Finite words . . . 19

2.1.2 Structures . . . 20

2.1.3 Graphs . . . 21

2.1.4 Deterministic trees . . . 22

2.2 Linear orderings . . . 22

2.2.1 Ordinals . . . 24

2.2.2 Scattered orderings and Hausdorff rank . . . 26

2.2.3 Orders in a deterministic tree . . . 29

2.3 Logic . . . 30

2.3.1 First-order logic . . . 30

2.3.2 Monadic second-order logic . . . 31

2.3.3 Decidability . . . 31

2.4 Graph transformations . . . 33

2.4.1 Graph interpretations . . . 33

2.4.2 Graph expansions . . . 35

2.5 The pushdown hierarchy . . . 37

2.5.1 Definition . . . 37

2.5.2 Some properties . . . 38

3 Linear order construction **43**

3.1 Ordinals in the pushdown hierarchy . . . 43

3.2 Powers of ζ . . . 45

3.3 n-regular presentation . . . 48

3.3.1 Prefix-recognizable graphs . . . 49

3.3.2 Configuration graphs of n-hopdas . . . 49

3.3.3 Encoding ordinals . . . 50

3.4 Covering graphs . . . 53
3.4.1 Fundamental sequence . . . 54
3.4.2 Covering graphs . . . 56
3.4.3 Other properties of covering graphs . . . 60
3.4.4 Strictness of covering graphs in the hierarchy . . . 62
3.4.5 The case of $\mathcal{G}_{\varepsilon_0}$ . . . 65

4 The structure of tree frontiers **67**
4.1 Tree-walking automaton . . . 67
4.2 From graphs to frontiers . . . 69
4.3 Ordinals . . . 73
4.4 Scattered linear orders . . . 78
4.4.1 Trees with scattered frontiers . . . 78
4.4.2 Permutation of subtrees . . . 79
4.4.3 Cantor-Bendixson rank of deterministic trees . . . 81
4.4.4 Hausdorff rank of scattered orders in Graph_n . . . 86
4.5 Finite combs . . . 87

5 Schemes and morphic words **91**
5.1 Recursion schemes . . . 92
5.1.1 Definition . . . 92
5.1.2 Schemes in the pushdown hierarchy . . . 94
5.2 Morphic words . . . 97
5.2.1 Definition and properties . . . 97
5.2.2 Construction in the pushdown hierarchy . . . 99
5.2.3 Words in Graph_2 are morphic, direct proof . . . 99
5.2.4 Words in Graph_2 are morphic, by recursion schemes . . . 102
5.3 Second order . . . 106
5.3.1 Second-order morphic words . . . 106
5.3.2 Second-order scheme ω-frontiers . . . 109
5.3.3 Liouville word . . . 117

List of notations **119**

Index **121**

Bibliography **123**

List of Figures

1.1 Un graphe fini et une propriété de ce graphe. 7
1.2 L'arbre binaire complet. 8
1.3 A finite graph and one of its properties. 13
1.4 The complete binary tree. 14

2.1 Example of a graph : the ladder 21
2.2 The graph representation of the ordinal $\omega + 2$. 24
2.3 The ladder and its unfolding. 36
2.4 Treegraph of the complete binary tree. 36
2.5 The pushdown hierarchy. 38
2.6 Exemple of graph constructions in the hierarchy. 39

3.1 Finite graph G_3 which unfolding has frontier ω^3. 44
3.2 Folded graphs of trees of frontier ζ, η and $\omega(1 + \eta)$. 45
3.3 Folded regular graph of a tree of frontier ζ^ω. 46
3.4 $\{0, 1\}$-treegraph of ω. 48
3.5 The operation $\mathsf{inc}(\omega^\omega)$. 52
3.6 Covering graph of ω^ω. 56
3.7 Exponentiation of the covering graph of ω. 62

4.1 Order in a finite tree t and "arranged" tree $s(t)$. 72
4.2 Non-full tree of frontier ω having a branch with infinitely many 0's. 74
4.3 A finite graph G, "spanning tree" T, completed tree $\bar{T}$, and unfolding. . . . 76
4.4 Orders in a tree. 84

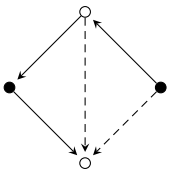
5.1 Rules for $\mathfrak{S}_1$, of frontier ω^ω. 95
5.2 General rules for $\mathfrak{S}_n$, of frontier $\omega \uparrow\uparrow (n + 1)$. 95
5.3 Paperfolding sequence. 98
5.4 A graph which unfolding yields the morphic word $abaab \dots a^{2^i} b \dots$. . . . 100
5.5 General shape of the folded graph. 101
5.6 Rules of a scheme which frontier is a morphic word. 105
5.7 Order-2 safe scheme which frontier is the Champernowne word. 107
5.8 Order-2 safe scheme which frontier is the Liouville word. 118

Chapter 1

Introduction

1.1 (en français)

En logique mathématique, on distingue les objets mathématiques, ou *structures*, et leurs propriétés, ou *logique*. Les structures à leur tour sont constituées d'*éléments* et de *relations* entre ces éléments. Dans cette thèse, nous travaillerons uniquement sur ce que l'on appelle des "graphes colorés dont les arcs sont étiquetés", c'est-à-dire des structures dont les relations sont d'arité au plus 2. De plus, ces structures seront dénombrables — on peut numéroter chaque objet avec un entier — et de présentation finie — il existe une quantité finie d'information permettant de représenter la structure de façon non ambigüe.



"Un chemin entre deux sommets blancs est en pointillés ou passe par un sommet noir."

Figure 1.1: Un graphe fini et une propriété de ce graphe.

La logique la plus simple est celle dite du premier ordre, car elle manipule simplement les éléments proprement dits de la structure considérée. Elle permet d'exprimer des propriétés du type "tout sommet colorié en noir a un voisin blanc" ou "s'il existe un sommet blanc, alors il existe 3 arcs distincts en pointillés". Elle est cependant restreinte à des propriétés ponctuelles et ne dit rien sur la structure dans sa globalité. Une solution est alors de considérer une logique plus forte, qui permet de manipuler directement des ensembles d'éléments. C'est la logique du second ordre monadique, qui permet d'exprimer des propriétés raisonnablement compliquées. Des exemples classiques sont "il y a un chemin entre x et y" ou "le sous-arbre t est infini". On la notera souvent par le sigle MSO.

De façon générale, la logique du second ordre désigne l'ensemble des formules qui parlent des relations entre les éléments. Mais elle a le défaut d'être paradoxalement trop puissante; elle échappe ainsi aux outils usuels de l'informatique, selon la définition donnée

ci-après. C'est pourquoi on se restreint aux relations qui ne prennent qu'une variable : ce sont bien les ensembles. Ceci explique l'adjectif *monadique*.

Notre problème n'est pas de connaître entièrement une structure donnée, mais de savoir exprimer ses propriétés. Cette nuance autorise l'usage d'un automatisme. En effet, la question qui se pose est alors la suivante.

> Étant données une logique et une structure, existe-t-il un algorithme qui prenne en entrée une formule close de cette logique et renvoie OUI ou NON selon que la structure satisfait la formule ou non?

C'est la question de la *décidabilité de du model-checking* de la logique à laquelle appartiennent les formules que l'on veut tester. Savoir construire un tel algorithme est un défi majeur de l'informatique d'aujourd'hui, car cette question est naturellement liée à la notion de vérification de programmes. En effet, savoir si un programme fait effectivement ce que l'on veut — par exemple, savoir s'il termine — est expressible par une formule, que doit vérifier la structure des configurations du programme.

Construction de structures

Par exemple, un résultat fondemental de Rabin [Rab69] indique que l'arbre binaire complet, esquissé Figure 1.2, a une théorie monadique décidable. Plus généralement, pour une logique donnée, peut-on espérer trouver une caractérisation de toutes les structures qui jouissent de la même propriété? Probablement pas, mais cela n'empêche pas les scientifiques de chercher à en décrire le plus possibles. En particulier, pour la logique du second ordre monadique, l'intérêt de la communauté scientifique se porte sur la *hiérarchie à pile* [Cau03], aussi appelée *hiérarchie de Caucal*.

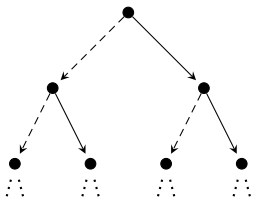


Figure 1.2: L'arbre binaire complet.

Pour décrire historiquement cette hiérarchie, il nous faut remonter à [MS85], où les auteurs s'intéressent aux graphes des configurations des automates à piles en partant d'une configuration donnée. Il caractérisent exactement ces graphes par un critère géométrique : si on fixe un sommet, et que l'on retire successivement ce sommet, puis les sommets à distance 1, puis à distance 2 et ainsi de suite, on obtient une suite de graphes. Le résultat fondamental est que l'ensemble de ces graphes est fini à isomorphisme près : la suite se

répète. Cette propriété leur permet d'étendre le résultat de Rabin à tout une classe de graphes. Ce résultat sera étendu dans [Cou90, Cou11] aux graphes HR-équationnels.

On a donc deux manières de décrire ces graphes, que l'on peut appeler définition *interne* — en choisissant un ensemble pour les sommets, et les arcs comme des relations entre ces sommets — ou *externe* — par les propriétés structurelles du graphe. Si l'on préfère, ces deux approches considèrent que les objets importants sont respectivement les sommets ou les arcs. Dans [Cau96], le même principe est appliqué pour définir une classe plus large, celle des graphes *préfixe-reconnaissables*. Ceux-ci sont à la fois définis par des relations de récriture préfixe entre des mots, et par des transformations de graphes à partir d'un graphe fini.

Dans cette thèse, nous utiliserons le plus souvent la caractérisation externe, qui décrit un graphe en donnant une série de transformations à partir d'un arbre fini; un graphe sera donc par définition un ensemble d'arcs. Il y aura deux transformations fondamentales. D'une part, le dépliage d'un graphe renvoie l'arbre des chemins dans ce graphe à partir d'un sommet donné. D'autre part, l'interprétation monadique permet de "réorganiser" de façon régulière la structure d'un graphe. Ces deux opérations préservent la décidabilité de la logique monadique. En partant de la classe des graphes finis, puis en appliquant un dépliage suivi d'une interprétation, on obtient la classe des préfixe-reconnaissables. En itérant ce processus, on obtient une suite de classes de graphes distinctes : c'est la hiérarchie à pile, décrite par Caucal [Cau03].

Ordinaux

Une autre famille de graphes ayant une logique MSO décidable est bien plus connue. C'est la classe des ordinaux dénombrables, ou classe d'ordres totaux ayant la propriété de bon ordre, c'est-à-dire que chaque sous-ensemble non vide doit avoir un plus petit élément. Ils ont été introduits par Cantor [Can97] qui en formule les propriétés de base. Ces objets sont fréquemment utilisés dans les mathématiques actuelles. Ils généralisent l'arithmétique des nombres entiers, et surtout bénéficient des preuves par induction : si on prouve qu'une propriété vraie pour tout $\beta < \alpha$ est vraie pour α, on a alors établi la propriété de façon générale.

Büchi [Bü65, Büc73], en utilisant des automates, a prouvé la décidabilité de la logique MSO pour les ordinaux dénombrables; il a été suivi par Shelah [She75] par une méthode compositionnelle entièrement différente.

Il est donc naturel de chercher l'intersection de ces deux grandes familles. Cette thèse prend donc sa racine dans la question suivante :

Quels sont exactement les ordinaux du niveau n de la hiérarchie à pile?

La question est d'autant plus importante que les exemples concrets de graphes dans la hiérarchie sont nombreux au premier niveau, mais beaucoup moins par la suite. Depuis [Mas74] et [KNU02], on sait que la hiérarchie est bien séparée en classes distinctes, mais

avoir une collection d'exemples aussi simples que les ordinaux établirait une forme de "mesure de complexité" de la hiérarchie. Dans l'autre sens, la réponse permettrait de mieux comprendre les ordinaux par l'étude des méchanismes nécessaires à leur construction

Dans le même ordre d'idée se pose la question plus générale des ordres linéaires, c'est-à-dire des classes d'ordres totaux n'ayant pas la propriété de bon ordre, dont $\mathbb{Z}$ en est le premier exemple. Evidemment, il existe des ordres linaires dénombrables dont la logique MSO n'est pas décidable, même assez simples. On cherche donc un seul sens : un critère nécessaire pour ces ordres. Pour commencer, il est naturel de chercher du côté des ordres *dispersés* (*scattered*), c'est-à-dire ne contenant pas de sous-ordre isomorphe à $\mathbb{Q}$. Cette classe bien connue a été classifiée par Hausdorff [Hau08] qui donne une "mesure ordinale" pour chaque ordre. Il existe donc un lien des ordres dispersés vers le cas plus simple et plus connu des ordinaux.

Automates et ordres

L'intersection du domaine des structures liées à des automates et de celui des ordinaux a fait naître bien d'autres résultats. Au sens le plus large, d'anciens travaux de Church [Chu38] et Kleene [Kle38] établissent quels sont les plus grands ordinaux récursifs, c'est-à-dire qui peuvent être exprimés par une machine de Turing. Nous nous intéressons ici à des modèles de calcul plus simples, plus proches du domaine des structures *automatiques*, c'est-à-dire dont les relations sont définies par un *transducteur* fini, c'est-à-dire un automate à deux entrées. Les résultats [Del04, KRS05] établissent que les ordinaux automatiques sont plus petits que ω^ω, et que les ordinaux arbre-automatiques — où le transducteur reconnaît des relations sur les arbres — sont ceux plus petits que ω^{ω^ω}. De la même manière, de récents résultats [BÉ09, BÉ10] définissent les ordres linéaires et les ordinaux aux premiers niveaux de la hiérarchie à pile : les ordinaux du deuxième niveau sont plus petits que ω^{ω^ω}, et les ordres dispersés ont un rang de Hausdorff plus petit que ω^ω. C'est cette voie que cette thèse poursuivra.

Il est également naturel de considérer la notion d'ordre linéaire *coloré*, c'est-à-dire de mots sur un alphabet fini, mais indexés par un ordre infini : par exemple, un nombre réel entre 0 et 1 peut être vu comme un mot sur l'alphabet $[0, 9]$ indexé par $\mathbb{N}$, ou de façon équivalente comme l'ordre ω coloré par l'alphabet $[0, 9]$. Les mots infinis acceptés par automates de Büchi [Büc62] sont le premier exemple de tels ordres. Un tel automate accepte un mot indexé par $\mathbb{N}$ s'il passe infiniment de fois dans un état final. Cette définition a été adaptée dans [NP82] pour considérer les mots bi-infinis, c'est-à-dire indexés par $\mathbb{Z}$. Büchi [Bü65] décrit également le processus d'un automate acceptant des mots indexés par des ordinaux, ce qui lui permet de montrer la décidabilité de la logique MSO. Plus récemment, Bruyère et Carton [BC07, BC02, BC06a] ont considéré des automates acceptant des mots indexés par des ordres dispersés, et obtiennent un théorème de Kleene.

Arbres solutions de schémas récursifs

La notion de mot infini indexé par $\mathbb{N}$ mérite également qu'on la recherche dans la hiérarchie à pile. Les premiers mots infinis que l'on rencontre sont les mots ultimement périodiques, qui sont les plus simples des mots infinis. A l'étape suivante, on voit apparaître des mots plus complexes, connus sous le nom de mots morphiques : ce sont les points fixes d'application de morphismes. Si Δ est une lettre, et que τ est un morphisme tel que $\tau(\Delta)$ est un mot commençant par Δ, alors $\tau(\tau(\Delta))$ également, et ainsi de suite : on obtient alors un mot infini. Ce sont là — à codage près — les mots morphiques.

Cette définition s'approche assez de la construction de termes de la hiérarchie à pile par les schémas récursifs. Ces objets, introduits semble-t-il par Ianov [Ian60] au premier niveau, puis par Nivat [Niv72], ont été amenés à l'ordre supérieur par Damm [Dam77, Dam82]. On peut parler de grammaires de termes : considérons deux ensembles typés dits *terminaux* et *non-terminaux*, et également l'ensemble des termes sur ces ensembles en suivant le typage. Chaque non-terminal F a une règle de récriture prenant en compte les arguments de F, de telle façon qu'un terme ayant pour tête ce non-terminal se récrit en un terme. On répète alors l'opération sur un nouveau non-terminal. Même si l'opération est infiniment répétée, de tels schémas peuvent avoir un arbre limite, point fixe de l'opération : c'est une *solution* du schéma récursif. On a donc une manière simple de construire un arbre infini avec des règles de récriture.

Ces schémas ont récemment été remis sur le devant de la scène, notamment grâce aux travaux [KNU01, KNU02] sur la décidabilité de la logique MSO sous une contrainte dite de *sûreté*, puis en retirant cette contrainte dans [AdMO05, Ong06, KNUW05]. En particulier, les arbres sûrs sont exactement les arbres-termes de la hiérarchie. Pour en revenir aux questions évoquées plus haut, il semble naturel de relier les schémas du premier ordre avec les mots morphiques, et de vérifier que l'on obtient bien les seconds comme ordres sous-tendus par les premiers. Se pose alors la question de la généralisation : comment faire évoluer la notion de mot morphique pour coller à celle de schéma récursif d'ordre supérieur?

Plan et contributions

Le chapitre 2 fixe les notations et les objets utilisés, en commençant par les notions de graphe et d'arbre. Nous rappelons les définitions et premières propriétés des ordres linéaires, ainsi que le cas particulier des ordinaux. Enfin, nous décrivons la hiérarchie à pile par les transformations de graphes.

Dans le chapitre 3, nous illustrons la définition interne de la hiérarchie par la construction d'ordinaux et de puissances du type d'ordre de $\mathbb{Z}$ par des transformations de graphes. Le résultat important est que les ordinaux plus petits qu'une tour exponentielle d'ω de taille $n+1$ sont dans le n-ième niveau de hiérarchie; ceci inclut donc tous les ordinaux plus petits que ε_0, qui est le plus petit ordinal tel que $\omega^{\varepsilon_0} = \varepsilon_0$. Une définition interne due à

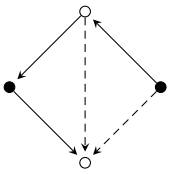
[Car05] est donnée, et est encore illustrée par les ordinaux. Enfin, nous nous interrogeons sur une propriété essentielle des ordinaux : d'après Büchi [Bü65], la logique monadique ne peut pas toujours distinguer deux ordinaux. Pour chaque ordinal plus petit que ε_0, nous exhibons alors une structure aussi expressive n'ayant pas cette contrainte. Les résultats de ce chapitre apparaissent pour la plupart dans [Bra09].

Le chapitre 4 établit le résultat inverse : il prouve que la tour d'exponentielle de taille $n + 1$ ne peut être dans le n-ième niveau de la hiérarchie. Pour ce faire, nous commençons par établir l'égalité entre ordres comme graphe de la hiérarchie et structure des feuilles, dans l'ordre lexicographique, des arbres de la hiérarchie. Ce résultat nous permet de raisonner par récurrence sur le niveau de la hiérarchie. Nous obtenons un résultat similaire sur le rang de Hausdorff des ordres dispersés, qui mesure une certaine complexité de l'ordre. Enfin, les mêmes techniques aboutissent à un résultat sur la taille des sous-arbre finis des peignes de la hiérarchie, c'est-à-dire des arbres ayant une unique branche infinie, qui est la branche la plus à droite. Ce chapitre apparaît en grande partie dans [BC10].

Le dernier chapitre approche la hiérarchie pas le biais des schémas de récursion d'ordre supérieur. Nous y considérons les feuilles des arbres solutions de ces schémas formant des mots de type ω. Si l'on considère des arbres réguliers, il est simple de voir que ces mots sont ultimement périodiques. Au niveau suivant, nous prouvons que l'on obtient exactement les mots dits morphiques. Ce résultat est alors étendu au niveau supérieur et définit une nouvelle classe de mots bénéficiant des propriétés des graphes de la hiérarchie. Ces résultats font l'objet d'un article en cours de préparation [Bra].

1.2 (in English)

Mathematical logic distinguishes mathematical objects, or *structures*, and their properties, or *logic*. These structures are in turn made of *elements* and *relations* between these elements. In this thesis, we will only work on "colored graphs with labeled arcs", that is structures where relations are of arity at most 2. Moreover, these structures will be countable — the objects can be numbered — and of finite presentation — there is a finite amount of data allowing an unambiguous representation of the structure.



"Each path between two white vertices is hashed or goes through a black vertex."

Figure 1.3: A finite graph and one of its properties.

The simplest logic is the so-called first-order logic, and it only quantifies the proper elements of the given structure. We can express properties like "every black vertex has a white neighbour" or "if there at least one white vertex, then there are at least 3 distinct hashed arcs". It is nonetheless limited : for instance, over graphs of bounded degree, it is restricted to local properties, as testified by Gaifman's locality theorem [Gai82]. A solution is then to consider a stronger logic, which directly considers sets of elements. This is called the monadic second-order (MSO) logic; it can express more complex properties, like "there is a path between x and y", or "the deterministic subtree t is infinite".

More generally, second-order logic is a system where variables range over relations between elements. But is has the paradoxal drawback of being too powerful; usual tools (e.g. decision procedures) of computer science cannot reach this logic. This is why we restrict ourselves to relations with only one variable : these are exactly sets. This explains the adjective *monadic*.

Our problem is not to know entirely a given structure, but to know its properties. This subtelty allows the use of an algorithm. The question is indeed the following.

> For a given logic and structure, is there an algorithm taking a closed formula of this logic and returns whether the formula is true in the structure or not?

It is the question of *decidability of the model-checking* of the logic in a given structure. Knowing how to build such an algorithm is a challenge of modern computer science, since this question is naturally linked to program verification. Indeed, the required behaviour of a program can be expressed by a formula, which should be checked by the structure of configurations of the program.

Structure constructions

For instance, a fundamental result of Rabin [Rab69] states that the complete binary tree, drawn in Figure 1.4, has a decidable monadic theory. More generally, for a given logic, can we hope to find a characterisation of all structures enjoying the same property? Probably not, but this does not forbid scientists to try and describe as much of these structures as possible. For monadic second-order logic, the scientific community is interested in the *pushdown* or *Caucal hierarchy*.

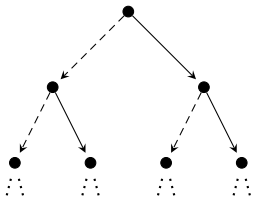


Figure 1.4: The complete binary tree.

The history of this hierarchy goes back to the result of Muller and Schupp [MS85], where the authors study the configuration graphs of pushdown automata starting from a given configuration. They characterize these graphs with a geometrical criterion. Given a vertex, we remove this vertex, then the vertices a distance 1, then distance 2, and so on. We get a sequence of graphs. The fundamental result is that the resulting set of connected graphs is finite up to isomorphism : the sequence repeats itself. This property extends the result of Rabin to a larger class of graphs, and is extended to HR-equational graphs in [Cou90, Cou11].

We have therefore two ways to describe these graphs, which can be called *inner* definition — by giving a set of vertices and constructing relations between vertices — and *outer* definition — via structural properties of the graph. Said differently, this approach considers that the main objects are respectively the vertices or the arcs. In [Cau96], the same idea is applied to define the larger set of *prefix-recognizable* graphs. These graphs also have a twofold definition : one by rewriting relations on words, and the second with graph transformations from a finite graph.

In this thesis, we will most of the time use the external characterization. That is, a graph will be described by a sequence of transformations from a finite tree. There will be two main transformations. First, the unfolding of a graph from one of its vertices yields the tree of the paths from this vertex. Second, the monadic interpretation "reorganizes" the structure in a regular way. These two operations preserve the decidability of model-checking for monadic second order logic. Starting from the set of finite graphs, and applying an unfolding and an interpretation, we get the set of prefixe-recognizable graphs. By iterating this process, we get a sequence of distinct classes of graphs : it is the pushdown hierarchy, described by Caucal [Cau03].

Ordinals

Another well-known class of graphs having a decidable MSO logic is the class of countable ordinals, or total orders having the well-ordering property, *i.e.* where each nonempty subset has a smaller element. They have been introduced by Cantor [Can97] who formulated basic properties. For instance, they generalize the arithmetic of natural number, and enjoy induction proofs : if a property true for all $\beta < \alpha$ is true for α, then it is true for all ordinals.

With the use of automata, Büchi [Bü65, Büc73] proved the decidability of model-checking of MSO logic for each countable ordinal; it was followed by Shelah, who proves the same result by the very different compositional method.

It is therefore natural to study the intersection of these two large families. This thesis takes its root in the following question :

> What are exactly the ordinals found in the n^{th} level of the pushdown hierarchy?

The importance of this question relies on the fact that we have many known examples of graphs in the hierarchy in the first levels, but very few in the next. Since [Mas74] and [KNU02], we know that the hierarchy is separated into distinct classes. Nonetheless, a collection of examples as simple as ordinals would establish a "complexity mesure" of the hierarchy. In the other direction, we could have a better understanding of ordinals by studying the mechanisms used in their construction.

The question of more general linear orders follows immediately. We consider the classes of total orders not having the property of well-ordering, the first example being $\langle \mathbb{Z}, < \rangle$. Of course, many of those orders have an undecidable monadic theory, even when restricting to countable orders. We look for one direction, *i.e.* a necessary criterion for these orders. For a start, it is natural to look at *scattered* orders, *i.e.* not having any suborder isomorphic to $\mathbb{Q}$. This well-known class of orders has been classified by Hausdorff [Hau08] where he gives an "ordinal measure" for countable scattered orders. There is therefore a link between scattered orders and the easier case of ordinals.

Orders and automata

It is not the first time that the domain of structures linked to automata meets ordinals. For the upper bounds, works of Church [Chu38] and Kleene [Kle38] state which are the greatest recursive ordinals, *i.e.* which can be expressed by a Turing machine. We study here simpler models of computation. For instance, this thesis is closer to works related to *automatic* structures, which relations are defined by a finite *tranducer*, *i.e.* an automaton with several entries. Results of [Del04, KRS05] state that automatic ordinals are smaller that ω^ω, and that tree-automatic ordinals — where the tranducer works on trees instead of words — are smaller that ω^{ω^ω}. In the same way, recent results [BÉ09, BÉ10] characterize

orders of the first levels of the pushdown hierarchy : ordinals of the second level are smaller than ω^{ω^ω}, and scattered orders have a Hausdorff rank smaller than ω^ω. This thesis follows this direction.

It would also be natural to consider the notion of *colored* linear order, that is words on a finite alphabet, but indexed by an infinite order : for instance, a real number between 0 and 1 can be seen as a word on the alphabet $[0, 9]$ indexed by $\mathbb{N}$, or equivalently as the order ω colored by the alphabet $[0, 9]$. Infinite words accepted by Büchi automata [Büc62] are the first examples of these orders. Such an automaton accepts a word indexed by $\mathbb{N}$ if it passes infinitely many time through a final state. This definition was adapted in [NP82] to consider bi-infinite words, *i.e.* indexed by $\mathbb{Z}$. Büchi [Bü65] also describes the process of words indexed by ordinals, and shows decidability of MSO logic for each such structure. Recently, Bruyère and Carton [BC07] considered automata accepting words indexed by scattered orderings. They reached a stronger Kleene theorem, which was then extended in [BC02, BC06a].

Solutions of recursion schemes

The notion of infinite words indexed by $\mathbb{N}$ is of central importance and deserves to be also studied in the hierarchy. The first encountered infinite words are the ultimately periodic words, which are the simplest ones. At the next level, more complex words appear : they are known as *morphic words*, because they can be built as fixpoints of morphisms on letters. If Δ is a "starting" letter, and τ is such that $\tau(\Delta)$ begins with Δ, then $\tau(\tau(\Delta))$ also does, and so on. We get therefore an infinite word. Up to a final coding, this defines the morphic words.

This definition is rather close of the construction of terms in the hierarchy by recursion schemes. These objects were introduced by Ianov [Ian60], then Nivat [Niv72], and have been brought to the higher order by Damm [Dam77, Dam82]. They can be called term grammars. Let *terminals* and *nonterminals* be two typed sets; we can consider the set of terms on these sets respecting the typing rules. Each nonterminal F has a rewriting rule taking in consideration the arguments of F, in such a way that a subterm having F as head symbol can be naturally rewritten in another subterm. The operation is then repeated on another nonterminal. Even if this algorithm does not terminate, a recursion scheme admits a limit (possibly infinite) tree : it is called the solution of the recursion scheme. We have therefore a simple way to construct an infinite tree with rewriting rules.

These schemes were recently reconsidered by the works [KNU01, KNU02] on the decidability of MSO logic under the constraint of *safety*, then removing this constraint in [AdMO05, Ong06, KNUW05]. In particular, the safe trees are exactly the term-tress of the hierarchy. To go back to the forementioned questions, it is natural to link the first-order schemes with morphic words. We can check that we get the latter as orders hidden in the former. The natural question is therefore : what happens in the next levels? How can we extend the notion of morphic word to stick to the notion of recursion scheme?

Outline and contributions

The first chapter sets notations and defines the objects used. We begin by the notions of graph and tree, then linear orders and the particular case of ordinals. We then describe the pushdown hierarchy through graph transformations.

Chapter 3 illustrates the internal definition by the construction of ordinals and powers of $\mathbb{Z}$ with the help of graph transformations. The important result is that ordinals smaller that a exponential tower of ω of size $n+1$ are all in the n^{th} level of the hierarchy; this includes therefore all ordinals smaller than ε_0, the smallest ordinal such that $\omega^{\varepsilon_0} = \varepsilon_0$. An internal definition due to [Car05] is given and illustrated by ordinals again. Next we try and avoid an essential property of ordinals : according to Büchi [Bü65], MSO logic cannot distinguish two ordinals in general. For each ordinal smaller than ε_0, we exhibit a structure as expressive as the ordinal but which can be characterized by MSO logic. Most of the results of this chapter appear in [Bra09].

Chapter 4 states the converse result : it shows that the $n+1$-exponential tower of ω cannot be in the n^{th} level of the hierarchy. For this result, we start by stating the equality between orders as graphs of the hierarchy and the structure of leaves in lexicographic order of trees of the hierarchy. This result allows the application of induction on the hierarchy. We get a similar result on the Hausdorff rank of scattered orders, measuring a certain complexity of orders. The same techniques lead to a result on the size of subtrees of *combs* of the hierarchy, *i.e.* trees having a unique infinite branch. A large part of this chapter appears in [BC10].

The last chapter defines the hierarchy by higher-order recursion schemes. We consider here the case where leaves of trees solution of these schemes form words of type ω. If we consider regular trees, it is easy to see that these words are ultimately periodic. At the next level, we show that we get exactly the morphic words. This result is then extended to the next level and defines a new class of words enjoying the properties of graphs of the hierarchy. These results are gathered in an article in preparation [Bra].

Chapter 2

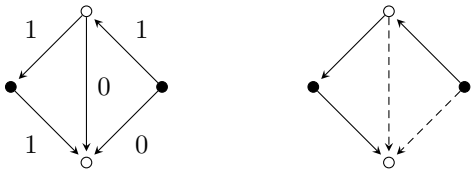
Preliminaries

This chapter introduces the notions used thoughout the thesis. It begins with simple structures, namely words, graphs and trees. Then orders are detailed, in particular ordinals and other scattered orderings. Then we look at the logics used to express properties. This leads us to the logical graph transformations, which in turn defines the pushdown hierarchy.

2.1 Notations and first structures

This thesis is about countable structure enjoying a *finite presentation*, *i.e.* described by a finite quantity of information.

We make a frequent use of representation of graphs with arcs labeled by $\{0, 1\}$. To lighten the pictures, arcs labeled by 1 are drawn with plain lines and arcs labeled by 0 with hashed lines. Exceptions as Figure 5.4 will be clearly labeled.



The *powerset* of a set S is noted $\mathcal{P}(S)$; the closed interval between a and b is noted $[a, b]$; the index i of a sequence $\vec{s}$ is noted $\vec{s}_i$, even in the case when i is an infinite ordinal.

2.1.1 Finite words

The set of *words* over a finite alphabet Σ is noted Σ^*. The *length* of $u \in \Sigma^*$ is noted $|u|$, and the empty word is ε. We say that u is *prefix* of v, noted $u \sqsubseteq v$, if there is $w \in \Sigma^*$ such that $u \cdot w = v$. If $w \neq \varepsilon$, u is a strict prefix of v, noted $u \sqsubset v$. On the contrary, if u, v are incomparable by $\sqsubseteq$, we note $u \perp v$. In any case, the longest common prefix of two words u, v is noted $u \wedge v$.

When Σ is provided with an order $<$, the lexicographic order $<_{\text{lex}}$ on Σ^* is defined by

$$u <_{\text{lex}} v \iff \begin{cases} u \sqsubset v, \text{ or} \\ u = w \cdot a \cdot w' \text{ and } v = w \cdot b \cdot w'' \\ \qquad\qquad \text{where } w, w', w'' \in \Sigma^*, a < b. \end{cases}$$

The collection of regular languages over an alphabet Σ is defined recursively.

$$L := \emptyset \mid \{a\} \text{ where } a \in \Sigma \mid L \cdot L \mid L \cup L \mid L^*$$

2.1.2 Structures

The notion of structure can be seen as a formal logical framework to express mathematical objects. The following definition will be seldom used as such, but it is the general definition of the objects found in this thesis.

A *signature* is a finite set $(R_i)_{i\in I}$ of relation symbols, each symbol R_i having an *arity* $|R_i|$. A *structure* on this signature is a pair (U, ν) where U is a set called *universe* and the valuation ν is a mapping $R_i \mapsto \mathcal{P}(U^{|R_i|})$ called the interpretation of the signature. Commonly $I = [1, k]$, and a structure is written $\langle\, U, (R_1, \ldots, R_k) \,\rangle$ where the valuation is implicit.

Given a universe U and a valuation ν, a binary relation $\nu(R)$ is said

$$\begin{aligned} \text{reflexive if } & \forall x \in U, (x, x) \in \nu(R); \\ \text{symmetric if } & \forall x, y \in U, (x, y) \in \nu(R) \Rightarrow (y, x) \in \nu(R); \\ \text{antisymmetric if } & \forall x, y \in U, (x, y) \in \nu(R) \Rightarrow \neg(y, x) \in \nu(R); \\ \text{transitive if } & \forall x, y, z \in U, (x, y) \in \nu(R) \wedge (y, z) \in \nu(R) \Rightarrow (x, z) \in \nu(R). \end{aligned}$$

Two structures $\langle\, U, \nu \,\rangle$ and $\langle\, U', \nu' \,\rangle$ are *isomorphic*, noted $\langle\, U, \nu \,\rangle \simeq \langle\, U', \nu' \,\rangle$, when there is a bijection π between their signatures which preserves arity, and a bijection κ between U and U' preserving valuation of corresponding relation symbols, *i.e.* for any relation R, $\kappa(\nu(R)) = \nu'(\pi(R))$.

A structure is *countable* if is isomorphic to a structure whose universe is a subset of $\mathbb{N}$. In this thesis, we only consider this kind of structures.

Example 2.1.1. A finite word of length $k > 0$ over an alphabet Σ is a structure of universe $[1, k]$ whose signature consists of one binary relation S (successor) and unary relations $(R_a)_{a\in\Sigma}$. The valuation maps S to all pairs $(i, i+1)$ and each R_a to the set of indexes of letter a; thus $[1, k] = \biguplus_{a\in\Sigma} \nu(R_a)$. The same definition is used for ω-words.

Later on, we will equivalently consider words as structures with binary relation S^*, *i.e.* the reflexive and transitive closure of S, with valuation on all pairs (i, j) such that $i < j$. ▲

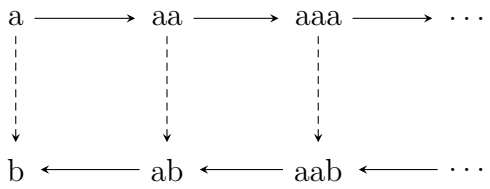


Figure 2.1: Example of a graph : the ladder

2.1.3 Graphs

The graphs we consider are countable, directed and labeled. Let Σ be a finite set called *arc label set*. A labeled graph G, or simply *graph*, is a subset of $V \times \Sigma \times V$ where V is a finite or countable set. An element (s, a, t) of $V \times \Sigma \times V$ is an *arc* of source s, target t and *label* a, and is written $s \xrightarrow[G]{a} t$ or simply $s \xrightarrow{a} t$ if G can be understood from the context. The notation $s \to t$ means "$s \xrightarrow{a} t$ for some a". Note that we do not consider graphs with multiplicity : there can be only one arc labeled by a between s and t.

Let Γ be another finite set called *color set*. A *colored graph* (or *labeled and colored* graph) is a subset of $(V \times \Sigma \times V) \cup (\Gamma \times V)$. More commonly, a graph labeled by Σ is called a Σ-graph, and a Σ-graph colored by Γ is called a Σ, Γ-graph. Note that an equivalent definition would be to allow only one color in $\mathcal{P}(\Gamma)$ per vertex; we prefer the more versatile definition which allows to add or remove colors.

The set of all vertices appearing in G is its *support* V_G, *i.e.* having a color or being linked to an arc. Hence, graphs are always considered up to mute vertices, *i.e.* not appearing in G. This can be a matter of discussion for limit cases, especially when a graph has only one vertex. Therefore we may always suppose that all vertices of the support are colored with a "base color".

A graph is *deterministic* if there are no arcs with the same label that share the same source, *i.e.* for all $a \in \Sigma$, if $s \xrightarrow{a} t$ and $s \xrightarrow{a} t'$ then $t = t'$. The in-degree (resp. out-degree) of a vertex x is the cardinal of the set $\{y \mid y \to x\}$ (resp. $\{y \mid x \to y\}$).

Example 2.1.2. The graph shown in Figure 2.1 is known as the *ladder*. Its support is $V_G = \{aa^i, a^i b\}_{i \in \mathbb{N}}$ and it is defined by

$$G = \{a^i \xrightarrow{1} a^{i+1} \mid i > 0\} \cup \{a^i \xrightarrow{0} a^{i-1}b \mid i > 0\} \cup \{a^i b \xrightarrow{1} a^{i-1}b \mid i > 0\}. \qquad \blacktriangle$$

From a logical point of view, a (colored) graph G can be associated to a structure of universe V_G and of signature $(R_a)_{a \in \Sigma} \cup (R_c)_{c \in \Gamma}$ where R_b has arity 2 or 1 when b belongs respectively to Σ or Γ. The valuation of this structure maps $R_a \mapsto \{(x, y) \mid x \xrightarrow{a} y \in G\}$ for $a \in \Sigma$ and similarly $R_c \mapsto \{x \mid (c, x) \in G\}$ for $c \in \Gamma$. We will often confuse the graph and its associated structure. Two graphs are isomorphic (still noted $\simeq$) if the

corresponding structures are.

A (colored) *path* in a colored graph G is a sequence $\vec{p}$ on G such that if $\vec{p}_i = x \xrightarrow{a} y$ or $\vec{p}_i = (c, y)$ then $\vec{p}_{i+1}$ belongs to $(\{y\} \times \Sigma \times V_G \cup \Gamma \times \{y\}) \cap G$. An uncolored path is such a sequence only of arcs of G. A path is *simple* if the p_i are pairwise distinct.

To consider only the "shape" of the graph, one can consider the *delabeled graph* where all colors are removed and all arcs labels are replaced by the same label. An *unlabeled graph* is a class of graphs isomorphic under the delabeling operation. When the arc label set of a graph is labeled by a singleton, we sometimes confuse this graph and the corresponding unlabeled graph.

2.1.4 Deterministic trees

A vertex r of a graph G is called a *root* when there is a path from r to any other vertex. A graph G is a tree if it has a unique root r such that for any vertex in the graph there exists a unique path from the root r to this vertex. The notions of Σ- and Σ, Γ-graph yield the respective notions of Σ- and Σ, Γ-tree.

For deterministic trees, we may therefore consider a *tree presentation* of the graph as follows. The root is associated to the empty word ε and for each arc $u \xrightarrow{a} v$, we identify v and $u \cdot a$. Formally, a *deterministic tree* over an ordered alphabet Σ is a subset T of Σ^* closed by prefix. If $u \sqsubset v$, we say that u is an *ancestor* of v or equivalently that v is a *descendant* of u. Immediate ancestors and descendants are respectively called father and son. Elements of T are called nodes and nodes without proper descendant are called leaves.

Finally, a *colored deterministic tree* t is a mapping from a deterministic tree T to a finite set of colors Γ. We note $\mathrm{Dom}(t) = T$ the domain of this mapping.

A deterministic tree is *prefix of its leaves* (or is a *prefix tree*) if it is equal to the prefix-closure of its set of leaves. A deterministic tree is a *binary tree* when $\Sigma = \{0, 1\}$. It is said to be *full* if every node has exactly 0 or $|\Sigma|$ sons, and *complete* when every node has $|\Sigma|$ sons . See the complete (uncolored) binary tree in Figure 1.2.

A *branch* of t is a maximal subset B of $\mathrm{Dom}(t)$ such that if $x \in B$, then all ancestors of x and at most one son of x are in B.

2.2 Linear orderings

Linear orderings are the main object of study throughout this thesis. The reason is that they form easily understandable examples with simple properties and nonetheless an arbitrary complexity, for instance in the sense detailed in Section 2.2.2. For a comprehensive introduction to linear orderings, see [Ros82, Roi90].

A *linear ordering*, *total ordering* or simply *ordering*, is a structure whose signature consists in only one binary relation $\leq$ which is reflexive, antisymmetric and transitive.

Nonetheless, when the universe is not a singleton, we will most of the time consider the structure over the associated irreflexive relation $<$; this avoids numerous case distinctions.

The *order type* is an isomorphism class of orders. In our graph vocabulary, it is equivalent to say that an order type is an unlabeled graph of linear orderings. Since we work up to isomorphism, we will often confuse a given ordering with its order type.

The *reverse* operation * is a mapping from an ordering $\langle V, < \rangle$ to the ordering $\langle V, <^* \rangle$ where $x <^* y \iff y < x$. Notable order types include

- the finite orderings noted $\mathbf{0}, \mathbf{1}, \ldots, \mathbf{k}, \ldots$;
- the usual ordering of $\mathbb{N}$, noted ω;
- the ordering of $\mathbb{N}$ in reverse order, ω^*;
- the usual ordering of $\mathbb{Z}$, noted ζ;
- the usual ordering of $\mathbb{Q}$, noted η;
- the usual ordering of $\mathbb{R}$.

This thesis is dedicated to countable structures, so we will never consider the ordering of $\mathbb{R}$.

The *subordering* relation $\preceq$ is defined on order types by $\alpha \preceq \beta$ iff there is an ordering of type β which has a restriction of type α. This relation is extended to orderings when the order types are similarly ordered. For instance we have $\omega \preceq \zeta \preceq \eta$, but neither $\omega \preceq \omega^*$ nor $\omega^* \preceq \omega$. As we will see later, an order not having η as a suborder is called scattered. An *interval* I of L is a restriction of L to a subset where if x, y are elements of I and there is z in L such that $x < z < y$, then z is an element of I.

Let Γ be a finite set of colors. A *colored ordering* of an ordering $\langle V, < \rangle$ is a mapping $V \mapsto \Gamma$. As hinted by the notations, a (resp. colored) ordering can be associated to a (resp. colored) graph. Consequently, we deliberately confuse the objects and will indifferently use the structural or graph notation.

An *ω-word* or (mono-)infinite word is a colored ordering of type ω.

There is an available arithmetics on linear orderings : more precisely we use addition and multiplication. Formally, these noncommutative operations are defined as follows. See also [Ros82].

$$\begin{aligned}
\langle U, <_U \rangle + \langle V, <_V \rangle &= \langle U \uplus V, <_{U,V} \rangle \\
\text{where } x <_{U,V} y \text{ iff} \quad & x, y \in U \text{ and } x <_U y \\
\text{or} \quad & x, y \in V \text{ and } x <_V y \\
\text{or} \quad & x \in U \text{ and } y \in V. \\
\langle U, <_U \rangle . \langle V, <_V \rangle &= \langle V \times U, <_{\text{lex}} \rangle . \\
\text{where } (u, v) <_{\text{lex}} (u', v') \text{ iff} \quad & u <_U u' \\
\text{or} \quad & u = u' \wedge v <_V v'.
\end{aligned}$$

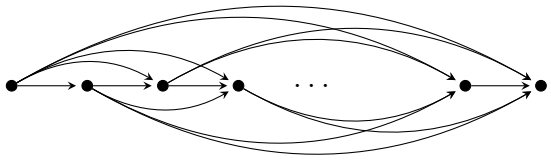


Figure 2.2: The graph representation of the ordinal $\omega + 2$.
Graph representations of ordinals are hardly readable due to many arcs. The lighter notion of covering graphs will be described in Section 3.4.

2.2.1 Ordinals

A particular kind of linear orderings are the *well-orderings*, which have the following equivalent properties :

- each nonempty subset has a smallest element;
- there is no infinite strictly decreasing sequence.

An *ordinal* is the order type of some well-ordering. However, as noted in Section 2.1.3, we often will identify the order type with the ordering; for instance, the sentence "the ordinal α belongs to the class X" means actually "there is an ordering of order type α in the class X".

All ordinals are themselves well-ordered by the subordering relation. For any ordinal α, the set of ordinals greater than α has a smallest element, which we call the *successor* of α. The reverse operation, the *predecessor*, is not defined everywhere. The set where predecessor is defined is naturally called the set of successors, or ordinals *of the first kind* in old literature; its complementary is called the set of limit ordinals, or ordinals *of the second kind*. The supremum $\sup(X)$ of a set X of ordinals is the smallest ordinal greater than each ordinal in X.

The set-theoretical (or Von Neumann) approach defines each ordinal as the set of smaller ordinals. For instance, $\mathbf{0} = \emptyset$ and $\mathbf{1} = \{\emptyset\}$. We borrow this encoding to define the *canonical graph of* α as the ordinal graph where vertices are exactly ordinals smaller than α. The fact that we consider graphs up to mute vertices is not a problem, because the equality relation is always implicit; in particular $\mathbf{1} \neq \mathbf{0}$, because there is one non-mute vertex in $\mathbf{1}$.

Arithmetics

Arithmetic operations can be defined in two ways : either by transfinite iteration or by giving an isomorphic structure. The very first operation available is the successor relation. Then addition is the transfinite iteration of successor, multiplication is the iteration of

addition, and exponentiation is the iteration of multiplication :

$$\begin{aligned} \alpha + 0 &= \alpha, \\ \alpha + (\beta + 1) &= (\alpha + \beta) + 1, \\ \text{for limit } \lambda, \alpha + \lambda &= \sup_{\beta<\lambda}(\alpha + \beta). \end{aligned}$$

$$\begin{aligned} \alpha.0 &= 0, & \alpha^0 &= 1, \\ \alpha.(\beta + 1) &= \alpha.\beta + \alpha, & \alpha^{\beta+1} &= \alpha^\beta.\alpha, \\ \text{for limit } \lambda, \alpha.\lambda &= \sup_{\beta<\lambda} \alpha.\beta. & \alpha^\lambda &= \sup_{\beta<\lambda} \alpha^\beta. \end{aligned}$$

This definition of addition and multiplication matches the structural definition for general linear orders given above. A similar direct definition is available for exponentiation [Ros82, Exercise 3.45]. The *reverse lexicographic order* means $\vec{s} <_{\text{rlex}} \vec{t}$ iff $\exists \alpha$ such that $\vec{s}_\alpha < \vec{t}_\alpha$ and for all $\delta > \alpha$, $\vec{s}_\delta = \vec{t}_\delta$.

Proposition 2.2.1. *β^α is isomorphic to the set of α-sequences of β where finitely many elements are non-zero, ordered by reverse lexicographic order.*

Proof. When $\alpha = 0$, $\beta^0 = \mathbf{1}$, and there is indeed only one empty sequence. When $\alpha = \gamma+1$, $\beta^\alpha = \beta^\gamma.\beta$ is the cartesian product $\beta \times \beta^\gamma$ by lexicographic order, or equivalently $\beta^\gamma \times \beta$ in reverse lexicographic order. By induction β^γ is the set of (finitely non-zero) γ-sequences of β in reverse lexicographic order, which yields the result.

When α is a limit ordinal, let $\vec{x}$ be an α-sequence where finitely many elements are non-zero. In particular, there is a smallest index γ such that $\vec{x}_{\gamma'} = 0$ for all $\gamma' \geq \gamma$. Then $\vec{x}$ can be mapped to an ordinal smaller than ω^γ. This mapping is an isomorphism from (finitely non-zero) α-sequences to ω^α. □

When $\beta = \omega$, the following definition is easier to use. This is the form we will adopt later on.

Corollary 2.2.2. *ω^α is isomorphic to the set of finite decreasing sequences of α in lexicographic order.*

There is a similar form for some scattered orders; see Section 3.2.

We use the *Knuth notation*[Knu76] to express more complex operations. The operation $\uparrow^1$ is the exponentiation.

$$\begin{aligned} \alpha \uparrow^{n+1} 0 &= 1; \\ \alpha \uparrow^{n+1} (\beta + 1) &= \alpha \uparrow^n (\alpha \uparrow^{n+1} \beta); \\ \alpha \uparrow^{n+1} \lambda &= \sup_{\beta<\lambda} \alpha \uparrow^{n+1} \beta \text{ for limit } \lambda. \end{aligned}$$

In this thesis we use the case $n = 2$ and note $\uparrow^2 = \uparrow\uparrow$. It will mostly be used to express ordinals of the form

$$\omega \uparrow\uparrow k = \underbrace{\omega^{\omega^{\cdots^{w}}}}_{k}.$$

Indeed, this thesis is restricted to ordinals smaller than $\varepsilon_0 = \omega \uparrow\uparrow \omega$.

Remark 2.2.3. This notation is not the generalization of addition, multiplication and exponentiation, because the iteration is done on the right side. If we chose to follow this definition, the operation $\uparrow\uparrow'$ succeding to exponentiation would rather be defined by

$$\begin{aligned} \alpha \uparrow\uparrow' 0 &= 1; \\ \alpha \uparrow\uparrow' (\beta+1) &= (\alpha \uparrow\uparrow' \beta)^\alpha; \\ \alpha \uparrow\uparrow' \lambda &= \sup_{\beta<\lambda} \alpha \uparrow^{n+1} \beta \text{ for limit } \lambda, \end{aligned}$$

which would give $\omega \uparrow\uparrow' 2 = \omega^\omega$, but $\omega \uparrow\uparrow' 3 = (\omega^\omega)^\omega = \omega^{\omega.2}$ and $\omega \uparrow\uparrow' \omega = \omega^{\omega^\omega}$. ■

This notation is closely related to the family of Veblen functions [Veb08]. Let φ_0 be a continuous increasing function, *i.e.* an increasing function such that $\lim_n \varphi_0(\alpha_n) = \varphi_0(\lim_n \alpha_n)$. For $\alpha > 0$, φ_α is the continuous increasing function enumerating common fixed points of $(\varphi_\beta)_{\beta<\alpha}$. In particular, the case $\varphi_0 : x \mapsto \omega^x$ yields a family of functions known as the Veblen hierarchy, and $\varphi_1(\alpha) = \omega \uparrow\uparrow \alpha$. These functions define in fact the ε-numbers.

Cantor normal form

Cantor states a fundamental tool for ordinal analysis.

Theorem 2.2.4 ([Can97])**.** *Let α be an ordinal. Then α can be uniquely written in the form*

$$\omega^{\gamma_1}.c_1 + \cdots + \omega^{\gamma_k}.c_k$$

*where $\gamma_1 > \cdots > \gamma_k$ and $k, c_1, \ldots, c_k$ are natural numbers (*i.e. *finite ordinals).*

We will call this form the *reduced Cantor normal form* (*RCNF*), denoted by $\hat{\alpha}$. We call *Cantor normal form* (*CNF*) the following version

$$\alpha = \omega^{\gamma_1} + \cdots + \omega^{\gamma_k}$$

where $\gamma_1 \geq \cdots \geq \gamma_k$.

Since this thesis is restricted to ordinals smaller than ε_0, we have also the additional property $\alpha > \gamma_1$, which allows induction.

2.2.2 Scattered orderings and Hausdorff rank

A linear order is *dense* if for each $x < y$, there is a z such that $x < z < y$. There are only five countable dense order types, depending on whether there is an upper and/or lower bound. They are $\mathbf{1}, \eta, \mathbf{1}+\eta, \eta+\mathbf{1}$ and $\mathbf{1}+\eta+\mathbf{1}$.

A linear order is *scattered* if it does not contain any infinite dense subordering. Ordinals are a particular case of scattered linear orders. However, scattered orders are not

necessarily well-orderings; consider for instance ζ or $\omega + \omega^*$. For a detailed presentation, we refer the reader to [Ros82].

The following result shows that all linear orders are combinations of these two kinds.

Theorem 2.2.5 (Hausdorff [Hau08])**.** *Any linear ordering L is a dense sum of scattered linear orderings; that is, there is a dense linear ordering D and a map h from D to scattered orderings such that $L = \sum_{i \in D} h(i)$.*

In this section we focus on countable scattered orders. A more constructive characterization is provided by Hausdorff Theorem which also gives a measure of the *complexity* of such orders. From now on, we only consider countable scattered orders.

Theorem 2.2.6 (Hausdorff [Hau08])**.** *A countable linear order is scattered if and only if it belongs to $\bigcup_\alpha V_\alpha$ where*

$$\begin{aligned} V_0 &= \{\mathbf{0}, \mathbf{1}\} \\ V_\beta &= \left\{ \sum_{i \in \mathbb{Z}} L_i \mid \forall i, L_i \in \bigcup_{\alpha < \beta} V_\alpha \right\} \end{aligned}$$

The *Hausdorff rank* of a scattered order L, written $r(L)$ (or sometimes $VD(L)$ in the literature), is the smallest α such that L belongs to V_α. For instance, we have $r(\zeta) = r(\omega) = 1$ and $r(\omega + \omega^*) = 2$.

As the classes V_α are not closed under finite sum, we do not have in general that $r(A + B) = \max(r(A), r(B))$. It is natural to consider W_α, the closure under finite sums of V_α (*i.e.* $L \in W_\alpha$ iff $L = \sum_{i \in [1,m]} L_i$ for some $L_1, \ldots, L_m \in V_\alpha$). The associated notion of rank, called $\sim$-rank and written $\tilde{r}(L)$, is the smallest ordinal α such that $L \in W_\alpha$. This definition can be found in [KRS05] under the denomination $VD_*(L)$. As $V_\alpha \subseteq W_\alpha \subseteq V_{\alpha+1}$, we have $\tilde{r}(L) \leq r(L) \leq \tilde{r}(L)+1$. Along this thesis, we will mostly use this alternate version of the Hausdorff rank.

For instance, the following proposition states that the $\sim$-rank of the ordinal ω^α is α. More generally if α is written $\sum_{i=1}^k \omega^{\alpha_i}$ in Cantor's normal form then $\tilde{r}(\alpha) = \alpha_1$.

Proposition 2.2.7. *For any ordinal α, $\tilde{r}(\omega^\alpha) = r(\omega^\alpha) = \alpha$.*

Proof. To show that $\tilde{r}(\omega^\beta) = r(\omega^\beta)$, we only need to show that if $\omega^\beta \in W_\alpha$ then $\omega^\beta \in V_\alpha$. Assume that for some β, ω^β belongs to W_α. By definition, $\omega^\beta = \delta_1 + \ldots + \delta_n$ with $\delta_i \in V_\alpha$ for all $i \in [1, n]$. There exists $j \in [1, n]$ s.t. $\delta_j = \omega^\beta$. Otherwise, from the definition of ω^β, we would have $\delta_j \leq \omega^{\gamma_i} \cdot k_i$ for some $\gamma_i < \beta$ and $k_i < \omega$. We would have $\omega^\beta \leq \omega^{\max_i \gamma_i} \cdot \sum_i k_i < \omega^\beta$ which brings the contradiction.

A straightforward transfinite induction on α shows that for all ordinal β, $\omega^\beta \in V_\alpha$ if and only if $\beta \leq \alpha$. □

The following facts are useful properties on scattered orders.

Proposition 2.2.8. *Let $(L_i)_{i\in\mathbb{Z}}$ be a family of scattered orders and let α be an ordinal :*

1. *if $L_i \preceq L_j$ then $\tilde{r}(L_i) \leq \tilde{r}(L_j)$;*
2. *for all $n \geq 1$, there exists $j \in [1,n]$ s.t. $\tilde{r}(\sum_{i\in[1,n]} L_i) = \tilde{r}(L_j)$;*
3. *$\tilde{r}(\sum_{i\in\mathbb{Z}} L_i) \geq \alpha$ iff either there exists $i \in \mathbb{Z}$ s.t. $\tilde{r}(L_i) \geq \alpha$ or for all $\alpha' < \alpha$, there exist infinitely many i s.t. $\tilde{r}(L_i) \geq \alpha'$;*
4. *$\tilde{r}(\sum_{i\in\mathbb{Z}} L_i) \leq \alpha$ iff for all $i \in \mathbb{Z}$, $\tilde{r}(L_i) \leq \alpha$ and there are only finitely many i such that $\tilde{r}(L_i) = \alpha$.*

Remark 2.2.9. The two conditions of property 3 are not exclusive. Take for instance

$$\begin{aligned} L_0 &= (\omega^\omega)^*, \\ \text{for } k>0, \quad L_k &= \omega^k \\ \text{and} \quad L_{-k} &= 0. \end{aligned}$$

Then $\tilde{r}(L_0) = \tilde{r}(\sum_{\mathbb{Z}} L_i) = \omega$, and $\tilde{r}(\omega^k) = k$. ■

Remark 2.2.10. It is not difficult to see that we can "inverse $\mathbb{Z}$", *i.e.* that

$$\tilde{r}(\sum_{i\in\mathbb{Z}} L_i) = \tilde{r}(\sum_{i\in\omega+\omega^*} L_i).$$

We may therefore replace $\mathbb{Z}$ by $\omega+\omega^*$ in the previous proposition. This fact will be used in Proposition 4.4.7. ■

Proof. Property 1. Let L and L' be two scattered orders s.t. $L \preceq L'$. From [Ros82, Lem. 5.14], $r(L) \leq r(L')$. Assume that $\tilde{r}(L) = \alpha$. This means that L is equal to the finite sum $\sum_{i\in[1,n]} L_i$ where for all $i \in [1,n]$, $r(L_i) \leq \alpha$. As $L' \preceq L$, L' is equal to a finite sum $\sum_{i\in[1,n]} L'_i$ where for all $i \in [1,n]$, $L'_i \preceq L_i$. Hence for all $i \in [1,n]$, $r(L_i) \leq r(L_i) \leq \alpha$. This shows that $\tilde{r}(L) \leq r(L) = \alpha$.
Property 2. This property can be seen as a particular case of [KRS05, 4.2]. More simply, for $i \in [1,n]$, $\tilde{r}(L_i) \leq \alpha$ by property 1. If $\tilde{r}(L_i) < \alpha$ for all i, then $\sum_{[1,n]} L_i$ can be written as a finite sum of (Hausdorff) orders strictly smaller than α and therefore $\tilde{r}(\sum_{[1,n]} L_i) < \alpha$.
Property 3. [$\Rightarrow$] Suppose there is an $\alpha' < \alpha$ such that $\tilde{r}(L_i) < \alpha'$ for all i. Then each L_i is a finite sum of orders of (Hausdorff) rank smaller than α'. This means $L = \sum_{i\in\mathbb{Z}} L_i$ is in $V_{\alpha'}$ and therefore L has Hausdorff rank α'. This would mean $\tilde{r}(L) \leq \alpha' < \alpha$.

So for each $\alpha' < \alpha$, $\{i \mid \tilde{r}(L_i) \geq \alpha'\}$ is nonempty. If it is always infinite, the proof is done. Otherwise, there is $\alpha' < \alpha$ such that this set is finite. Let I be a finite interval containing it. If we write

$$L = \sum_{i<I} L_i + \sum_{i\in I} L_i + \sum_{i>I} L_i,$$

since for $i \notin I$ we have $\tilde{r}(L_i) < \alpha'$ then $\tilde{r}(\sum_{i<I} L_i) < \alpha$ and $\tilde{r}(\sum_{i>I} L_i) < \alpha$ by the above paragraph. By property 2, this means $\tilde{r}(\sum_{i<I} L_i) = \tilde{r}(L)$, and by property 2 again, this means there is a $i \in I$ such that $\tilde{r}(L_i) = \tilde{r}(L) \geq \alpha$.
[$\Leftarrow$] If there is i such that $\tilde{r}(L_i) \geq \alpha$, then by property 1, $\tilde{r}(L) \geq \tilde{r}(L_i) \geq \alpha$. We suppose only the second property is fulfilled. If α is limit, for each $\alpha' < \alpha$ there is a $L' \preceq L$ such that $\alpha' \leq r(L')$. So $\alpha' < r(L)$ for each $\alpha' < \alpha$ and $\alpha \leq r(L)$. Since $r(L) \leq \tilde{r}(L) + 1$ and α is limit, $\tilde{r}(L) \geq \alpha$. If $\alpha = \alpha' + 1$, there is infinitely many i such that $\tilde{r}(L_i) = \alpha'$. So $\tilde{r}(L) \geq \alpha'$, and if actually $\tilde{r}(L) = \alpha'$, there would only be finitely many such i by property 4, part [$\Rightarrow$]. So $\tilde{r}(L) > \alpha'$ and $\tilde{r}(L) \geq \alpha$.
Property 4. [$\Rightarrow$] Property 1 ensures that $\tilde{r}(L_i) \leq \tilde{r}(L) \leq \alpha$. Suppose $I = \{i \mid \tilde{r}(L_i) = \alpha\}$ is infinite. Since $\sum_I L_i \preceq L$, by property 1, $\tilde{r}(\sum_I L_i) = \alpha$. This means $\sum_I L_i$ is a finite sum of orders of (Hausdorff) rank smaller or equal to α. One of these orders M is such that there is an infinite $I' \subseteq I$ with $\sum_{I'} L_i \preceq M$. By the definition of the Hausdorff rank, we may write $\sum_{i \in I'} L_i \preceq \sum_{j \in \mathbb{Z}} M_i$ where $r(M_j) < \alpha$ for each $j \in \mathbb{Z}$. Consider any i which is not an extremum of I (i is neither the first nor the last element). Then there is j_i^- and j_i^+ such that $L_i \preceq \sum_{j=j_i^-}^{j_i^+} M_i$. So L_i is a finite sum of orders of rank $< \alpha$: $\tilde{r}(L_i) \leq \max_{j \in [j_i^-, j_i^+]} r(M_j) < \alpha$, which is a contradiction.
[$\Leftarrow$] If $\tilde{r}(L) > \alpha$, since there is no i such that $\tilde{r}(L_i) > \alpha$, by property 3 part [$\Rightarrow$], there would be infinitely many i such that $\tilde{r}(L) = \alpha$, which is not the case. So $\tilde{r}(L) \leq \alpha$. □

2.2.3 Orders in a deterministic tree

Whenever we talk about a deterministic tree, we always may assume that the alphabet is ordered. Thus, the nodes are also ordered by the lexicographic ordering. The *frontier* of a deterministic tree $\mathbf{Fr}(t)$ is the (colored) order of its leaves by lexicographic ordering. We sometime say that a node u is to the left (or right) of a node v to say that $u <_{\text{lex}} v$ (resp. $v <_{\text{lex}} u$).

Other orders in deterministic trees that are worth of interest include

- the lexicographic order $\langle \operatorname{Dom}(t), <_{\text{lex}} \rangle$ on the whole tree, not just on leaves.
- the Kleene-Brouwer ordering $\langle \operatorname{Dom}(t), <_{\text{KB}} \rangle$ as seen in [Rog87], where

$$x <_{\text{KB}} y \iff y \sqsubset x \vee (x \perp y \wedge x <_{\text{lex}} y).$$

The equivalence of these three orders with regard to Hausdorff rank will be established in Proposition 4.4.8, and needs the powerful Proposition 4.4.3. For the time being, let us satisfy with the following result.

Proposition 2.2.11. *If t is a det. prefix tree which frontier is a well-ordering, then* $\operatorname{Dom}(t)$ *is well-ordered by* $<_{\text{lex}}$.

Proof. By contraposition, suppose there is an infinite strictly decreasing sequence of nodes of t. In particular, there is an infinite strictly decreasing subsequence of strictly increasing lengths. If $x <_{\text{lex}} y$ and $|x| > |y|$, then $y \not\sqsubseteq x$ and for any y', x' such that $y \sqsubseteq y'$, $x \sqsubseteq x'$, we have $x' <_{\text{lex}} y'$. Since t is prefix, there is therefore an infinite strictly decreasing sequence of leaves in t, so t cannot yield a well-ordering. □

In contrast, as soon as t is an infinite tree, it has an infinite branch which has order type ω^* by $<_{\text{KB}}$; so the Kleene-Brouwer ordering is not a well-ordering unless t is finite.

2.3 Logic

2.3.1 First-order logic

We fix a countable set $\mathcal{V}_1$ of *first-order variables* $x, y, z, \ldots$. Let $(R_i)_{i \in I}$ be a signature. Formulæ over this signature are of the form

$$\varphi := \top \mid \varphi \wedge \varphi \mid \neg\varphi \mid \exists x\, \varphi \mid R_i(x_1, \ldots, x_{|R_i|})$$

where $x_1, \ldots, x_{|R_i|} \in \mathcal{V}_1$. Here $\top$ is the "true" constant, $\wedge$ is the logical conjonction, $\neg$ is the negation, $\exists$ is the existential quantifier. It is well-known that additional operators can be encoded, namely the constant $\bot$ ("false"), disjonction $\vee$, the implication $\Rightarrow$ and the universal quantifier $\forall$.

There is *a priori* no relation of equality between variables; however, it is often implicit in first-order logic, and will reveal itself useless in monadic logic.

The set of free variables of φ is the set of variables appearing but not quantified in φ. We note as usual $\varphi(x_1, \ldots, x_n)$ when the set of free variables of φ is $\{x_1, \ldots, x_n\}$. A formula without free variables is called a *closed formula* or statement. We note $S \models \varphi$ when the structure S satisfies a closed formula φ. More generally, for a given formula $\varphi(x_1, \ldots, x_n)$, we note $S \models \varphi[a_1, \ldots, a_n]$ when the structure S satisfies the formula φ where the variable x_i is interpreted as the element a_i.

The *first-order theory* of a structure S is the set of closed formulæ satisfied by S. It is said to have a decidable first-order theory when this set is recursive, as seen in Section 2.3.3.

Example 2.3.1. The determinism on Σ-graphs can be checked with the following formula when the equality relation is allowed.

$$\bigvee_{a \in \Sigma} \forall x, y, z \left((x \xrightarrow{a} y \wedge x \xrightarrow{a} z) \Rightarrow y = z \right) \qquad \blacktriangle$$

Example 2.3.2. The first-order logic can express the property that a given graph of label

set $\{<\}$ is an order graph.

$$\begin{array}{rc} \text{strict order :} & \left\{ \begin{array}{c} \forall x, y\ (\neg(x \overset{<}{\to} y \wedge y \overset{<}{\to} x)) \\ \forall x, y, r\ ((x \overset{<}{\to} y \wedge (y \overset{<}{\to} r) \Rightarrow x \overset{<}{\to} r) \end{array} \right. \\ \text{total order :} & \forall x, y\ (x \overset{<}{\to} y \vee y \overset{<}{\to} x \vee x = y) \end{array}$$ ▲

2.3.2 Monadic second-order logic

The *monadic logic* extends the first-order logic with new variables interpreted as sets, and a new relation $\in$ for membership. Let $\mathcal{V}_2$ be a new set of *second-order variables*, noted with uppercase letters. A formula φ is defined by

$$\varphi := R_i(x_1, \ldots, x_{|R_i|}) \mid x \in X \mid \varphi \wedge \varphi \mid \neg\varphi \mid \exists x\, \varphi \mid \exists X\, \varphi$$

where $x, x_1, \ldots, x_{|R_i|} \in \mathcal{V}_1$ and $X \in \mathcal{V}_2$.

Empty set $\emptyset$, subset relation $\subseteq$, union $\cup$, intersection $\cap$, complementation $\setminus$ are all naturally embedded in MSO-logic, as is second-order equality. By the means of a formula expressing singletons,

$$\text{singleton}(X) := \exists x \in X\ (\forall Y\ (x \in Y \Rightarrow X \subseteq Y)),$$

first-order equality is also naturally expressible, which solves the problem of having an equality relation in the signature or not.

Example 2.3.3. Adding the following MSO-formula to Example 2.3.2 characterizes structures of signature $\{<\}$ which are well-orderings.

$$\text{well order :} \qquad \forall X \neq \emptyset,\ \exists x(x \in X \wedge \forall y(y \in X \Rightarrow (x \overset{<}{\to} y \vee x = y))) \qquad ▲$$

The set of all monadic formulæ satisfied by a structure G is called the *monadic theory* of G and is noted $\text{MTh}(G)$.

2.3.3 Decidability

An important problem of logicians is to decide whether a formula is satisfied by a given structure.

Question 2.3.4. "Model-Checking Decidability" : For a given structure, find an algorithm taking as input a monadic formula and outputting whether the structure satisfies the formula or not.

On orders, the earlier result is by Büchi [Büc62]. He introduces the notion of (now called) *Büchi automaton*, which is a finite-state automaton accepting an infinite sequence iff there is a run which visits at least one of the final state infinitely often. He then proves that each monadic second-order formula can be effectively converted into a Büchi automaton.

Theorem 2.3.5 ([Büc73])**.** *The monadic theory of ω is decidable.*

Later on, the possibility of transfinite runs was added to the automata. Using Theorem 3.4.2 explained in Section 3.4, we get the following theorem. These results are summed up in [Büc73].

Theorem 2.3.6 ([Bü65])**.** *The monadic theory of each countable ordinals is decidable.*

Shelah [She75] (see also Gurevich [Gur85]) developped an "automata-free" method, now called the compositional method. For a finite sequence $\bar{k}$ of integers, it defines the notion of $\bar{k}$-type of a structure, which is expressive enough to say whether a given formula is satisfied in the structure or not. The composition theorem states that it is possible to compute the $\bar{k}$-type of a sum with the types of the summands. With this tools, the two results above are restated. A useful introduction to this method appears in [Tho97a], as well as a comparison between Büchi's and Shelah's methods.

For other orders, the theory of η was proved to be decidable. By using Theorem 2.2.5, it is possible to prove that the theory of all countable orders, *i.e.* the set of formulæ that are true for any order, is decidable. Some results also appeared for uncountable structures. Büchi proved that the theory of ω_1, the first uncountable ordinal, in decidable, whereas Shelah proved that the theory of the real order was not.

On other structures, the most famous case is the complete binary tree, already seen in Figure 1.2. The proof employs automata on infinite trees.

Theorem 2.3.7 ([Rab69])**.** *The monadic theory of the complete binary tree is decidable.*

This theorem was generalized by the regular, then prefix-recognizable graphs, and later by the whole pushdown hierarchy presented in Section 2.5. It may be defined by operations, *i.e.* graph transformations preserving decidability of MSO logic; some of them are shown in Section 2.4. For more on this subject, see for instance the recent survey [BCL07].

In the other way around, a classical example of undecidable monadic theory is the two-dimensional grid, which can be presented as the graph of support $\mathbb{N} \times \mathbb{N}$ and arcs

$$\{(i,j) \xrightarrow{a} (i+1,j), (i,j) \xrightarrow{b} (i,j+1) \mid i,j \in \mathbb{N}\}.$$

The grid is the configuration graph of an automaton with two unary pushdown stacks. It has the same expressive power as a Turing machine; therefore properties like reachibility are not decidable.

Proposition 2.3.8 ([See91])**.** *The grid has an undecidable monadic theory.*

2.4 Graph transformations

This section presents some of the transformation we will use through this thesis. They are here classified by increasing "impact" on the graph. The graph interpretations preserve vertices, but change arcs; a particular case is the graph coloring, which also preserves arcs. The graph expansions extend or completely change the set of vertices.

In regard of Section 2.3.3, it is important to note that all these transformations are *MSO-compatible*, *i.e.* they preserve decidability of the monadic theory.

2.4.1 Graph interpretations

These operations re-arrange arcs between existent vertices without adding new vertices. They are presented in decreasing strength. Later in Remark 4.1.4, the TWA-interpretation will be mentioned.

The generic form of interpretation is defined by $\mathcal{I} = \{\varphi_a\}_{a\in\Sigma} \cup \{\varphi_c\}_{c\in\Gamma}$, where $\{\varphi_a\}_{a\in\Sigma}$ is a set of (to be defined) binary formulæ and $\{\varphi_c\}_{c\in\Gamma}$ is a set of unary formulæ . The interpretation of a graph G is then

$$\mathcal{I}(G) = \{x \xrightarrow{a} y \mid G \models \varphi_a(x,y)\} \cup \{(c,x) \mid G \models \varphi_c(x)\}$$

An example of very strong interpretation is the set interpretation developped in [CL07]. In this case, each formula φ_a is a monadic formula with two second-order free variables. This interpretation is not MSO-compatible, but if G has a decidable weak monadic logic, then $\mathcal{I}(G)$ has a decidable first-order logic.

Monadic interpretation

A *MSO-interpretation* $\mathcal{I}$ is a finite set $\{\varphi_a\}_{a\in\Sigma} \cup \{\varphi_c\}_{c\in\Gamma}$ of monadic second-order (MSO) formulæ with two or one free individual variables. The interpretation of a graph $G \subseteq V \times \Sigma \times V$ is the graph $I(G)$ defined by

$$\begin{aligned} I(G) \;=\; & \{x \xrightarrow{a} y \mid x,y \in V \wedge G \models \varphi_a(x,y)\} \\ & \cup \{(c,x) \mid x \in V \wedge G \models \varphi_c(x)\}. \end{aligned}$$

A *MSO coloring* is a particular case of interpretation where binary formulæ are in the form $\varphi_a(x,y) = x \xrightarrow{a} y$, *i.e.* an interpretation which do not change the labeling of the graph.

Example 2.4.1. The frontier of a deterministic tree is a MSO-interpretation of this tree.

On a binary tree (arc labels $\{0,1\}$), let $\mathcal{I} = \{\varphi_<\} \cup \{\varphi_c\}_{c\in\Gamma}$.

$$\begin{array}{rrcl} & \varphi_<(x,y) & := & \neg\exists x'(x \to x') \wedge \neg\exists y'\ (y \to y') \\ & & & \wedge\exists z\ (z \xrightarrow{0(0+1)^*} x \wedge z \xrightarrow{1(0+1)^*} y) \\ \text{and for any } c, & \varphi_c(x) & := & (c,x) \wedge \neg\exists x'(x \to x'). \end{array}$$

This example is easily extended to the case of any alphabet Σ. ▲

Inverse rational mapping

Another particular case of monadic interpretation is inverse rational mapping. For a given alphabet Σ, we use the disjoint alphabet $\bar{\Sigma}$ to read the arcs backwards. For a rational language L over $\Sigma \cup \bar{\Sigma}$, the formula $p \xrightarrow{L} q$ is defined inductively.

$$\begin{array}{rcl} p \xrightarrow{\emptyset} q & \text{iff} & p \neq p \text{ (or any false formula)} \\ p \xrightarrow{\{\varepsilon\}} p & & \\ p \xrightarrow{\{c\}} p & \text{iff} & (c,p) \\ p \xrightarrow{\{a\}} q & \text{iff} & p \xrightarrow{a} q, \text{ for any } a \in \Sigma \\ p \xrightarrow{\{\bar{a}\}} q & \text{iff} & q \xrightarrow{a} p, \text{ for any } a \in \Sigma \\ p \xrightarrow{L+L'} q & \text{iff} & p \xrightarrow{L} q \vee p \xrightarrow{L'} q \\ p \xrightarrow{L.L'} q & \text{iff} & \exists r(p \xrightarrow{L} r \wedge r \xrightarrow{L'} q) \\ p \xrightarrow{L^\star} q & \text{iff} & \forall X(p \in X \wedge Closed_L(X) \Rightarrow q \in X) \\ & \text{where} & Closed_L(X) \equiv \forall x,y \in X((x \in X \wedge x \xrightarrow{L} y) \Rightarrow y \in X). \end{array}$$

An *inverse rational mapping* is a monadic interpretation $\mathcal{I} = \{\varphi_a\}$ where each formula is of the type $\varphi_a(x,y) = x \xrightarrow{h(a)} y$ or $\varphi_a(x) = x \xrightarrow{h(a)} x$ and $h(a)$ is a rational language. In this case $\mathcal{I}$ is also noted h^{-1}.

Example 2.4.2. The formula

$$\exists z\ (z \xrightarrow{0(0+1)^*} x \wedge z \xrightarrow{1(0+1)^*} y)$$

used in Example 2.4.1 can be translated into the more compact

$$x \xrightarrow{(\bar{0}+\bar{1})^*\bar{0}1(0+1)^*} y.$$ ▲

As this example hints, it was shown that the inverse rational mapping is not a strong constraint on deterministic trees.

Proposition 2.4.3 ([Car06, Prop. 3.2.1]). *For any Σ, Γ and monadic interpretation $\mathcal{I}$, there is a monadic coloring μ and a rational interpretation h^{-1} such that for all deterministic Σ, Γ-tree t,*

$$\mathcal{I}(t) = h^{-1}(\mu(t)).$$

There are even more specific variants of inverse rational mapping. We will see in Section 5.1.2 a version called *deterministic* rational mapping where the underlying automaton can only branch when looking at colors. It was designed to preserve determinism of graphs.

An even more narrow case is when the language L_a is a finite language for each a. This case is naturally called an *inverse finite mapping*.

2.4.2 Graph expansions

These transformations map a graph G towards a tree or tree-like structure based on G. They are also the source of new vertices for structure of the hierarchy.

Monadic transduction

The *MSO-transduction* (see the survey [Cou94] for more details) aims at making monadic interpretations more flexible by adding some vertices. Formally, when K is a finite alphabet, a K-copying operation associates to G a $(\Sigma \cup K)$-labeled graph G' :

$$\begin{aligned} V_{G'} &= V_G \cup (V_G \times K) \\ G' &= G \cup \{x \xrightarrow{k} (x,k) \mid k \in K\}. \end{aligned}$$

A *MSO-transduction* $\mathcal{T} = (K, \mathcal{I})$ is a K-copying operation followed by an MSO interpretation.

Unfolding

The *unfolding* of a graph from a given vertex is the tree of all paths in the graph from this vertex.

Formally, the unfolding $\mathrm{Unf}(G, r)$ of a graph G from a vertex $r \in V_G$ is the tree T s.t. for all $a \in \Sigma$, $\pi \xrightarrow{a} \pi' \in T$ if and only if π and π' are two paths in G starting from r and $\pi' = \pi \cdot (s \xrightarrow{a} t)$. Moreover for any color $c \in \Gamma$, $(c, \pi) \in T$ if and only if π is a path in G starting with r and ending in t with $(c, t) \in G$.

When the graph has exactly one root, the shortcut $\mathrm{Unf}(G)$ is used to designate the unfolding from this root. A classic example is the unfolding of the ladder (presented in Figure 2.1), shown in Figure 2.3.

The MSO-compatibility of this operation is a particular case of the Muchnik theorem (Th. 2.4.5 below).

Theorem 2.4.4 ([CW98]). *Unfolding from a MSO-definable vertex is MSO-compatible.*

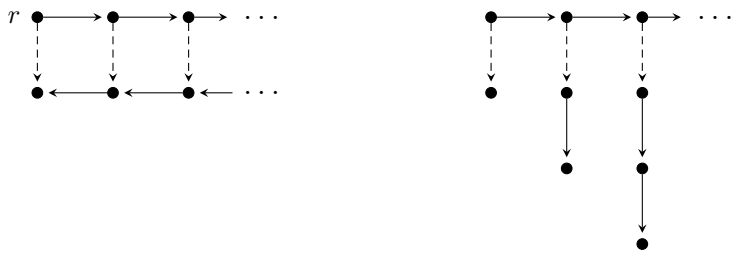


Figure 2.3: The ladder and its unfolding.

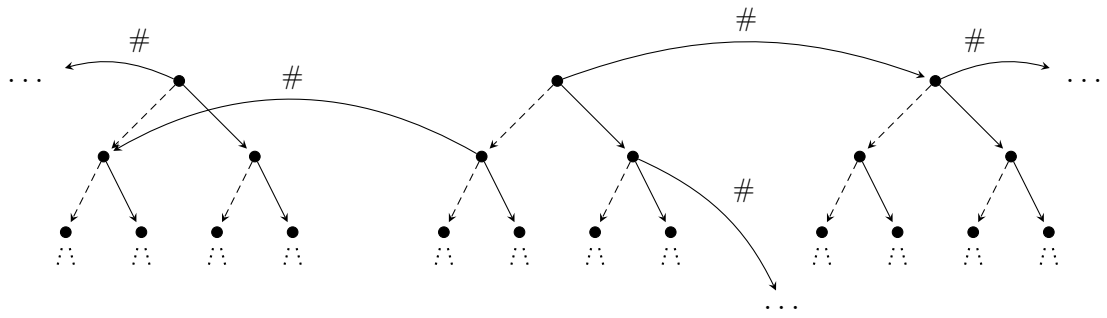


Figure 2.4: Treegraph of the complete binary tree.

It has to be noted that the root must be definable by a monadic formula. A counter-example in [Car06, Rem. 3.3.5] shows that the result is false without this constraint. The idea is that there is a forest of trees with decidable monadic theory where at least one tree with undecidable theory is "hidden".

Treegraph

This operation first appears in a weak form in [She75]. The standard form appears in [Sem84] and is attributed to Muchnik, who never published it. The *treegraph* of a graph G, noted Treegraph(G), is the set

$$\{x \xrightarrow{a} y\} \subseteq V_G^+ \times (\Sigma_G \cup \{\#\}) \times V_G^+$$

where $(x, y) \in V_G^+$ are sequences of vertices of G, and either

- $a \in \Sigma_G$, $x = wp$, $y = wq$ where $w \in V_G^*$ and $p \xrightarrow{a} q \in G$,
- or $a = \#$, $x = wp$ and $y = wpp$.

One can also see the treegraph as the fixpoint of the operation which, to each vertex which is not starting point of an # arc, adds this arc leading to the location of this vertex in a copy of G. The starting graph is called the *root graph*.

The following result is due to Muchnik, but the first complete proof only appears in [Wal02].

Theorem 2.4.5 (Muchnik)**.** *The treegraph operation is MSO-compatible.*

2.5 The pushdown hierarchy

The *pushdown hierarchy* — sometimes called the *Caucal hierarchy* — is a family of sets of graphs having a decidable monadic theory. It covers actually two families : one noted $(\text{Graph}_n)_{n\geq 0}$, and one only composed of trees, noted $(\text{Tree}_n)_{n\geq 0}$. For all $n \geq 0$, we have $\text{Tree}_n \subseteq \text{Graph}_n$, $\text{Tree}_n \subseteq \text{Tree}_{n+1}$, and $\text{Graph}_n \subseteq \text{Graph}_{n+1}$.

As stated in the introduction, the history of the pushdown hierarchy starts with the result of Muller and Schupp [MS85] about a geometrical property on configuration graphs of pushdown automata from a given configuration. The graph decomposition of a graph G from a vertex $r \in V_G$ is the family of the connected components of the subgraphs $(G_{r,n})_{n\geq 0}$. The n^{th} decomposition $G_{r,n}$ is the subgraph which support is the set of vertices at distance at least n of r. The graph is said finitely decomposable if, for any vertex r, the graph decomposition is finite up to isomorphism.

Example 2.5.1. If we decompose the complete binary tree from its root, the graph decomposition is reduced to the singleton of the binary tree itself. More generally, if we decompose this tree from a vertex of depth k, the cardinal of the graph decomposition is $k + 1$. ▲

Theorem 2.5.2 ([MS85])**.** *Configuration graphs of pushdown automata starting from a given configuration are exactly finitely decomposable graphs of finite degree.*

In particular, these graphs have a decidable monadic property. This work was extended to HR-equational graphs by [Cou90], and then to prefix-recognizable graphs [Cau92, Cau96].

In parallel, the notion of tree as infinite solution of a recursion scheme was brought back under the lights thanks to a result of decidability of MSO-theory under a condition of safety [KNU01, KNU02]. This approach will be detailed in Chapter 5. A similar result appears in [CK02]. Then [Cau02] shows that these term-trees are the same, and that they also are the unfoldings of prefix-recognizable graphs.

This results naturally lead to the full definition of the hierarchy in [Cau03] : trees are unfolded by graphs, which are in turn inverse rational mappings of trees. It was then showed in [CW03] that the inverse rational mappings can be replaced by general monadic interpretations, and that the graphs of the hierarchy were ε-closures of configurations graphs of higher-order pushdown automata.

2.5.1 Definition

We only define here the *outer* presentation, *i.e.* by graph transformations. An *inner* definition by higher-order stack relations, introduced in [Car05], appears in Section 3.3.2. Chapter 5 presents the recursion schemes approach.

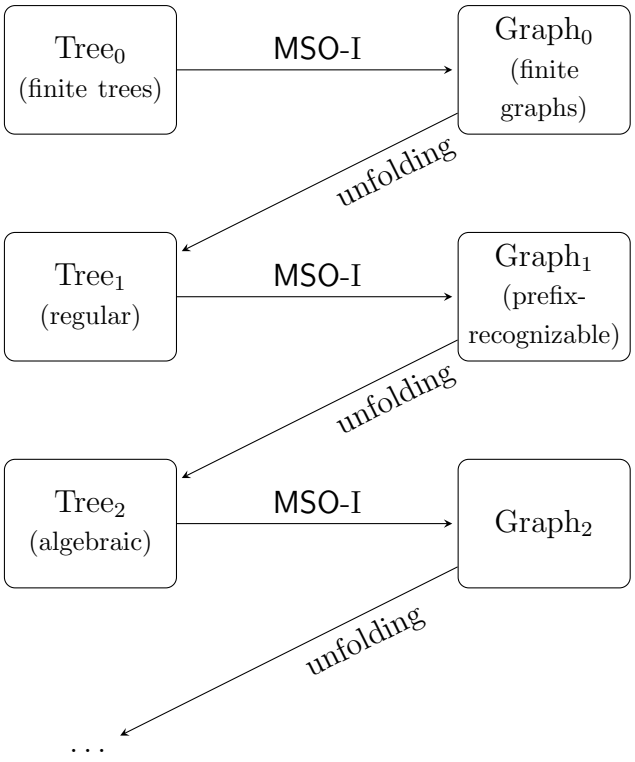


Figure 2.5: The pushdown hierarchy.

The pushdown hierarchy can defined as follows :

- Tree$_0$ is the set of finite trees,
- for $n \geq 0$, Graph$_n$ is the set of monadic interpretations of Tree$_n$,
- for $n \geq 0$, Tree$_{n+1}$ is the set of unfoldings of Graph$_n$.

The lower levels of this two-fold hierarchy are illustrated in Figure 2.5; a practical example of graph constructions can be seen in Figure 2.6.

A major consequence of the Theorem 2.4.4 is that the whole hierarchy enjoys a decidable monadic theory.

Theorem 2.5.3. *For all $n \geq 0$, the monadic theory of a graph in* Graph$_n$ *is decidable.*

2.5.2 Some properties

By [CW03], deterministic trees are enough : a graph of Graph$_n$ is the inverse rational mapping of a deterministic tree in Tree$_n$. The same paper also states that there is a *generator* Δ_2^n, *i.e.* a deterministic graph such that each graph is the inverse rational mapping of a rational marking of Δ_2^n. This generator is defined as follows :

- Δ_2^1 is the complete binary tree;

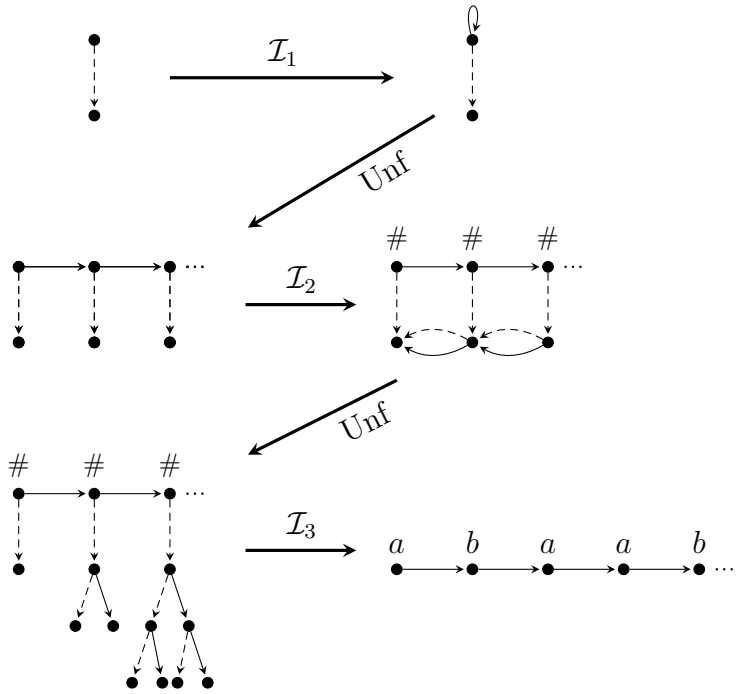


Figure 2.6: Exemple of graph constructions in the hierarchy.
These graphs belong successively to Tree_0, Graph_0, and so on up to Graph_2. Graphs are always unfolded from their root, *i.e.* their uppermost leftmost vertex. The three interpretations $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2$ are as follows :

$$
\begin{array}{rcrcl}
 & & \mathrm{leaf}(p) & = & \neg\exists r\ (p \to r) \\
\mathcal{I}_0 & : & \varphi_0(p,q) & = & p \xrightarrow{0} q \\
 & & \varphi_1(p,q) & = & p = q \wedge \neg\mathrm{leaf}(p) \\
\mathcal{I}_1 & : & \psi_0(p,q) & = & p \xrightarrow{0+\bar{0}\bar{1}0} q \\
 & & \psi_1(p,q) & = & p \xrightarrow{1+\bar{0}\bar{1}0} q \\
 & & \psi_\#(p) & = & \neg\mathrm{leaf}(p) \\
\mathcal{I}_2 & : & \eta_1(p,q) & = & \exists r\ (p \xrightarrow{\bar{1}^*\bar{0}} r \xrightarrow{0^*} q \wedge \mathrm{leaf}(p) \wedge \mathrm{leaf}(q) \wedge \neg(\#, r)) \\
 & & & & \vee(\mathrm{leaf}(p) \wedge p \xrightarrow{\bar{1}^*\bar{0}} q) \wedge (\#, q)) \\
 & & & & \vee(\mathrm{leaf}(q) \wedge p \xrightarrow{0^*} q) \wedge (\#, p)) \\
 & & \eta_a(p) & = & \exists q\ (\eta_1(p,q)) \wedge \mathrm{leaf}(p) \\
 & & \eta_b(p) & = & \exists q\ (\eta_1(p,q)) \wedge (\#, p)
\end{array}
$$

The final structure is actually the infinite (morphic) word $abaabaaaab\ldots$, which we will see again in Chapter 5.

- $\Delta_2^{n+1} = \text{Treegraph}(\Delta_2^n)$.

In this thesis, we will extensively use the following closure properties of Tree_n and Graph_n.

Proposition 2.5.4 ([CW03])**.** *For all n,*

- *the deterministic trees of* Tree_n *are closed under* MSO*-coloring;*
- Graph_n *is closed under* MSO*-transduction.*

Deterministic trees of Tree_n is also closed by subtree. For any tree t and node u, the subtree of root u is noted $t_{/u}$.

Proposition 2.5.5. *For all deterministic tree $t \in \text{Tree}_n$ and for all $u \in \text{Dom}(t)$, $t_{/u}$ also belongs to* Tree_n.

Proof. Let t' be the tree obtained by coloring any node below u with a fresh color \$. By Prop. 2.5.4, t' belongs to Tree_n. Let $G \in \text{Graph}_n$ and $r \in V_G$ such that $t \simeq \text{Unf}(G, r)$. Let r' be the vertex of G corresponding to u and let $\mathcal{I}$ be the interpretation erasing all the vertices which are not colored by \$ and then removing \$. Clearly $t_{/u} \simeq \text{Unf}(\mathcal{I}(G), r')$. Hence $t_{/u}$ belongs to Tree_n. □

Finally, we will make some use of the following selection properties on graphs and trees. These properties are given for completeness sake.

Proposition 2.5.6. *Let $G \in \text{Graph}_n$ be a deterministic tree, a* MSO*-formula $\varphi(X)$ and a fresh color \$. If $G \models \exists X, \varphi(X)$ then there exists $U \subseteq V_G$ s.t. $G \models \varphi[U]$ and $G \cup \{(\$, u) \mid u \in U\}$ also belongs to* Graph_n.

Proof. In [Car06, Theorem 5.3.1] it is shown that for all $n \geq 0$, any graph in Graph_n can be MSO-interpreted in a unique graph written GStacks_n. In [Fra05], it is shown that GStacks_n has the selection property. That is to say if $\text{GStacks}_n \models \exists X, \varphi(X)$ then there exists $\psi(x)$ such that $G \models \varphi[U]$ where $U = \{u \in V_G \mid G \models \psi[u]\}$. A proof of this fact is also given in [Car06, Theorem 4.7.6.].

Let $G \in \text{Graph}_n$ and $\varphi(X)$ be an MSO-formula s.t. $G \models \exists X, \varphi(X)$. Let $\mathcal{I}$ be an MSO-intrepretation such that $G \equiv \mathcal{I}(\text{GStacks}_n)$. Let $\varphi'(X)$ be an MSO-formula such that for all set U of vertices of GStacks_n, $\mathcal{I}(\text{GStacks}_n) \models \varphi[U]$ iff $\text{GStacks}_n \models \varphi'[U]$. In particular $\text{GStacks}_n \models \exists X, \varphi'(X)$.

Let $\psi(x)$ be the formula obtained using the selection property on GStacks_n for $\varphi'(X)$. Consider the MSO-interpretation $\mathcal{I}'$ obtained by adding to the formulæ defining $\mathcal{I}$ a formula $\varphi_\$(x) = \psi(x)$. The graph $\mathcal{I}'(\text{GStacks}_n) \in \text{Graph}_n$ satisfies the desidered properties. □

Proposition 2.5.7. *Let $t \in \text{Tree}_n$ be a deterministic tree, a* MSO*-formula $\varphi(X)$ and a fresh color \$. If $t \models \exists X, \varphi(X)$ then there exists $U \subseteq \text{Dom}(t)$ s.t. $t \models \varphi[U]$ and $t \cup \{(\$, u) \mid u \in U\}$ also belongs to* Tree_n.

Proof. Let t be a deterministic tree in Tree_n and let $\varphi(X)$ be an MSO-formula s.t. $t \models \exists X, \varphi(X)$.

For all $U \subseteq \mathrm{Dom}(t)$, we write $t[U]$ the deterministic tree colored by $\{0,1\}$ with the same set of nodes as t and such that for all $u \in t$, $t[U](u) = 1$ iff $u \in U$. We extend this definition to tuples of sets and to graphs. Let $\mathcal{A} = (Q, q_0, \Delta, \Omega)$ be a parity tree automaton accepting the set of determinstic tree over Σ colored $\{0,1\}$ such that:

$$\mathcal{A} \text{ accepts } t[U] \Leftrightarrow t \models \varphi[U].$$

Let $\mathcal{B}$ be the automaton obtained by projecting the colors in $\mathcal{A}$ (*i.e.* the set of states of $\mathcal{A}$ is $Q \times \{0,1\}$ where Q is the set of states of $\mathcal{A}$). To obtain our result, it is enougth to show that we can color an accepting run for $\mathcal{B}$ on t and still remain in Tree_n. Indeed to every accepting run of $\mathcal{B}$ correponds a set $U \subseteq \mathrm{Dom}(t)$ and an accepting run of $\mathcal{A}$ on $t[U]$.

For technical reason, we are going to color a strategy for the automaton $\mathcal{B}$ and not directly an accepting run. A strategy Φ for $\mathcal{B}$ on t is a mapping form t to $\mathcal{F}$ where $\mathcal{F}$ is the finite set of partial functions f from Q to Δ such that for all $q \in Q$, $f(q)$ (if defined) is a transition starting from q. This strategy Φ is *winning* if that for all $u \in \mathrm{Dom}(t)$ s.t. $f = \Phi(u)$ and for all $q \in \mathrm{Dom}(f)$, the run of $\mathcal{B}$ induced by Φ starting from u in state q is accepting. Conversely, for all $u \in t$ and all state q of $\mathcal{B}$, if $\mathcal{B}$ admits an accepting run from q starting from u on t. As $\mathcal{F}$ is a finite set, a strategy for $\mathcal{B}$ can be coded by a tuple of sets.

Obviously if we show that t colored with any winning strategy for the automaton $\mathcal{B}$ belongs to Tree_n. We have also shown that t colored by an accepting run (or colored by a set U such that $\mathcal{A} \models t[U]$) belongs to Tree_n.

The key property we are going to exploit is that we can restrict our attention to so called *regular winning strategies*. A winning strategy is regular if for all $u, v \in t$, if $t_{/u} \equiv t_{/v}$ then $\Phi(u) = \Phi(v)$. It follows from the positional determinacy of parity games that any automaton admits a regular winning strategy [Car06, Lemma 1.4.6].

The interest of regular winning strategies becomes apparent when we recall that every tree in t is obtained by unfolding a graph in Graph_n. Let G be a graph in Graph_n and let r be a vertex in V_G such that $t \equiv \mathrm{Unf}(G, r)$. By considering regular strategies, we can assume that all points originating from the same vertex of G in the unfolding are assigned the same value by the strategy. This allows us to color t with a regular winning strategy "before unfolding". It follows form [Car06, Proposition 1.4.9] that there exists a formula $\psi(\bar{X})$ such that for any tuple $\bar{U}$ of vertices of G, $G \models \psi[U]$ iff $\mathrm{Unf}(G[U], r)$ correspond to t colored with a winning strategy. By Prop. 2.5.6, there exists $\bar{U}$ such that $G \models \psi[\bar{U}]$ and $G[\bar{U}]$ belongs to Graph_n. □

Chapter 3

Linear order construction

This chapter locates some particular scattered orders in the pushdown hierarchy. It begins with the construction of ordinals by graph transformations : the towers of ω of height n are found in Graph_n. The same method is then extended to powers of ζ. For ordinals, we also give a higher-order n-regular presentation, *i.e.* relations on stacks of stacks. In Chapter 5, we will see a third construction of ordinals by schemes. Therefore these well-known structures offer a panorama of techniques that can be used in the hierarchy.

While using MSO logics on ordinals, an important result can be noticed. Büchi's Theorem 3.4.2 states that ordinals larger than ω^ω have a "redundant" MSO theory; for instance, $\mathrm{MTh}(\omega^\omega) = \mathrm{MTh}(\omega^{\omega^\omega})$. In other words, a given ordinal cannot be recognized by a MSO formula. The last section of this chapter studies the structure of the so-called canonical fundamental sequences of an ordinal, which is very close to the structure of the ordinal itself, but has the MSO-recognizability. These structures also belong to the hierarchy for ordinals smaller than ε_0.

3.1 Ordinals in the pushdown hierarchy

Finite ordinals are all in Graph_0, which is the set of finite graphs. On the next level, ordinals smaller than ω^ω are also easy to locate in the first level of the hierarchy. This has been proven by [Hei80, BC01]. We restate this result.

Proposition 3.1.1. *For any $\alpha < \omega^\omega$, $\alpha \in \mathrm{Graph}_1$.*

Proof. Let G_k be the finite graph of vertices $\{a^i \mid 0 \leq i \leq k\}$, as shown in Figure 3.1.

$$G_k = \{a^i \xrightarrow{1} a^i, a^i \xrightarrow{0} a^{i-1} \mid 0 < i \leq k\}$$

The leaves of unfolding of G_k by a^k are of the form $1^{c_{k-1}}01^{c_{k-2}} \dots 01^{c_0}0$. Ordered by lexicographic ordering, this set is isomorphic to $(\omega^{k-1}.c_{k-1} + \dots + c_0)_{\forall i<k, c_i \geq 0}$, which is ω^k. We have seen in Example 2.4.1 that the frontier of a deterministic tree is a MSO-interpretation of this tree, so $\omega^k \in \mathrm{Graph}_1$. For details on the sum closure, see Lemma 3.1.3. □



Figure 3.1: Finite graph G_3 which unfolding has frontier ω^3.

The natural question is whether greater ordinals belong to the hierarchy, and up to which bound. In particular, [Cac06] asks whether $\omega^\omega \in \text{Graph}_2$ or not. This section partially answers this question by giving examples of ordinals; the other direction will be seen in Chapter 4.

Theorem 3.1.2. *If* $\alpha < \omega \uparrow\uparrow (n+1)$*, then* $\alpha \in \text{Graph}_n$.

Using the Cantor normal form of ordinals smaller than ε_0, we only have to implement addition and the operation $\alpha \mapsto \omega^\alpha$. Here, we prove that ordinals of Graph_n are closed by addition, and that the ω-exponentiation only reaches the next level.

Lemma 3.1.3. *If* $\alpha, \beta \in \text{Graph}_n$*, then* $\alpha + \beta \in \text{Graph}_n$.

Proof. It suffice to prove that for any $G_1, G_2 \in \text{Graph}_n$, $x_1 \in V_{G_1}, x_2 \in V_{G_2}$, if x is a new vertex and $\#_1, \#_2$ new letters, the graph $G = G_1 \cup G_2 \cup \{x \xrightarrow{\#_1} x_1, x \xrightarrow{\#_2} x_2\}$ is also in Graph_n. Indeed, if $G_1 = \alpha$ and $G_2 = \beta$, we can supply a monadic interpretation to get the sum.

This is true for finite graphs. For $n > 0$, if $G_1 = \mathcal{I}_1(Unf(H_1, r_1))$ and $G_2 = \mathcal{I}_2(Unf(H_2, r_2))$, then by induction we define H as above is in Graph_{n-1} : $H = H_1 \cup H_2 \cup \{r \xrightarrow{\#_1} r_1, r \xrightarrow{\#_2} r_2\}$. Then $Unf(H, r) \in \text{Tree}_n$. It is easy to mark sets isomorphic to V_{G_1} and V_{G_2} and to restrict respectively $\mathcal{I}_1$ and $\mathcal{I}_2$ to these sets. We get the required G. □

Lemma 3.1.4. *If* $\alpha \in \text{Graph}_n$ *then* $\omega^\alpha \in \text{Graph}_{n+1}$.

Proof. By Corollary 2.2.2, the structure ω^α is isomorphic to the finite decreasing sequences of ordinals smaller than α. Let $G = \text{Treegraph}(\alpha)$. We are going to find an interpretation $\mathcal{I}$ such that $\mathcal{I}(G) \simeq \omega^\alpha$. For an illustration, the Figure 3.7 page 62 shows the case of ω in the simpler frame of covering graphs, which will be defined in Section 3.4.

G has exactly one root 0. Let M be the formula

$$M(x) := x = 0 \vee 0 \xrightarrow{<\#(\bar{<}\#)^*} x.$$

By the definition of treegraph, for any x such that $M(x)$ and $x \neq 0$, we have

$$0 \xrightarrow{<\#} \gamma_1\gamma_1 \xrightarrow{\bar{<}\#} \gamma_1\gamma_2\gamma_2 \xrightarrow{\bar{<}\#} \dots \xrightarrow{\bar{<}\#} \gamma_1 \dots \gamma_n\gamma_n = x.$$

We note $\vec{s}_x$ the sequence $(\gamma_1, \dots, \gamma_{n-1}, \gamma_n)$ with $\alpha > \gamma_1 \geq \dots \geq \gamma_n$. Conversely, each such sequence is associated to a x such that $M(x)$. We define now a monadic interpretation $\mathcal{I} = \{\varphi_<\}$ on these marked vertices so that $\varphi_<(x, y)$ iff $\vec{s}_x <_{\text{lex}} \vec{s}_y$. Define

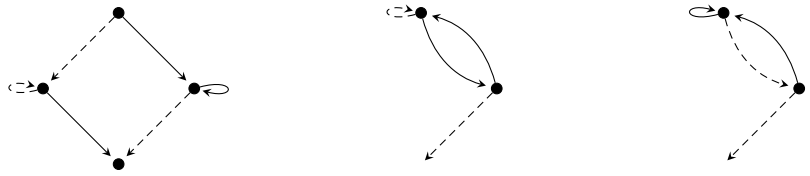


Figure 3.2: Folded graphs of trees of frontier ζ, η and $\omega(1+\eta)$.

$$\varphi_<(x,y) := M(x) \wedge M(y) \wedge x = 0 \vee x \xrightarrow{(\bar{\#}<)^*(\#\bar{<})^+} y.$$

The case $x = 0$ is easy, so we suppose that $x \neq 0$ and $\varphi_<(x,y)$. Then there is a z such that $x \xrightarrow{(\bar{\#}<)^*} z \xrightarrow{(\#\bar{<})^*} y$. Then there is k such that $\vec{s}_x = (\gamma_1, \ldots, \gamma_k, \ldots, \gamma_n)$ and $\vec{s}_z = (\gamma_1, \ldots, \gamma_{k-1}, \gamma'_k)$ with $\gamma_k < \gamma'_k$. In the same way $\vec{s}_y = (\gamma_1, \ldots, \gamma_{k-1}, \gamma'_k, \ldots, \gamma'_{n'})$, so $\vec{s}_x <_{\text{lex}} \vec{s}_y$. The converse is also true. □

As a direct consequence of Theorem 3.1.2, ordinals below ε_0 can be expressed by higher-order pushdown automata. This approach is explained in Section 3.3. In fact, this result is useful to illustrate the techniques that may be used in the hierarchy. It is therefore proved four times :

- directly by the above proof;
- by n-regular relations, in Section 3.3;
- by covering graphs, as an application of Propositions 3.4.4 and 3.4.13;
- by higher-order schemes, as an application of Example 5.1.2.

3.2 Powers of ζ

More complex orders can also be found in the pushdown hierarchy. In particular, trees of frontier ζ or η (resp. order types of $\mathbb{Z}$ and $\mathbb{Q}$), showed in Figure 3.2, are in Graph_1.

In this chapter, we focus on a particular family of scattered orders : namely, the successive ordinal powers of ζ. They are defined as follows.

$$\begin{aligned}
\zeta^0 &= 1 \\
\zeta^{\beta+1} &= \zeta^\beta \cdot \omega^* + \zeta^\beta + \zeta^\beta \cdot \omega \\
\zeta^\lambda &= \left(\sum_{\alpha<\lambda} \zeta^\alpha.\omega\right)^* + 1 + \sum_{\alpha<\lambda} \zeta^\alpha.\omega \qquad \text{for } \lambda \text{ limit.}
\end{aligned}$$

Note that the last line alone is actually enough for a complete definition, for any ordinal λ.

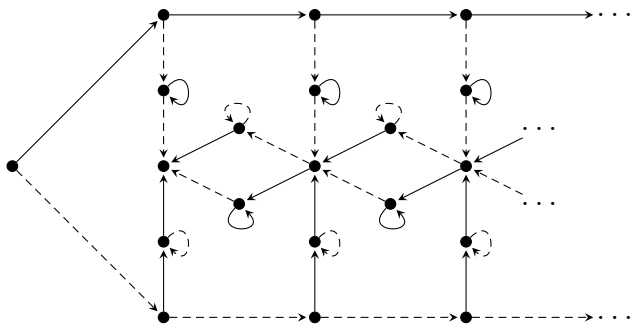


Figure 3.3: Folded regular graph of a tree of frontier ζ^ω.

From [Ros82, Thm. 5.37], the Hausdorff-rank of ζ^α is α. Furthermore, ζ^α is complete among the scattered orders of Hausdorff-rank α in the following sense: a scattered order has Hausdorff-rank less than α if and only if it is a subordering of ζ^α.

The following proposition is an extension of Theorem 3.1.2.

Proposition 3.2.1. *For all $n > 0$ and any ordinal $\alpha < \omega \uparrow\uparrow n$, ζ^α is in* Graph_n.

For instance, the graph of Figure 3.3 is regular and thus in Graph_1. Its unfolding by its root (the leftmost vertex) is in Graph_2 and yields the order ζ^ω.

To prove the result, we need to extend the definition of the treegraph operation to go in more than one direction. A similar approach can also be found in [Pil04].

Definition. Let Σ, Γ be two alphabets. The Γ-*treegraph* of a Σ-graph G is the graph in $(V_G \cup \Gamma)^* \times \Sigma \cup \Gamma \times (V_G \cup \Gamma)^*$ given by

$$\begin{aligned} G \;=\; & \{ux \xrightarrow{b} uy \mid x \xrightarrow{b} y \in G\} \\ & \cup \{ux \xrightarrow{a} uxax \mid a \in \Gamma\} \end{aligned}$$

Note that the constraint $\Sigma \cap \Gamma = \emptyset$ is not required, but this will be the case in our application, and it is easy to see that it is not a restriction.

The main property of this treegraph version is that it does not add more complexity than the standard treegraph, which is isomorphic to the case $\Gamma = \{\#\}$.

Proposition 3.2.2. *The Γ-treegraph of a graph in* Graph_n *is in* Graph_{n+1}.

Proof. Suppose $\Sigma \cap \Gamma = \emptyset$ up to final renaming. Let $G \in \mathrm{Graph}_n$. For each $a \in \Gamma$, let $\mathcal{T}$ be the transduction adding an arc labeled by a from each vertex of G. Then the Γ-treegraph of G is isomorphic to the interpretation of $\mathrm{Treegraph}(\mathcal{T}(\alpha))$ where each path labeled $a\#\bar{a}$ is changed into a. □

Proof of Prop. 3.2.1. This proof is an adaptation of the construction for ordinals. By Theorem 3.1.2, for any $n > 0$, any ordinal smaller than $\omega \uparrow\uparrow n$ is in Graph_{n-1}. So it is enough to prove that if $\alpha \in \mathrm{Graph}_{n-1}$, then $\mathbb{Z}^\alpha$ is in Graph_n.

In a way similar to Proposition 2.2.1, ζ^α is isomorphic to the set of α-sequences over $\mathbb{Z}$ where a finite number is non-zero, and ordered by reverse lexicographic order — see again [Ros82, 5.36]. For two α-sequences $\vec{s}, \vec{t}$, we have $\vec{s} <_{\text{rlex}} \vec{t}$ iff $\exists\gamma$ such that $\vec{s}_\gamma < \vec{t}_\gamma$ and $\forall\delta > \gamma$, $\vec{s}_\delta = \vec{t}_\delta$.

Let $\alpha \in \text{Graph}_{n-1}$ such that $\alpha < \omega \uparrow\uparrow n$. Let G be the $\{0,1\}$-treegraph of α; by Proposition 3.2.2, $G \in \text{Graph}_{n+1}$. G has still exactly one root r (a vertex co-accessible from V_G). Let $\mathcal{M}$ be a marking adding a special color on r and any x such that

$$r \xrightarrow[G]{(<+\varepsilon)(0^+ + 1^+)(\bar{<}(0^+ + 1^+))^*} x.$$

Here $x \neq r$ is of the form $(\gamma_0 a_0)^{c_1}.....(\gamma_k a_k)^{c_k}$ where for all $i \geq 0$, $c_i > 0$, $a_i \in \{0,1\}$ and $\gamma_i < \gamma_{i-1}$. To each such x we associate the α-sequence $\vec{s}(x)$ where

$$\begin{array}{rcll} \vec{s}(x)_{\gamma_i} & = & -c_i & \text{if} \quad a_i = 0 \\ \vec{s}(x)_{\gamma_i} & = & c_i & \text{if} \quad a_i = 1 \\ \vec{s}(x)_\gamma & = & 0 & \text{if} \quad \gamma \neq \gamma_i \text{ for any } i \end{array}$$

The path is finite so a finite number of indexes are non-zero. Conversely, to each α-sequence with a finite number of non-zero indexes is associated a unique vertex in V_G.

We check that the relation $x < y$ on marked vertices holds iff $\vec{s}(x) <_{\text{rlex}} \vec{s}(y)$ where $<_{\text{rlex}}$ is the reverse-lexicographic order definable by a MSO-interpretation in G. The cases $x = 0$ and $y = 0$ are handled separately. Otherwise let

$$\begin{array}{rcl} x & = & (\gamma_0 a_0)^{c_0}.....(\gamma_k a_k)^{c_k}\gamma_k \\ y & = & (\delta_0 b_0)^{d_0}.....(\delta_h b_j)^{d_h}\delta_h. \end{array}$$

We have $x < y$ iff there is a j such that $a_i = b_i, \gamma_i = \delta_i, c_i = d_i$ for all $i < j$, and one of the following is true. We note $\uparrow = ((\bar{0} + \bar{1})^* <)$ and $\downarrow = (\bar{<}(0 + 1)^*)$. For an example, see Figure 3.4.

$$\begin{array}{lllll} \text{Either} & \gamma_j < \delta_j, & b_j = 1 & \Longrightarrow & x \xrightarrow{\uparrow^+ 1^+ \downarrow^*} y \\ \text{or} & \delta_j < \gamma_j, & a_j = 0 & \Longrightarrow & x \xrightarrow{\uparrow^* \bar{0}^+ \downarrow^+} y \\ \text{or} & \gamma_j = \delta_j, & j > 0 \text{ and} & & \\ & & a_j = b_j = 0, c_j > d_j & \Longrightarrow & x \xrightarrow{\uparrow^* \bar{0}^+ \downarrow^*} y \\ & \text{or} & a_j = b_j = 1, d_j > c_j & \Longrightarrow & x \xrightarrow{\uparrow^* 1^+ \downarrow^*} y \\ & \text{or} & a_j = 0, b_j = 1 & \Longrightarrow & x \xrightarrow{\uparrow^* \bar{0}^+ 1^+ \downarrow^*} y \\ \text{or} & k < j \leq h, & b_j = 1 & \Longrightarrow & x \xrightarrow{\bar{<} 1^+ \downarrow^*} y \\ \text{or} & h < j \leq k, & c_j = 0 & \Longrightarrow & x \xrightarrow{\uparrow^* \bar{0}^+ >} y. \end{array}$$

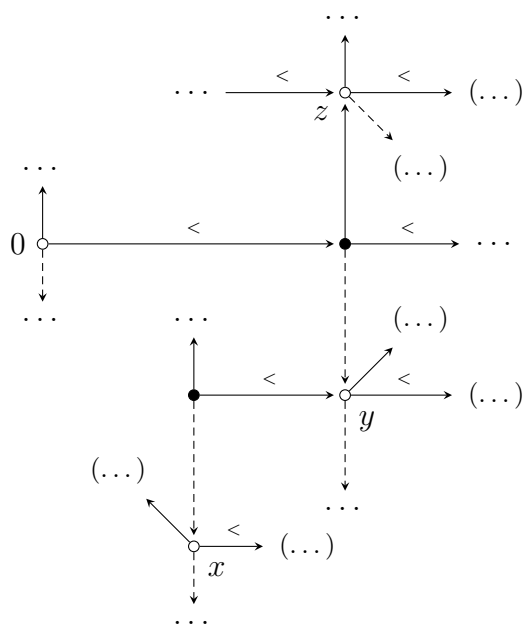


Figure 3.4: $\{0,1\}$-treegraph of ω.
White vertices are those used by the interpretation; bracketed dots represent subgraphs where no vertex will be used. Here $\vec{s}_x = (-1,-1,0,\dots)$, $\vec{s}_y = (0,-1,0,\dots)$, $\vec{s}_z = (0,1,0,\dots)$. We check for instance that $x \xrightarrow{\bar{0}<\bar{0}1} z$, noted $x \xrightarrow{\uparrow\bar{0}1} z$.

In the other direction,

$$\begin{aligned}
x \xrightarrow{\uparrow^*1^+\downarrow^*} y \quad &\Longrightarrow \quad \gamma_j < \delta_j, b_j = 1 \\
&\quad\ \text{or } \gamma_j = \delta_j, a_j = b_j = 1, d_j > c_j \\
x \xrightarrow{\uparrow^*\bar{0}^+\downarrow^*} y \quad &\Longrightarrow \quad \delta_j < \gamma_j, a_j = 0 \\
&\quad\ \text{or } \gamma_j = \delta_j, a_j = b_j = 0, c_j > d_j \\
x \xrightarrow{\uparrow^*\bar{0}^+1^+\downarrow^*} y \quad &\Longrightarrow \quad \gamma_j = \delta_j, a_j = 0, b_j = 1 \\
x \xrightarrow{\bar{<}1^+\downarrow^*} y \quad &\Longrightarrow \quad k < j \leq h, b_j = 1 \\
x \xrightarrow{\uparrow^*\bar{0}^+<} y \quad &\Longrightarrow \quad h < j \leq k, c_j = 0.
\end{aligned}$$

There is therefore a monadic interpretation $\mathcal{I}$ such that $\mathcal{I}(G) \simeq \zeta^\alpha$. Since Graph_n is closed under MSO-interpretation, $\zeta^\alpha \in \mathrm{Graph}_n$. □

3.3 n-regular presentation

The graphs on the first level of the hierarchy were originally defined with a prefix-recognizable presentation [Cau96]. Then this presentation was extended to any level with the help of higher-order pushdown automata of level n [CW03], i.e. automata which use nested stacks of stacks of depth n. Using these results, the construction by monadic interpretations and unfolding can be translated into a pushdown automata description.

Instead of doing so, we prefer the equivalent — and more fluent — notion of regularity [Car05] on n-stacks. This notion offers a natural encoding of ordinals by their Cantor normal form.

This section gives a presentation of prefix-recognizable graphs, and then of the more general n-regular relations. We show then how to actually build ordinals with this latter tool. As in Section 3.1, this latter operation is split in two parts : one for the $\alpha \mapsto \omega^\alpha$ operation and the other for addition.

3.3.1 Prefix-recognizable graphs

A prefix-recognizable Σ-graph is up to isomorphism a graph of the form

$$\{uw \xrightarrow{a} vw \mid (U, a, V, W) \in \Delta, u \in U, u' \in V w \in W, a \in \Sigma\}$$

where Δ is a finite set of tuples (U, a, V, W) such that U, V and W are regular languages.

Theorem 3.3.1 ([Cau96, Theorem 3.3])**.** *Prefix-recognizable graphs are up to isomorphism inverse rational mappings of the complete binary tree.*

This means that Graph_1 is (up to isomorphism) the set of prefix-recognizable graphs. As a result, the same paper states that the monadic theory of prefix-recognizable graphs is decidable.

3.3.2 Configuration graphs of n-hopdas

For a detailed presentation of these notions, see [Car05]. A 1-stack, or simply stack, over a finite alphabet Γ is a word over Γ. To avoid later confusion, we forbid letters of $\mathbb{N}$ (see below for the difference between pop_x and pop_n). The empty 1-stack is noted $[\,]_1$. For all $n > 1$, a n-stack over Γ is a non-empty finite sequence of $(n-1)$-stacks over Γ. The empty n-stack containing only the empty $(n-1)$-stack is noted $[\,]_n$. For all n, the set of n-stacks is noted $Stacks_n$ (or $Stacks_n(\Gamma)$) and the set of all stacks is $Stacks = \cup_{n\in\mathbb{N}} Stacks_n$.

The *operations* on a 1-stack $[a_1, \ldots, a_m]_1$ are the usual push and pop. We add pop_1 which can pop any letter.

$$\begin{aligned} \mathsf{push}_x([a_1, \ldots, a_m]_1) &:= [a_1, \ldots, a_m, x]_1, \\ \mathsf{pop}_x([a_1, \ldots, a_m = x]_1) &:= [a_1, \ldots, a_{m-1}]_1. \\ \mathsf{pop}_1([a_1, \ldots, a_m]_1) &:= [a_1, \ldots, a_{m-1}]_1. \end{aligned}$$

For $n > 1$ and a n-stack $[s_1, \ldots, s_m]_1$, the extended operations are as follows. The operation copy_n replicates the top-most $(n-1)$-stack, and $\overline{\mathsf{copy}}_n$ is its symetric : it deletes the top-most $(n-1)$-stack if it equals the penultimate. The pop_n operation simply removes the top-most $(n-1)$-stack. For $k < n$, the operation on k-stacks are simply propagated

in the top-most stack.

$$\begin{array}{rcll}
\mathsf{copy}_n([s_1,\dots,s_m]_n) & := & [s_1,\dots,s_m,s_m]_n & \\
\overline{\mathsf{copy}}_n([s_1,\dots,s_m,s_m]_n) & := & [s_1,\dots,s_m]_n & \\
\mathsf{pop}_n([s_1,\dots,s_m]_n) & := & [s_1,\dots,s_{m-1}]_n & \\
\mathsf{copy}_k([s_1,\dots,s_m]_n) & := & [s_1,\dots,\mathsf{copy}_k(s_m)]_n & \text{for } 1<k<n \\
\mathsf{pop}_k([s_1,\dots,s_m]_n) & := & [s_1,\dots,\mathsf{pop}_k(s_m)]_n & \text{for } 1\leq k<n \\
\mathsf{pop}_x([s_1,\dots,s_m]_n) & := & [s_1,\dots,\mathsf{pop}_x(s_m)]_n & \text{for } x\in\Gamma \\
\mathsf{push}_x([s_1,\dots,s_m]_n) & := & [s_1,\dots,\mathsf{push}_x(s_m)]_n &
\end{array}$$

Let also be an identity function id defined on $Stack$. The set of operations defined at least one n-stack is denoted Ops_n.

Instead of dealing with rough higher-order pushdown automata, one can use directly a regularity on stacks. The set of operations forms a monoid with the composition operation. Let $Reg(Ops_n)$ the closure of the finite subsets of this monoid under union, product and iteration, i.e. the set of *regular expressions* on Ops_n.

To each expression $E \in Reg(Ops_n)$ we associate the test operation $\mathsf{test}_E = \mathsf{id}_{|E([]_n)}$. By an abuse of language $\mathsf{test}_{[\,]_k} = \mathsf{id}_{|[\,]_k}$. Let $Test_n$ be the set of tests. To each $F \in Reg(Ops_n \cup Test_n)$ we now associate

$$\begin{array}{rl}
\text{the set of stacks} & S_n(F) = F([\,]_n), \\
\text{the set of relations} & R(F) = \{(s,s') | s' \in F(s)\}.
\end{array}$$

The set of stacks $S_n(F)$ will be noted $S(F)$ if n is clear. Given F and $(F_a)_{a\in\Sigma}$ in $Reg(Ops_n \cup Test_n)$, the graph of support $S(F)$ and with arcs $s \xrightarrow{a} s'$ iff $(s,s') \in R(F_a)$ is a *configuration graph of a n-hopda*. They describe precisely the graphs of the hierarchy.

Theorem 3.3.2 ([Car05])**.** *The family of configuration graphs of n-hopdas is equal up to isomorphism to* Graph_n.

3.3.3 Encoding ordinals

For each ordinal, we are going to define the expressions dom and inc which respectively fix the domain of the structure and the order relation. We also will build an expression dec to perform the symmetric of inc. In other words, we want the structure $\langle S(\mathsf{dom}(\alpha)), R(\mathsf{dec}(\alpha)), R(\mathsf{inc}(\alpha))\rangle$ to be isomorphic to the structure $\langle \alpha, >, <\rangle$.

Small ordinals

For ω, we consider the set of all 1-stacks (i.e. integers). In this case, $\mathsf{dom}(\omega)$ is obtained by iterating push_a on the empty stack with a fixed letter a. The other operations are also

straighforward.

$$\begin{aligned} \mathsf{dom}(\omega) &:= \mathsf{push}_a^* \\ \mathsf{inc}(\omega) &:= \mathsf{push}_a^+ \\ \mathsf{dec}(\omega) &:= \mathsf{pop}_a^+ \end{aligned}$$

Since we only consider ordinals in Graph_n for $n > 1$, our interest is focused on infinite ordinals. However, since we are going to encode these ordinals through the Cantor normal form, we have to define finite ordinals. For a finite ordinal $k > 0$, the domain is simply the restriction of $\mathsf{dom}(\omega)$ to 1-stacks of size bounded by $k - 1$.

To allow iteration and re-prove Theorem 3.1.2, we also need more than ω : we have to encode all ordinals smaller than ω^ω with 1-stacks. This is done with more letters. Let $\alpha = \omega^{k_1}.c_1 + \cdots + \omega.c_{k-1} + c_k$. The stack alphabet is $\{a_1, \ldots, a_k\}$. Stacks belong to a subset of $S(\mathsf{push}_{a_1}^* \ldots \mathsf{push}_{a_k}^*)$ where if $s = [a_k^{d_k} \ldots a_1^{d_1}]_1$, then the sequence $(d_i)_{i\in[0,k]}$ is smaller than $(k_i)_{i\in[0,k]}$ in lexicographic order. Relations also respect this order. Increasing a stack is done by popping all letters a_i where $i > j$ for a given j, then pushing one a_j and pushing anything. Decreasing a stack is done by popping all letters a_i where $i \geq j$ for a given j, then pushing only letters a_i with $i > j$.

One step further : exponentiation

Let α be any ordinal smaller than ε_0, and let n be the smallest value such that $\mathsf{dom}(\alpha)$, $\mathsf{inc}(\alpha)$ and $\mathsf{dec}(\alpha)$ are all in $Reg(Ops_{n-1})$. Informally, each ordinal $\gamma < \omega^\alpha$ is either 0 or may be written as $\gamma = \omega^{\gamma_0} + \cdots + \omega^{\gamma_k}$ with $\gamma_i < \alpha$; so we code γ as a sequence of stacks respectively coding $\gamma_0 \ldots \gamma_k$.

Let $\mathsf{tail}(\alpha) := \mathsf{copy}_n.(\mathsf{id} + \mathsf{dec}(\alpha))$. This operation takes the last stack (representing γ_k) and adds a stack coding an ordinal $\leq \gamma_k$, so that the CNF constraint is respected. For the relation $<$, inc either adds a decreasing sequence (by tail), or it first pops stacks, then increases a given one before adding a tail.

$$\begin{aligned} \mathsf{dom}(\omega^\alpha) &:= \mathsf{dom}(\alpha).\mathsf{tail}(\alpha)^* \\ \mathsf{inc}(\omega^\alpha) &:= [\mathsf{pop}_n^*.\mathsf{inc}(\alpha) + \mathsf{tail}(\alpha)].\mathsf{tail}(\alpha)^* \\ \mathsf{dec}(\omega^\alpha) &:= \mathsf{pop}_n^*.[\mathsf{pop}_n + \mathsf{dec}(\alpha).\mathsf{tail}(\alpha)^*] \end{aligned}$$

See Figure 3.5 for an example of operators at work.

Proposition 3.3.3. *If $n > 0$ and $\langle S(\mathsf{dom}(\alpha)), R(\mathsf{dec}(\alpha)), R(\mathsf{inc}(\alpha)) \rangle \simeq \langle \alpha, >, < \rangle$, then the same properties are true for ω^α.*

Proof. Let $n > 1$, and there exist $\mathsf{dom}(\alpha), \mathsf{inc}(\alpha), \mathsf{dec}(\alpha)$ operations in $Reg(Ops_{n-1})$ such that $\langle S(\mathsf{dom}(\alpha)), R(\mathsf{dec}(\alpha)), R(\mathsf{inc}(\alpha)) \rangle$ is isomorphic to $\langle \alpha, >, < \rangle$. It is not a restriction to suppose that $\mathsf{dec}(\alpha)(S(\mathsf{dom}(\alpha))) \subseteq S(\mathsf{dom}(\alpha))$: indeed, it suffices to concatenate the operation $\mathsf{test}_{dom(\alpha)}$. For any $\gamma < \alpha$, we note s_γ the corresponding $(n-1)$-stack.

Note that if $k < n$, all operations on k-stacks are valid on n-stacks. So if $f \in$

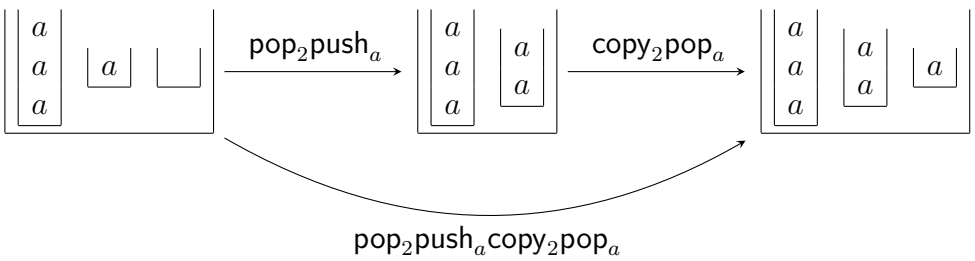


Figure 3.5: The operation $\mathsf{inc}(\omega^\omega)$.
The stacks here represent $\omega^3+\omega+1$, $\omega^3+\omega^2$ and $\omega^3+\omega^2+\omega$.

$Reg(Ops_k)$ and s, s' are two k-stacks such that $(s, s') \in R(f)$, and if p, p' are the same n-stack except for the top-most k-stack which is respectively s and s', then $(p, p') \in R(f)$.

Let $S = \mathsf{dom}(\omega^\alpha)$ and let $p \in S$ be a finite sequence of $(n-1)$-stacks, so $p = [s_{\gamma_0}, \ldots, s_{\gamma_k}]$. In the definition of $\mathsf{dom}(\omega^\alpha)$, there is no pop_n operation, and by definition $s_{\gamma_0}, \ldots, s_{\gamma_k}$ are all in $S(\mathsf{dom}(\alpha))$. By hypothesis on $\mathsf{dec}(\alpha)$, we also have $s_{\gamma_0} \geq \cdots \geq s_{\gamma_k}$. As a consequence, the mapping

$$p = [s_{\gamma_0}, \ldots, s_{\gamma_k}] \mapsto \lambda = \omega^{\gamma_0} + \cdots + \omega^{\gamma_k}$$

is well defined and is injective. In fact, it is a bijection between S and $[1, \omega^\alpha[$; omitting 0 is not a problem for infinite ordinals. We therefore note p_λ the n-stack associated to λ.

Now let $0 < \lambda < \lambda' < \alpha$ be two ordinals, with $\lambda = \omega^{\gamma_0} + \cdots + \omega^{\gamma_k}$ in CNF. Then

$$\begin{array}{rrcll} \text{either} & \lambda' & = & \omega^{\gamma_0} + \cdots + \omega^{\gamma_k} + \cdots + \omega^{\gamma_{k'}} & \text{with } k < k', \\ \text{or} & \lambda' & = & \omega^{\gamma_0} + \cdots + \omega^{\gamma_i} + \cdots + \omega^{\gamma'_{i+1}} + \cdots + \omega^{\gamma_{k'}} & \text{for some } i < k, \end{array}$$

with $\gamma_{i+1} < \gamma'_{i+1}$. For the first case, the use of $\mathsf{tail}(\alpha)$ on p_λ has already been discussed, so $(p_\lambda, p_{\lambda'}) \in R(\mathsf{tail}(\alpha)^+)$. In the second case, $\mathsf{pop}_n^{(k-i-1)}(p_\lambda) = [s_{\gamma_0}, \ldots, s_{\gamma_{i+1}}]$ and, by induction,

$$([s_{\gamma_0}, \ldots, s_{\gamma_{i+1}}], [s_{\gamma_0}, \ldots, s_{\gamma'_{i+1}}]) \in R(\mathsf{inc}(\alpha)).$$

Again, the tail operation is used. The converse — if $(p_\lambda, p_{\lambda'}) \in S^2 \cap R(\mathsf{inc}(\omega^\alpha))$ then $\lambda < \lambda'$ — is straightforward. So $\langle S, R(\mathsf{inc}(\omega^\alpha))\rangle$ is indeed isomorphic to $\langle \alpha, <\rangle$.

The dec operation is similar. In the first case, $\mathsf{pop}_n^{(k'-k)}(p_{\lambda'}) = p_\lambda$ with $k' - k \geq 1$. In the second case, $\mathsf{pop}_n^{(k'-i-1)}(p'_\lambda) = [s_{\gamma_0}, \ldots, s_{\gamma'_{i+1}}]$ and

$$([s_{\gamma_0}, \ldots, s_{\gamma'_{i+1}}], [s_{\gamma_0}, \ldots, s_{\gamma_{i+1}}]) \in R(\mathsf{dec}(\alpha)).$$

The converse is direct as well, and proves in the same time the last needed induction property : $\mathsf{dec}(\omega^\alpha)(S) \subseteq S$. Note that this was not true with inc : $\mathsf{inc}(\omega^\alpha)(S) \not\subseteq S$, because we could lose the decreasing constraint of the CNF.

Finally $\langle S, R(\mathsf{inc}(\omega^\alpha), R(\mathsf{dec}(\omega^\alpha))\rangle$ is isomorphic to $\langle \alpha, <, >\rangle$ and the induction prop-

erties are fulfilled. □

Completing with addition

To perform exponentation, $\mathsf{dom}(\omega^\alpha)$ only uses the letters used by $\mathsf{dom}(\alpha)$. For addition, we add some markers.

Let $\alpha_1, \alpha_2 < \varepsilon_0$ and let n be the smallest integer such that $\mathsf{dom}(\alpha_1), \mathsf{dom}(\alpha_2) \in Ops_n^*$. We add new letters α_1, α_2. We note $\mathsf{test}_{\bar{\alpha}_i}$ for $\mathsf{test}_{(Ops_n \setminus \{\mathsf{push}_{\alpha_i}\})^*}$, meaning in this context "there is no α_i in any stack".

$$\begin{aligned}
\mathsf{dom}(\alpha_1+\alpha_2) &= \mathsf{push}_{\alpha_1}\mathsf{dom}(\alpha_1) + \mathsf{push}_{\alpha_2}\mathsf{dom}(\alpha_2)\\
\mathsf{inc}(\alpha_1+\alpha_2) &= \mathsf{test}_{\bar{\alpha}_2}(\mathsf{inc}(\alpha_1) + \mathsf{dec}(\alpha_1)\mathsf{pop}_{\alpha_1}\mathsf{test}_{\bar{\alpha}_1}\mathsf{push}_{\alpha_2}\mathsf{dom}(\alpha_2))\\
&\quad +\mathsf{test}_{\bar{\alpha}_1}\mathsf{inc}(\alpha_2)\\
\mathsf{dec}(\alpha_1+\alpha_2) &= \mathsf{test}_{\bar{\alpha}_1}(\mathsf{dec}(\alpha_2) + \mathsf{dec}(\alpha_2)\mathsf{pop}_{\alpha_2}\mathsf{test}_{\bar{\alpha}_2}\mathsf{push}_{\alpha_1}\mathsf{dom}(\alpha_1))\\
&\quad +\mathsf{test}_{\bar{\alpha}_2}\mathsf{dec}(\alpha_1)
\end{aligned}$$

Proposition 3.3.4. *If $n > 0$ and $\langle\, S(\mathsf{dom}(\alpha_i)), R(\mathsf{dec}(\alpha_i)), R(\mathsf{inc}(\alpha_i))\,\rangle \simeq \langle\, \alpha_i, >, <\rangle\,\rangle$ for $i \in \{1,2\}$, then the same is true for $\alpha_1 + \alpha_2$.*

Proof. The α_1 and α_2 parts are respectively encoded by the new marker of the same name at the beginning of each stack. The only way to remove a first letter α_1 is to use pop_1 or pop_{α_1}, both of which are never used by $\mathsf{inc}(\alpha_1)$ or $\mathsf{dec}(\alpha_1)$. So the relations inc and dec cannot accidentally remove this marker; the set $\mathsf{dom}(\alpha_1)$ is closed by $\mathsf{inc}(\alpha_1)$ and $\mathsf{dec}(\alpha_1)$.

To increase a stack, either this stack begins with α_1 or α_2 : this is checked by the operation $\mathsf{test}_{\mathsf{push}_{\alpha_i} Ops_n^*}$. In the case of α_2, using $\mathsf{inc}(\alpha_2)$ is enough. For stacks in the α_1 part, greater stacks are accessible by the $\mathsf{inc}(\alpha_1)$ operations and by taking all stacks in the α_2 part. To reach the latter, it is enough to pop everything including the α_1 marker, and start with $\mathsf{push}_{\alpha_2}\mathsf{dom}(\alpha_2)$. The dec operation works the same way. □

Main result

To sum up, we have defined small ordinals in terms of configuration graphs of (standard) pushdown automata. Then we showed how to perform exponentiation by increasing the order, and addition at the same order. We get therefore the main result of this section.

Theorem 3.3.5. *For any $\alpha < \omega \uparrow\uparrow n+1$, α is isomorphic to the configuration graph of a n-hopda.*

By the equivalence of Theorem 3.3.2, this result is therefore a new proof of Theorem 3.1.2.

3.4 Covering graphs

We have seen in Example 2.3.2 that it is possible to recognize the structure of an ordering with the help of FO-logic, and well-ordering with MSO-logic in Example 2.3.3. But is

there a formula φ_α defining precisely the ordinal α, *i.e.* such that only the ordinal α satisfies φ_α? For ordinals smaller than ω^ω, it is a simple exercise.

Proposition 3.4.1. *For $\alpha < \omega^\omega$, there is a MSO-formula φ_α such for any $\{<\}$-graph G, $G \models \varphi_\alpha$ if and only if $G \simeq \alpha$.*

Proof. The conjunction of formulæ of Examples 2.3.2 and 2.3.3 checks the well-ordering of the structure. We define the formulæ φ_k recognizing the k-limit vertices, *i.e.* limits of $(k-1)$-limit vertices.

$$\begin{aligned} \varphi_0(x) &:= \top, \\ \varphi_k(x) &:= \forall y < x, \varphi_{k-1}(y) \Rightarrow \exists z\ (\varphi_{k-1}(z) \wedge y < z < x). \end{aligned}$$

To recognize the ordinal $\omega^k.c_k + \cdots + \omega.c_1 + c_0$, it is enough to find exactly c_k vertices satisfying φ_k, then c_{k-1} vertices greater than the previous ones and satisfying φ_{k-1}, and so on. □

This method is not powerful enough for higher ordinals, and in the general case the property was proven false in [Bü65, Büc73]. To present this result we need an additional definition. Any ordinal α has a unique representation

$$\alpha = \omega^\omega.\nu + \omega^k.c_k + \cdots + \omega^1.c_1 + c_0$$

The ordinal $\omega^\omega.\nu$ is called the *ω-head* of α, and $\omega^k.c_k + \cdots + \omega^1.c_1 + c_0$ is its *ω-tail*. The monadic theory of α only depends on the latter and whether or not $\alpha < \omega^\omega$.

Theorem 3.4.2 ([Büc73, Th. 4.9])**.** *For any countable ordinals α and β, the following statements are equivalent :*

- $\mathrm{MTh}(\alpha) = \mathrm{MTh}(\beta)$
- *either $\alpha = \beta < \omega^\omega$ or else $\omega^\omega \leq \alpha, \beta$ and α, β have the same ω-tail.*

This result means informally that ordinals are not easily manipulable by MSO formulæ; for instance, in general it is not possible to get a ordinal from a greater one by a monadic interpretation. In this section, we by-pass this problem by considering a structure based on the well-known notion of fundamental sequence. This structure is called the covering graph of an ordinal. One of its important properties is its finite out-degree, which is worked out to bring a specific monadic formula for each covering graph, thus allowing to differentiate them.

3.4.1 Fundamental sequence

For any ordinal α, a subset $S \subseteq \alpha$ is *cofinal* in α if for any $\beta \in \alpha$, there is $\beta' \in S$ such that $\beta \leq \beta'$. Such S has an ordinal order type. The *cofinality* of α is the minimal order

type of a subset cofinal in α. The cofinality [Ros82] of any countable ordinal is ω. This means that to each limit ordinal α we may associate an ω-sequence whose supremum is α.

Definition. For $\alpha \leq \varepsilon_0$, $\alpha = \beta + \omega^\gamma$ with $\beta < \alpha$, $\gamma < \alpha$ and ω^γ is the last term in the CNF of α, the (canonical) *fundamental sequence* $(\alpha[n])_{n<\omega}$ is defined as follows :

$$\alpha[n] = \begin{cases} \beta + \omega^{\gamma'}.(n+1) & \text{if } \gamma = \gamma' + 1 \\ \beta + \omega^{\gamma[n]} & \text{otherwise.} \end{cases}$$

We note[1] $\alpha' \lessdot \alpha$ whenever there is k such that $\alpha' = \alpha[k]$, or if $\alpha' + 1 = \alpha$.

The adjective "canonical" will be implicit for the rest of this section.

Example 3.4.3. The fundamental sequence of ω is the sequence of strictly positive integers. The sequence of ω^ω is therefore $(\omega, \omega^2, \omega^3, \dots)$, whereas the sequence of ω^2 is $(\omega, \omega.2, \omega.3, \dots)$. Here is an example of successively related ordinals.

$$0 \lessdot 1 \lessdot \omega \lessdot \omega + 1 \lessdot \omega.2 \lessdot \omega^2 \lessdot \omega^\omega \dots$$ ▲

Taking the transitive closure of this relation gives back the original order, so there is no information loss.

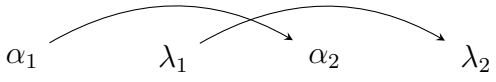
Proposition 3.4.4. *The transitive closure of $\lessdot$ is $<$.*

Proof. Let $0 \leq \lambda_1 \leq \lambda_2$ be two ordinals. We prove that $\lambda_1 \lessdot^k \lambda_2$ for some finite k by induction on λ_2. If λ_2 is a successor, consider λ_2' such that $\lambda_2' + 1 = \lambda_2$, so $\lambda_2' \lessdot \lambda_2$ and $\lambda_1 \leq \lambda_2'$. Otherwise, since the fundamental sequence of λ_2 bounds all smaller ordinals, there is a smallest n such that $\lambda_1 \leq \lambda_2[n] \lessdot \lambda_2$, so let $\lambda_2' = \lambda_2[n]$. In both cases, by induction $\lambda_1 \lessdot^{k'} \lambda_2'$ and thus $\lambda_1 \lessdot^{k'+1} \lambda_2$. □

Moreover, the relation is *crossing-free* as described below, which is a helpful technical tool.

Proposition 3.4.5. *If $\alpha_1 < \lambda_1 < \alpha_2$, $\alpha_1 \lessdot \alpha_2$ and $\lambda_1 \lessdot \lambda_2$, then $\lambda_2 \leq \alpha_2$.*

To put it more simply, this is the forbidden case :



Proof. We proceed by induction on $\alpha_1 = \beta + \omega^\gamma$, $\gamma < \alpha_1$. Note that $\alpha_1 + 1 < \alpha_2$. According to the definition of $\lessdot$, α_1 is in the fundamental sequence of α_2, which leaves two distinct cases. We suppose $\lambda_1 + 1 < \lambda_2$, otherwise the lemma is trivially true.

[1] Unlike in [Bra09], we adopt the notation $\lessdot$ instead of $\prec$, in order to avoid confusion with the suborder relation $\preceq$ between linear orders.

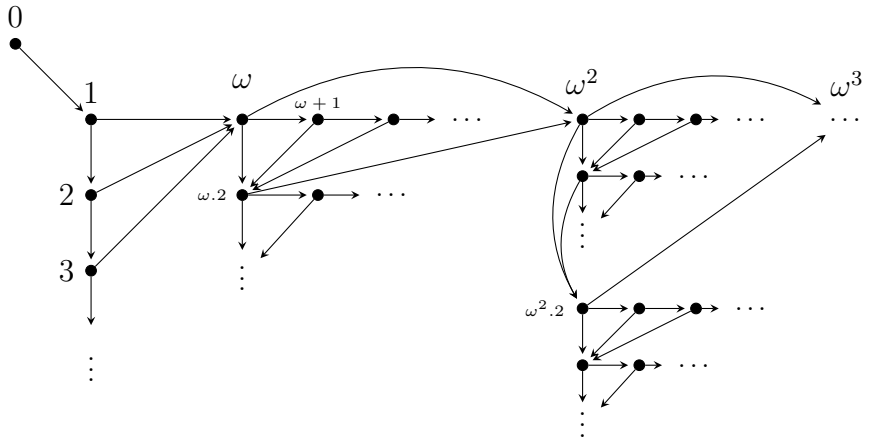


Figure 3.6: Covering graph of ω^ω.

Recall that $\hat{\beta}$ denotes the RCNF of β. In the first case, $\alpha_2 = \hat{\beta} + \omega^{\gamma+1}$. Then, in RCNF, $\lambda_1 = \hat{\beta} + \omega^\gamma.c_1 + \hat{\delta}_1$, $c_1 > 0$ and $\delta_1 < \omega^\gamma$. Now if $\delta_1 = 0$, then $c_1 > 1$ and $\lambda_2 = \hat{\beta} + \omega^{\gamma+1} = \alpha_2$. If $\delta_1 \neq 0$, note that the only part that changes between an ordinal and a member of its fundamental sequence is the last term in RCNF. So λ_2 is written $\hat{\beta} + \omega^\gamma.c_2 + \hat{\delta}_2$ in RCNF with $\delta_2 < \omega^\gamma$, and therefore $\lambda_2 < \alpha_2$.

In the second case, $\alpha_2 = \hat{\beta} + \omega^{\gamma'}$ with $\gamma \lessdot \gamma'$, $\gamma + 1 < \gamma'$. In RCNF, $\lambda_1 = \hat{\beta} + \omega^{\mu_1}.c_1 + \hat{\delta}_1$ with $c_1 > 0$, $\delta_1 < \omega^{\mu_1}$, $\gamma \leq \mu_1 < \gamma'$ and at least one of the following is true : $\delta_1 \neq 0$, or $\gamma < \mu_1$, or $c_1 > 1$. Again, we have to deal with several cases.

Either $\delta_1 = 0$ and $\gamma = \mu_1$; then $c_1 > 0$ and $\lambda_2 = \hat{\beta} + \omega^{\gamma+1} < \alpha_2$.

Or $\delta_1 = 0$ and $\gamma < \mu_1$; then $\lambda_2 = \hat{\beta} + \omega^{\mu_2}$ and $\mu_1 \lessdot \mu_2$; this is where the induction property is applied to get $\mu_2 \leq \gamma'$, and $\lambda_2 \leq \alpha_2$.

Finally, if $\delta_1 \neq 0$, as before $\lambda_2 = \hat{\beta} + \omega^{\mu_1}.c_2 + \hat{\delta}_2 < \hat{\beta} + \omega^{\gamma+1} < \alpha_2$. □

3.4.2 Covering graphs

Definition. Let $\mathcal{G}_\alpha = \{\lambda_1 \lessdot \lambda_2 \,|\, \lambda_1, \lambda_2 < \alpha\}$ be the graph of successor and fundamental sequence relation, or *covering graph* of the ordinal α.

For instance, a representation of $\mathcal{G}_{\omega^\omega}$ is given in Figure 3.6.

We first remark the finite out-degree of the covering graphs.

Proposition 3.4.6. *For any $\omega \uparrow\uparrow (n-1) < \alpha \leq \omega \uparrow\uparrow n$ and $n > 0$, the out-degree of $\mathcal{G}_\alpha$ is n.*

Proof. The output degree of $\lambda < \alpha$ in $\mathcal{G}_\alpha$ is the cardinal of $\{\mu \,|\, \lambda \lessdot \mu < \omega \uparrow\uparrow n\}$. If $n = 0$, $\lambda = 0$ and $\omega \uparrow\uparrow 0 = 1$, so the set is empty. For $n > 0$, let $\lambda = \beta + \omega^\gamma$ and $\lambda \lessdot \mu$, then either $\mu = \lambda + 1$ or $\mu = \beta + \omega^{\gamma'}$ with $\gamma \lessdot \gamma'$. Since $\gamma' < \omega \uparrow\uparrow (n-1)$, by induction $|\{\gamma' \,|\, \gamma \lessdot \gamma' < \omega \uparrow\uparrow (n-1)\}| \leq n-1$, which leads to $|\{\mu \,|\, \lambda \lessdot \mu < \omega \uparrow\uparrow n\}| \leq n$.

For the lower bound, if $n = 1$, then $\alpha \in [2, \omega]$, and $0 \lessdot 1$ has degree 1. For $n > 1$, if $\alpha > \omega \uparrow\uparrow (n-1)$ then

$$\begin{aligned} \omega \uparrow\uparrow (n-2) &\lessdot \omega \uparrow\uparrow (n-2) + 1 \\ &\lessdot \omega^{\omega \uparrow\uparrow (n-3)+1} \\ &\dots \\ &\lessdot \omega^{\dots^{\omega+1}} \\ &\lessdot \omega^{\dots^{\omega^2}} \\ &\lessdot \omega \uparrow\uparrow (n-1). \end{aligned}$$

For instance, let $n = 3$, $\alpha = \omega^\omega + 1$:

$$\begin{aligned} \omega &\lessdot \omega + 1 \\ &\lessdot \omega^2 \\ &\lessdot \omega^\omega. \end{aligned}$$

So $\omega \uparrow\uparrow (n-2)$ has degree n in $\mathcal{G}_\alpha$. □

In the following, we refine this property to get a characterisation of an ordinal covering by the degree of its vertices. We define the *degree word* $\mathfrak{d}(\alpha)$ of a covering graph as follows.

Definition. Consider the *greatest sequence* σ of $\mathcal{G}_\alpha$ starting from 0, i.e. $\sigma_0 = 0$ and for $k \geq 0$, σ_{k+1} is the greatest ordinal smaller that α such that $\sigma_k \lessdot \sigma_{k+1}$ if any. The previous lemma ensures that $\{\lambda \mid \sigma_k \lessdot \lambda\}$ is finite, so σ_{k+1} exists. Such a sequence may be finite.

The degree word $\mathfrak{d}(\alpha)$ is a finite or infinite word over $[0, n]$ when $\alpha \leq \omega \uparrow\uparrow n$, and its k^{th} letter is the out-degree of σ_k in $\mathcal{G}_\alpha$. Note that $\mathfrak{d}(0) = \varepsilon$ and $\mathfrak{d}(1) = 0$.

Example 3.4.7. Consider $\mathfrak{d}(\omega^\omega)$. Its greatest sequence is $(0, 1, \omega, \omega^2, \omega^3, \dots)$, where all have degree 2 in $\mathcal{G}_{\omega^\omega}$ except the first; so $\mathfrak{d}(\omega^\omega) = 12^\omega$. Now consider $\mathfrak{d}(\omega^3 + \omega^2)$: the sequence is now

$$0, 1, \omega, \omega^2, \omega^3, \omega^3 + 1, \omega^3 + \omega, \omega^3 + \omega + 1, \dots$$

which loops into $(\dots, \omega^3 + \omega.k, \omega^3 + \omega.k + 1, \dots)$ so $\mathfrak{d}(\omega^3 + \omega^2) = 12221(21)^\omega$. ▲

These examples hint that the degree word is regular.

Lemma 3.4.8. *For any $\alpha \leq \omega \uparrow\uparrow n$, if α is a successor ordinal then $\mathfrak{d}(\alpha)$ is a finite word of $[0, n]^*$; otherwise $\mathfrak{d}(\alpha)$ is an ultimately periodic word of $[1, n]^\omega$.*

Proof. Lemma 3.4.6 ensures that the degree word of $\alpha \leq \omega \uparrow\uparrow n$ is a word on the alphabet $[1, n]$. Since the transitive closure of $\mathcal{G}_\alpha$ is isomorphic to α, the greatest sequence σ of $\mathcal{G}_\alpha$ is unbounded, i.e. $\forall \lambda < \alpha, \exists n (\sigma_n \geq \lambda)$. In particular, if $\alpha = \lambda + 1$, there is n such that $\sigma_n = \lambda$, and the sequence is finite. The last element has out-degree 0.

If α is a limit ordinal, each $\alpha[n]$ must be in σ. Indeed, let m be such that $\sigma_m \leq \alpha[n] \leq \sigma_{m+1}$; if the inequalities are strict, since $\alpha[n] \lessdot \alpha$, by Proposition 3.4.5 we have $\sigma_{m+1} \geq \alpha$ which is a contradiction. So one of σ_m or σ_{m+1} must be $\alpha[n]$.

We want now to prove that the pattern between the $(\alpha[n])_{n<\omega}$ is always the same. Let $\alpha = \beta + \omega^\gamma$. As before, we have two cases. If $\gamma = \gamma' + 1$, then $\alpha[n] = \beta + \omega^{\gamma'}.(n+1)$. Given n, there is a path in the greatest sequence

$$\alpha[n] \lessdot \alpha[n] + \delta_1 \lessdot \cdots \lessdot \alpha[n] + \delta_h \lessdot \alpha[n+1]$$

with $\delta_i < \omega^\gamma$ for each i, and in fact δ_{i+1} is the greatest such that $\delta_i \lessdot \delta_{i+1}$ and $\delta_{i+1} \leq \omega^\gamma$. This defines the (δ_i) sequence independently of n. If i is fixed and n varies, $\alpha[n] + \delta_i \lessdot \alpha[n] + x$ whenever $\delta_i < x$ and $x \leq \omega^\gamma$, so the degree is still the same. The degree word is therefore ultimately periodic.

In the second case, $\alpha[n] = \beta + \omega^{\gamma[n]}$ and $\gamma[n] + 1 \leq \gamma[n+1] < \gamma$. So $\beta + \omega^{\gamma[n]+1} < \alpha$. Since $\alpha[n] \lessdot \beta + \omega^{\gamma[n]+1}$, then the following element of $\alpha[n]$ in the greatest sequence is greater or equal to $\beta + \omega^{\gamma[n]+1}$ and is therefore of the form $\beta + \omega^{\delta_1}$ with $\gamma[n] \lessdot \delta_1$. In general

$$\alpha[n] = \beta + \omega^{\gamma[n]} \lessdot \beta + \omega^{\delta_1} \lessdot \cdots \lessdot \beta + \omega^{\delta_h} \lessdot \alpha[n+1]$$

are in the greatest sequence. Then $\gamma[n], \delta_1, \ldots, \gamma[n+1]$ are in the greatest sequence of γ and their output degrees are respectively the same than those of $\omega^{\gamma[n]}, \omega^{\delta_1}, \ldots, \omega^{\gamma[n+1]}$ in α, minus 1. By induction, if the sequence of γ is ultimately periodic, so is the sequence of α. □

The degree word is actually a morphism from ordinals to words over the alphabet $[0, n]$ by lexicographic order.

Lemma 3.4.9. *If $\alpha < \alpha' \leq \omega \uparrow\uparrow n$, then $\mathfrak{d}(\alpha) <_{\text{lex}} \mathfrak{d}(\alpha')$.*

Proof. For any $\beta > 1$, $\mathfrak{d}(0) < \mathfrak{d}(1) < \mathfrak{d}(\beta)$. Suppose therefore $n > 0$. As before, note that the greatest sequence is unbounded, and that $\sigma_0 = \sigma'_0 = 0$. Thus if $0 < \alpha < \alpha'$ and σ' is the greatest sequence of $\mathcal{G}_{\alpha'}$, there is a smallest $n > 0$ such that $\sigma_n \neq \sigma'_n$, or σ_n doesn't exist whereas σ'_n does. In both cases, the output degree of σ_{n-1} is less in $\mathcal{G}_\alpha$ than in $\mathcal{G}_{\alpha'}$, so $\mathfrak{d}(\alpha) <_{\text{lex}} \mathfrak{d}(\alpha')$. □

A ultimately periodic pattern can be captured by a monadic formula. This is the goal of the the following lemma.

Lemma 3.4.10. *For each finite or infinite word w over $[0, n]$ and a given ordinal α, there is a monadic formula φ^w such that $\mathcal{G}_\alpha \models \varphi^w$ iff $w = \mathfrak{d}(\alpha)$.*

Proof. The fact that the degree word is finite or ultimately periodic permits to use a finite number of variables. We consider the ultimately periodic case, and $\mathfrak{d}(\alpha) = uv^\omega$.

To simplify the writing, we consider the following shortcuts :

- $\tau(p, q)$ if q is the greatest such that $p \lessdot q$;
- $\partial_k(p)$ if the output degree of p is k;

- root(X, p) and end(X, p) when p is co-accessible (resp. accessible) from each vertex of X, with the entire path in X; root(p) looks for a root of the whole graph;
- inline(X) if the subgraph of support X is a finite or infinite path;
- size$_k(X)$ when $|X| = k$.

All these notations stand for monadic formulæ . For instance, the inline(X) property is true when there is a root in X and each vertex has output degree 1, and each except the root has input degree 1.

Now we may write the formula φ^w. For this, we need two finite sets $p_1 \dots p_{|u|} \in U$ for the static part, $q_1 \dots q_{|v|} \in V'$ for the beginning of the periodic part and an infinite set V with $V' \subseteq V$.

$$\begin{aligned}
\varphi^w \;:=\; & \exists p_1, \dots, p_{|u|}, V, q_1, \dots, q_{|v|} \in V \;: \\
& \mathrm{root}(p_1) \wedge \left(\bigwedge_{i=1}^{|u|-1} \tau(p_i, p_{i+1}) \wedge \partial_{u_i}(p_i) \right) \wedge \tau(p_{|u|}, q_1) \wedge \partial_{u_{|u|}}(p_{|u|}) \\
& \wedge \mathrm{root}(V, q_1) \wedge \left(\bigwedge_{i=1}^{|v|-1} \tau(q_i, q_{i+1}) \wedge \partial_{v_i}(q_i) \right) \wedge \partial_{v_{|v|}}(q_{|v|}) \\
& \wedge \mathrm{inline}(V) \wedge \forall q \in V, \exists X \subseteq V, q' \in X \;: \\
& \qquad \mathrm{inline}(X) \wedge \mathrm{size}_{|v|+1}(X) \wedge \mathrm{root}(X, q) \wedge \mathrm{end}(X, q') \\
& \qquad \wedge \left(\bigwedge_{k \leq n} \partial_k(q) \Rightarrow \partial_k(q') \right)
\end{aligned}$$

We check that p_1 is the general root 0, and q_1 the root of V, which is an infinite path. Formulæ τ and ∂_k force the degree of the uv part. For the periodic part, each $q \in V$ must be the root of a finite path $X_q \subseteq V$ of size $|v| + 1$, which end has the same degree that q. □

The combination of Lemmas 3.4.9 and 3.4.10 yields the following theorem.

Theorem 3.4.11. *For any ordinals $\alpha \neq \alpha'$ smaller than ε_0,* $\mathrm{MTh}(\mathcal{G}_\alpha) \neq \mathrm{MTh}(\mathcal{G}_{\alpha'})$.

This result is the central point of this section, because it it opposed to the Theorem 3.4.2 on ordinals : the family of covering graphs has the MSO-discernability property. One must take care to the fact that here $\mathcal{G}_\alpha$ and $\mathcal{G}_{\alpha'}$ are implicitely supposed to be covering graphs. Whether $\mathrm{MTh}(\mathcal{G}_\alpha)$ is unique among all $\{\lessdot\}$-graphs (up to isomorphism) is open, and is conjectured false.

A whole set of properties could be tested on covering graphs. For instance, the MSO selection property is known to fail for ordinals greater than ω^ω [RS08]. It would be interesting to know if covering graphs can also raise this limit.

As a consequence of Theorem 3.4.11, there is no generic monadic interpretation from an ordinal greater than ω^ω to its covering graph. Below this limit, there is an interpretation, because it is possible to distinguish successive limit ordinals, as in Proposition 3.4.1.

3.4.3 Other properties of covering graphs

We study here other monadic properties of covering graphs and remark that they also belong to the hierarchy. First of all, by Proposition 3.4.4, there is a monadic interpretation $\mathcal{I}$ such that $\mathcal{I}(\mathcal{G}_\alpha) = \alpha$. In fact, we have much more : all ordinals smaller than α are interpretable in $\mathcal{G}_\alpha$.

Proposition 3.4.12. *For any $\alpha < \beta < \varepsilon_0$, there is a MSO interpretation $\mathcal{I}$ such that $\mathcal{G}_\alpha = \mathcal{I}(\mathcal{G}_\beta)$.*

Proof. Following the definition, we look for an interpretation $\mathcal{I} = \{\psi_\lessdot\}$. We use again the fact that the degree word is unique and MSO-definable. Defining the greatest sequence of $\mathcal{G}_\alpha$ provides a MSO marking on $\mathcal{G}_\beta$, which bounds the set of vertices. More precisely, let $\Psi^w(p)$ be an expression similar to φ^w of Lemma 3.4.10 but where the part $\tau(p_i, p_{i+1}) \wedge \delta_{u_i}(p_i)$ has been replaced by $\tau_{u_i}(p_i, p_{i+1})$ meaning "p_{i+1} is the u_i^{th} such that $p_i \lessdot p_{i+1}$"; the same goes for the q_j and for $\tau(p_{|u|}, q_1) \wedge \delta_{u_{|u|}}(p_{|u|})$. Also add the condition that p is a part of the sequence : $(\bigvee_i p = p_i) \vee p \in V$. Then $\Psi^w(p)$ is a marking of the greatest sequence associated to w. For a given α, $\mathcal{I}$ simply adds the condition of co-accessibility to a vertex marked by $\Psi^{\mathfrak{d}(\alpha)}$.

$$\psi_\lessdot(p, q) := p \xrightarrow{\lessdot} q \wedge \exists r \, (\Psi^{\mathfrak{d}(\alpha)}(r) \wedge q \xrightarrow{\lessdot^*} r)$$

Then $\mathcal{G}_\alpha = \mathcal{G}_\beta \cap \{p \xrightarrow{\lessdot} q \mid \exists r \, (\Psi^{\mathfrak{d}(\alpha)}(r) \wedge q \xrightarrow{\lessdot^*} r)\}$. □

We end this section by noting that covering graphs also are in the hierarchy.

Proposition 3.4.13. *If $\alpha < \omega \uparrow\uparrow (n+1)$, then $\mathcal{G}_\alpha \in \mathrm{Graph}_n$.*

Proof. For any finite α, $\mathcal{G}_\alpha$ is in fact a finite path labeled by $\lessdot$ and is in Graph_0. By Lemma 3.4.14 below iterated n times, every $\omega^{\cdots^{\omega^k}}$ is in Graph_n when there are n times ω and $1 < k < \omega$. Smaller ordinals are captured by a restriction as in Proposition 3.4.12. □

We note $p \xrightarrow{a^\bullet} q$ for the longest possible path labeled by a, and $p \xrightarrow{S} q$ a shortcut for the successor relation, i.e.

$$\begin{aligned} p \xrightarrow{a^\bullet} q &:= p \xrightarrow{a^*} q \wedge \neg\exists r \, (q \xrightarrow{a} r) \\ p \xrightarrow{S} q &:= p \xrightarrow{\lessdot} q \wedge \neg\exists r (p \xrightarrow{\lessdot} r \wedge r \xrightarrow{\lessdot^*} q). \end{aligned}$$

Now let $\mathcal{I} = \{\varphi_\lessdot\}$ and $\mathcal{M}(p)$ respectively be the interpretation and marking

$$p \xrightarrow{\bar{\lessdot}^\bullet\# + \bar{\#}^\bullet S\# + \bar{\#}\lessdot\#} q$$

$$\begin{aligned} \varphi_\lessdot(p,q) &:= M(p) \wedge M(q) \wedge p \xrightarrow{\bar{\lessdot}^\bullet\#} q \vee p \xrightarrow{\bar{\#}^\bullet S\#} q \vee p \xrightarrow{\bar{\#}\lessdot\#} q \\ M(p) &:= \exists r : \mathrm{root}(r) \wedge r \xrightarrow{\lessdot^*\#(\bar{\lessdot}^*\#)^*} p \end{aligned}$$

The marking $\mathcal{M}(p)$ allows to start anywhere on the root graph, but as soon as a #-arc has been followed, $\lessdot$-arcs can only be followed backwards. We consider only goals of a #-arc.

The $\varphi_\lessdot(p,q)$ formula states the relation on these vertices, leaving three choices : either to follow $\lessdot$-arcs as long as possible (in practice, until a copy of 0) and go down one #-arc; or on the contrary, to follow # backwards as long as possible, then take the successor and one #-arc; or just to follow one # backwards, one $\lessdot$ and one #.

Lemma 3.4.14. $\mathcal{G}_{\omega^\alpha} = \mathcal{I}(\mathrm{Treegraph}(\mathcal{G}_\alpha))$.

Proof. As stated in Corollary 2.2.2, ω^α is isomorphic to the set of decreasing sequences of ordinals smaller than α in lexicographic order. Let $T = \mathrm{Treegraph}(\mathcal{G}_\alpha)$; the 0 of the root graph is still the only root, we call it r. Each $p \in V_T$ marked by $\mathcal{M}$ can be mapped into a decreasing sequence. If $r \xrightarrow{\lessdot^*\#(\bar{\lessdot}^*\#)^*} p$, then there is a finite sequence $(p_i)_{i\leq k}$ such that $r \xrightarrow{\lessdot^*\#} p_0$, $p_i \xrightarrow{\bar{\lessdot}^*\#} p_{i+1}$ for $i < k$ and $p_k = p$, with the same properties as in the proof of Theorem 3.1.2. Each p is thus mapped to an ordinal $\beta_p < \omega^\alpha$.

The interpretation $\varphi_\lessdot$ provides the relation to make this bijection an isomorphism. Let $G = \varphi_\lessdot \circ \mathrm{Treegraph}(\mathcal{G}_\alpha)$. We distinguish the three cases of the definition of $\lessdot$.

- If $p \xrightarrow{\bar{\lessdot}^\bullet\#} q$, then q is mapped to $(\gamma_0, \ldots, \gamma_k, 0)$. This is the successor case $\beta_p + 1 = \beta_q$.
- If $p \xrightarrow{\bar{\#}^\bullet S\#} q$, then let l be the smallest integer such that $\gamma_l = \gamma_{l+1} = \cdots = \gamma_k$. Then q is mapped to $(\gamma_0, \ldots, \gamma_{l-1}, \gamma_l + 1)$. This corresponds to the case $\beta_p = \beta + \omega^{\gamma_l}.(k-l) \lessdot \beta + \omega^{\gamma_l+1}$.
- If $p \xrightarrow{\bar{\#}\lessdot\#} q$, then $q \mapsto (\gamma_0, \ldots, \gamma_{k-1}, \gamma)$ with $\gamma_k \lessdot \gamma$. The marking $\mathcal{M}$ ensures that q is mapped to a decreasing sequence. This is the recursive case, where $\beta_p = \beta + \omega^{\gamma_k}$, $\beta_p = \beta + \omega^{\gamma'_k}$ and $\gamma_k \lessdot \gamma'_k$. □

Example 3.4.15. Consider $\mathcal{G}_\omega$, which is an infinite path. A representation of its treegraph is given in Figure 3.7 (plain lines for $\lessdot$, dotted lines for #). The white labeled vertices are the ones marked by $\mathcal{M}$ and therefore they are the only ones kept by the interpretation $\mathcal{I}$. We are allowed to go anywhere on the root $\mathcal{G}_\omega$ structure, but as soon as we follow # we can only go backwards. This reflects the construction of an ordinal smaller than ω^α as a decreasing sequence of ordinals : the first one is any ordinal smaller than α, but afterwards we only may decrease. ▲

Covering graphs therefore enjoy the properties of graphs of the hierarchy.

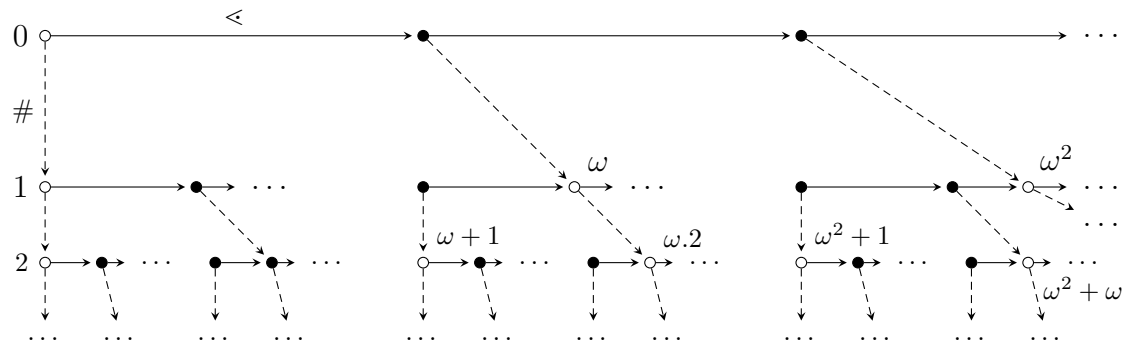


Figure 3.7: Exponentiation of the covering graph of ω.

For illustrations of the three cases in the proof of Lemma 3.4.14, we have

$$\omega^2 \xrightarrow{\bar{\lessdot}^\bullet \#} \omega^2 + 1, \qquad \omega^2 + 1 \xrightarrow{\bar{\#} \lessdot \#} \omega^2 + \omega, \qquad 2 \xrightarrow{\bar{\#}^\bullet \lessdot \#} \omega.$$

Corollary 3.4.16. *For any $\alpha < \varepsilon_0$, the monadic theory of $\mathcal{G}_\alpha$ is decidable.*

3.4.4 Strictness of covering graphs in the hierarchy

We will prove in Theorem 4.3.8 that ordinals greater than or equal to ε_0 do not belong to the hierarchy. As a consequence, the same is true for the associated covering graphs. We present here another version of this result. It is based on Theorem 4.5.3 which will be seen in the next chapter. It is actually a pretext to use logical properties of covering graphs, in which it is possible to interpret other separating graphs. In particular, for any function $f : \mathbb{N} \mapsto \mathbb{N}$, let $\mathcal{K}_f$ be the prefix tree defined by the following set of leaves : $\text{Leaves}(\mathcal{K}_f) = \{1^i 0^{f(i)}\}$.

Let the $\exp(m, n, i)$ operation be defined by

$$\begin{aligned} \exp(m, 0, i) &= i \\ \exp(m, n+1, i) &= m^{\exp(m,n,i)}, \end{aligned}$$

Theorem 4.5.3 shows that $\mathcal{K}_{\exp(2,n,\cdot)}$ is not in Graph_n. We use this fact for the following result.

Proposition 3.4.17. *If $\omega \uparrow\uparrow (n+1) \leq \alpha \leq \varepsilon_0$, then $\mathcal{G}_\alpha \notin \text{Graph}_n$.*

The proof is separated in several lemmas. Finding a monadic interpretation from $\mathcal{G}_\alpha$ to $\mathcal{K}_{\exp(2,n,\cdot)}$ is enough to prove $\mathcal{G}_\alpha \notin \text{Graph}_n$. But in fact, Proposition 3.4.12 already states that if $\omega \uparrow\uparrow n+1 \leq \alpha \leq \varepsilon_0$, then there is an interpretation from $\mathcal{G}_\alpha$ to $\mathcal{G}_{\omega\uparrow\uparrow n+1}$; so the interpretation from $\mathcal{G}_{\omega\uparrow\uparrow n+1}$ to $\mathcal{K}_{\exp(2,n,\cdot)}$ is enough. We build this interpretation.

First of all, let us select the ordinals which stand in the k^{th} finite subtree. Let C_n^k be the set of ordinals smaller than $\exp(\omega, n, k)$ where each coefficient in RCNF is at most 1, except for the top-most power :

- $[0, k-1] \in C_0^k$ for $k > 0$,
- $0 \in C_n^k$,
- if $\gamma_0, \ldots, \gamma_h$ are all distinct ordinals of C_{n-1}^k, then $\omega^{\gamma_0} + \cdots + \omega^{\gamma_h} \in C_n^k$.

Example 3.4.18. For instance,

$$\begin{aligned} C_1^2 &= \{0, 1, \omega, \omega+1\} \\ C_1^3 &= \{0, 1, \omega, \omega+1, \omega^2, \omega^2+1, \omega^2+\omega, \omega^2+\omega+1\} \\ C_2^2 &= \{0, 1, \omega, \omega+1, \omega^\omega, \omega^\omega+1, \omega^\omega+\omega, \omega^\omega+\omega+1, \\ &\quad \omega^{\omega+1}, \omega^{\omega+1}+1, \omega^{\omega+1,}+\omega, \omega^{\omega+1}+\omega+1, \\ &\quad \omega^{\omega+1}+\omega^\omega, \omega^{\omega+1}+\omega^\omega+1, \omega^{\omega+1}+\omega^\omega+\omega, \omega^{\omega+1}+\omega^\omega+\omega+1\}. \end{aligned}$$

▲

The following lemma is only a matter of cardinality of powersets.

Lemma 3.4.19. *The cardinality of the set C_n^k is* $\exp(2, n, k)$.

We abusively note $\alpha + C_n^k$ for the set $\{\alpha + \gamma \mid \gamma \in C_n^k\}$. The main difficulty of this section is to define this set with MSO logic.

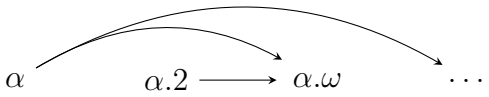
Lemma 3.4.20. *There is a monadic formula $\varphi(x, y)$ such that for all $n > 0$, $\varphi(\exp(\omega, n, k), y)$ is satisfied by $x \in \exp(\omega, n, k) + C_n^k$ in the covering graph of an ordinal greater than* $\exp(\omega, n, k).2$.

Proof. For each ordinal α, we define a sequence S_α of ordinals. We note $\tau(\alpha)$ the greatest β such that $\alpha \lessdot \beta$.

- $\alpha \in S_\alpha$, $\alpha + 1 \in S_\alpha$,
- if $\lambda \in S_\alpha$ and $\alpha < \lambda \lessdot \gamma$, then $\gamma \in S_\alpha$ unless $\exists \lambda' \leq \lambda$ such that $\lambda' \in S_\alpha$ and $\lambda' \lessdot \tau(\gamma)$.

There is a monadic formula $\varphi(\alpha, y)$ which is satisfied exactly by $y \in S_\alpha$. The rest of the proof aims at showing that the set $S_{\exp(\omega,n,k)}$ is $\exp(\omega, n, k) + C_n^k$ in the covering graph of an ordinal greater than $\exp(\omega, n, k).\omega$. As a starting point, note that if $\lambda \in S_\alpha$, then there is a path of vertices of S_α (labeled by $\lessdot$) from α to λ.

Let $\alpha = \exp(\omega, n, k)$. First of all, $\tau(\alpha.2) = \alpha.\omega$ and $\alpha \lessdot \alpha.\omega$ so $\alpha.2 \notin S_\alpha$. By Proposition 3.4.5, any path from α to an ordinal of $[\alpha.2, \omega^{\exp(\omega,n-1,k)+1})$ goes through $\alpha.2$, which is not in S_α. Also, paths from α to ordinals of $[\alpha.\omega, \exp(\omega, n, k+1))$ necessarily go through $\alpha.\omega$ which is not in S_α because $\alpha \lessdot \alpha.\omega$, so $S_\alpha \cap [\alpha.2, \exp(\omega, n, k+1)) = \emptyset$. Eventually the fundamental sequence of $\exp(\omega, n, k+1)$ is $(\alpha, \exp(\omega, n-1, \omega.2), \exp(\omega, n-1, \omega.3), \ldots)$, all of which but α are not in S_α. To sum up, S_α cannot contain ordinals greater or equal to $\alpha.2$.



Let $\lambda \in [\alpha, \alpha.2[$, $\lambda = \hat{\beta} + \omega^\gamma.c + \hat{\eta}$ in RCNF with $c > 1$ (recall that the notation $\hat{\beta}$ is used to note the RCNF of β). By Lemma 3.4.5 again, any path from α to $\alpha + \lambda$ goes through

$$\begin{array}{rcl} & \lambda' & = & \hat{\beta} + \omega^\gamma \\ \text{and} & \lambda'' & = & \hat{\beta} + \omega^\gamma.c \end{array}$$



But then $\lambda' \lessdot \beta + \omega^{\gamma+1} = \tau(\lambda'')$ when $c > 1$, so $\lambda'' \notin S_\alpha$.

Recursively, we suppose that any path from $\exp(\omega, n-1, k)$ to $\gamma \notin C^k_{n-1}$ with $\exp(\omega, n-1, k) \leq \gamma < \exp(\omega, n-1, k).2$ goes through γ' and γ'', with $\gamma' \lessdot \tau(\gamma'')$. Then if $\lambda = \hat{\beta} + \omega^\gamma + \hat{\eta}$, define

$$\begin{array}{rcl} & \lambda' & = & \hat{\beta} + \omega^{\gamma'} \\ \text{and} & \lambda'' & = & \hat{\beta} + \omega^{\gamma''} \end{array}$$

which propagates the property to level n. All this proves that if $\lambda = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\gamma_j}$ in CNF and $\lambda \in S_\alpha$, then all γ_i are distinct and are in C^k_{n-1}. Therefore $S_\alpha \subseteq \alpha + C^k_n$.

For the other side, let $\lambda \in \alpha + C^k_n$. If $\lambda = \alpha$ the case is done, otherwise

$$\lambda = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\gamma_h}$$

with each $\gamma_i \in C^k_{n-1}$.

We have to prove that $\exists \lambda' \lessdot \lambda$ in $\alpha + C^k_n$. By induction on n with $\alpha = 0$, for $\gamma_h > 0$, $\exists \gamma'_h \lessdot \gamma_h$ in C^k_{n-1}, so $\lambda' = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\gamma'_h}$ answers to the request (since the γ_i are decreasing, γ'_h is still distinct from the γ_i,$i < h$). If $\gamma_h = 0$, then we take $\lambda' = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\gamma_{h-1}}$.

If not, now $\tau(\lambda) = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\tau(\gamma_h)}$. If $\lambda' \in S_\alpha$ is such that $\lambda' \lessdot \tau(\lambda)$, then $\lambda' = \alpha + \omega^{\gamma_0} + \cdots + \omega^{\gamma_{h-1}} + \omega^\gamma$ for some $\gamma \in \gamma_h \cap C^k_{n-1}$, but then by induction we never have $\gamma \lessdot \tau(\gamma_h)$, which is a contradiction. □

Lemma 3.4.21. *The greatest sequence of $\omega \uparrow\uparrow (n+1)$ is ultimately $(\exp(\omega, n, k))_{k \geq 1}$.*

Proof. This is a corollary of the proof of Lemma 3.4.8, since as in the previous proof $\exp(\omega, n, k) \lessdot \exp(\omega, n, k+1)$. □

We are now ready to prove the main result of this section.

Proof of Prop. 3.4.17. We concatenate the previous lemmas. Recall that $\mathcal{K}_{\exp(2,n,\cdot)}$ is the prefix tree of leaves $\{1^k 0^{\exp(2,n,k)}\}$.

We have seen in Prop. 3.4.12 that the greatest sequence of α in selectable in $\mathcal{G}_\alpha$. Since we work in $\mathcal{G}_{\omega\uparrow\uparrow n+1}$, by Lemma 3.4.21 we can restrict ourselves to the sequence $(\exp(\omega, n, k))_{k\geq 1}$, which will be the "horizontal path" of $\mathcal{K}_{\exp(2,n,\cdot)}$. Let $\varphi_1(x, x')$ be the formula satisfied if x, x' are part of this sequence and $x \lessdot x'$.

Let $\varphi(x, y)$ be as defined by Lemma 3.4.20. Note that φ does not depend on k. If $x = \exp(\omega, n, k)$ for some k, then the set $\{y \mid \varphi(x, y)\}$ has cardinal $\exp(2, n, k)$. Moreover, this set is already ordered by $\lessdot^*$, so we may find a formula $\varphi_0(y, y')$ restricted to set and satisfied by $(y, y'$ such that $y \lessdot^* y'$ and there is no z between y and y'. We have therefore the "vertical path" hanging from $\exp(\omega, n, k)$ and of length $\exp(2, n, k)$.

The interpretation $\mathcal{I}_n = \{\varphi_0, \varphi_1\}$ maps therefore $\mathcal{G}_{\omega\uparrow\uparrow n+1}$ into $\mathcal{K}_{\exp(2,n,\cdot)}$. In fact, for any $\omega \uparrow\uparrow n + 1 \leq \alpha < \varepsilon_0$, $\mathcal{I}_n$ maps $\mathcal{G}_\alpha$ into $\mathcal{K}_{\exp(2,n,\cdot)}$. □

3.4.5 The case of $\mathcal{G}_{\varepsilon_0}$

The covering graph $\mathcal{G}_{\varepsilon_0}$ can be defined without changing the definition of fundamental sequence. It has unbounded degree, but has still the property of Proposition 3.4.12 : it can give any smaller ordinal via monadic interpretation. This yields the following corollary of Proposition 3.4.17.

Corollary 3.4.22. *$\mathcal{G}_{\varepsilon_0}$ does not belong to the pushdown hierarchy.*

It would be interesting to know more about the logical properties of this graph. This question is indeed tightly linked to the notion of a "limit operation" preserving decidability of MSO-theory. Some first ideas appear in [Tho08].

Conjecture 3.4.23. $\mathrm{MTh}(\mathcal{G}_{\varepsilon_0})$ *is decidable.*

Chapter 4

The structure of tree frontiers

The first question one may ask after Theorem 3.1.2 is the converse : if any ordinal smaller than $\omega \uparrow\uparrow n+1$ can be constructed at level n, is this classification is strict or not? For a start, can ω^ω be in Graph_1? For this first question, it is shown in [Del04, KRS05] that ω^ω cannot be a (word-)automatic ordinal, whereas structures in Graph_1 are. On the other hand, one can consider the orders composed by frontier of trees, as in [Tho86, Cou78]. At the second level, [BÉ10] considers frontiers of algebraic trees to find out that such ordinals are smaller than ω^{ω^ω}. The same paper conjectures that frontiers of trees in the hierarchy yield successive $\omega \uparrow\uparrow n$.

In order to follow this presentation, we first need to reduce ourselves to frontiers. Hence a first result of this chapter is that orders of Graph_n are exactly frontiers of Tree_n. By the result of [BÉ10], ordinals of Graph_2 are smaller than ω^{ω^ω}. In this chapter we work out apply a recursive argument and get the expected generalization for every level, in Theorem 4.3.8. In Section 4.4 we remark that the same result can be extended to general scattered orders, measured by Hausdorff rank. In parallel we study the related Cantor-Bendixson rank of these trees.

To obtain these results, Section 4.1 presents a particular version of monadic interpretation on deterministic trees, in the form of a automaton "walking" on the tree.

4.1 Tree-walking automaton

It is well-known that on deterministic trees MSO logic is captured by parity tree automata [Rab69]. This equivalence can be used to characterize the binary relations defined by MSO formulæ on such trees using a finite state automaton running on the tree. The *tree-walking automata* is not new [AU71], but they have mostly been used on finite trees [BC08, BC06b]; see [Boj08] for a survey. Here, we use these automata on infinite trees as a "weak form" of monadic interpretations.

Definition. A *non-deterministic tree-walking automaton* or simply *TWA* working on deterministic trees over Σ colored by Γ is a tuple $\mathcal{A} = (Q, q_0, F, \Delta)$ where Q is the finite

set of states, $q_0 \in Q$ is the initial state, F is the set of final states and Δ is the set of transitions. A transition is a tuple (p, c, q, a) with

- $p \in Q$ is the current state,
- $c \in \Gamma$ is the color of the current node,
- $q \in Q$ is the next state,
- $a \in \Sigma \cup \{\varepsilon, \uparrow\}$ is the action to perform. Intuitively ε corresponds to "staying in the current node", $\uparrow$ to "going to the parent node" and $d \in \Sigma$ corresponds to "going to the d-son".

A run of the automaton on a tree t is a sequence $(q_0, u_0) \dots (q_n, u_n)$ over $Q \times \mathrm{Dom}(t)$ where $q_n \in F$ and for all $i < n$, if $u_{i+1} = u_i a_i$ with $a_i \in \Sigma \cup \{\varepsilon\}$ then $(p_i, t(u_i), p_{i+1}, a_i) \in \Delta$, otherwise if $u_i \in u_{i+1}\Sigma$ then $(p_i, t(u_i), p_{i+1}, \uparrow) \in \Delta$. We say that $\mathcal{A}$ accepts the pair of nodes (u_0, u_n).

Remark 4.1.1. Note that the automaton is *a priori* not deterministic; [BC06b] shows that a TWA cannot generally be determinized, even when the tree is finite and the only run considered is from the root to the root. ■

We have restated in Proposition 2.4.3 that a monadic interpretation on a deterministic tree can be encoded in a rational inverse mapping up to monadic marking. But the actual proof of this definition is even stronger, as it constraints the rational mapping to the shortest path between two vertices. This leads to the following rephrasing of this result. A run is said to be *simple* if the sequence forms a simple path, *i.e.* each node is visited at most once.

Proposition 4.1.2 ([Car06, Prop. 3.2.1])**.** *For any deterministic tree t and any MSO-formula $\varphi(x, y)$, there exists an MSO-coloring $\mathcal{M}$ adding colors and a TWA $\mathcal{A}$ such that $t \models \varphi[u, v]$ if and only if $\mathcal{A}$ accepts (u, v) on $\mathcal{M}(t)$. In this case there exists a simple run from u to v.*

A TWA is therefore a "weak form" of a monadic interpretation. It means that up to a recoloring, the binary formula can be restricted to the smaller connex set containing the interpretations of the two free variables.

Remark 4.1.3. Conversely, for any simple TWA and coloring $\mathcal{M}$, there is a MSO formula φ such that (u, v) is accepted in $\mathcal{M}(t)$ iff $t \models \varphi[u, v]$ where φ only works on the simple path between u and v. In particular, if t' is the smallest subtree containing u and v, then $t' \models \varphi[u, v]$.

This can be directly seen by encoding the automaton in a logical formula. The formula defines two set U and V which are paths respectively from $u \wedge v$ to u and from $u \wedge v$ to v. Then each step of the automaton can be reproduced backward on U and forward on

V. More formally, there have to be sets $X_{q_0}, \ldots, X_{q_{|Q|}}$ representing states, with $u \in X_{q_0}$ and $v \in \bigcup_{q \in F} X_q$. For the U part, assuming we know that U is a path,

$$\forall x \in U \cap X_p \setminus \{u \wedge v\}, \exists y \in U \ : \ y \to x \wedge \bigvee_{q \in Q} \left(y \in X_q \Rightarrow \bigvee_{\delta(p,c)=(q,\uparrow)} (c, x) \right)$$

and for the V part,

$$\forall x \in V \cap X_p \setminus \{v\}, \exists y \in V \ : \ \bigvee_{a \in \Sigma} \left(x \xrightarrow{a} y \wedge \bigvee_{q \in Q} \left(y \in X_q \Rightarrow \bigvee_{\delta(p,c)=(q,a)} (c, x) \right) \right)$$

■

Remark 4.1.4. The equivalence between a MSO formula and recognizability by TWA allows us to define another kind of interpretation as in Section 2.4.1. The *TWA-interpretation* on a deterministic Σ-tree is a tuple $(\mathcal{M}, (\mathcal{A}_a)_{a \in \Gamma})$ where each $\mathcal{A}_a$ is a TWA on deterministic Σ-trees. The result of this interpretation on a Σ-tree t is the graph

$$\{x \xrightarrow{a} y \mid a \in \Gamma, \mathcal{A}_a \text{ accepts } (x, y) \text{ in } \mathcal{M}(t)\}.$$

It follows by Proposition 4.1.2 that graphs of Graph_n are exactly TWA-interpretations of deterministic trees of Tree_n. ■

4.2 From graphs to frontiers

In this section, we consider the linear orders defined by frontiers of deterministic trees, *i.e.* the leaves of a deterministic tree under lexicographic ordering as defined in Section 2.1.4. Example 2.4.1 shows that the frontier of a (colored) deterministic tree in t can be defined in t using an MSO-interpretation. Hence the frontiers of the (resp. colored) deterministic trees in Tree_n are (resp. colored) linear orders in Graph_n. This section proves that the converse inclusion holds : any order of Graph_n can be seen as a frontier of a deterministic tree in Tree_n. This is Theorem 4.2.6.

As a starting point, it is not difficult to restrict to some kind of normalized trees.

Proposition 4.2.1. *For any det. t, there is a full binary prefix tree t' with $\mathbf{Fr}(t) = \mathbf{Fr}(t')$. Moreover, if $t \in \text{Tree}_n$, then t' can be chosen in Tree_n.*

Proof. If the order is of type **1**, a tree reduced to a root is enough. Suppose otherwise. The first step is that we may only consider prefix trees by pruning. For the binary property, let t be prefix on a ranked alphabet Σ and let $\tau : \Sigma \to \{0,1\}^*$ a prefix binary mapping. For any $I \subseteq \Sigma$, let C_I be the finite tree of leaves $\{\tau(a) \mid a \in I\}$. We replace each node x which is not a leaf by $C_{\{a \mid xa \in \text{Dom}(t)\}}$, of root r_x, so that $r_{xa} = r_x.\tau(a)$ whenever $xa \in \text{Dom}(t)$.

The resulting tree is binary and has yields the same order. Finally, to get a full tree, we contract every path $1(1+0)^*$ (resp. $0(1+0)^*$)) without branching into 1-arcs (resp. 0-arcs).

To perform these operations within Tree_n, we work on the folded graph. The result is obvious for finite trees, so we will suppose that $n > 0$. Let $t \in \mathrm{Tree}_n$. We first use a MSO-marking μ to mark the prefix closure of leaves. Let $G \in \mathrm{Graph}_{n-1}, r \in V_G$ such that $\mu(t) = \mathrm{Unf}(G, r)$. The restriction to marked vertices accessible from r is an MSO-interpretation. Next, the operation substituting x with $C_{\{a \mid \exists y, x \xrightarrow[G]{a} y\}}$ is a MSO-transduction. Finally, the contraction of simple paths is again a MSO-interpretation. So all these operations give a graph still in Graph_{n-1}, which unfolding yields the same frontier than t. □

Finite orders are not a problem for us, so we start with $n \geq 1$, a finite set of colors Γ together with colored linear order $L : \langle D, <_L \rangle \mapsto \Gamma$ in Graph_n. This order is the MSO-interpretation $\mathcal{I}$ of a deterministic tree t in Tree_n. The following property recalls that we can suppose that D is exactly $\mathrm{Leaves}(t)$.

Proposition 4.2.2. *For any interpretation $\mathcal{I}$ and deterministic tree $t \in \mathrm{Tree}_n$, there is an interpretation $\mathcal{I}'$ and a det. binary prefix tree t' such that $\mathcal{I}(t) \simeq \mathcal{I}'(t')$ and vertices of $\mathcal{I}'(t')$ are exactly leaves of t'.*

Proof. In a deterministic t, we mark each vertex x considered by $\mathcal{I}$. In the folded graph G such that $\mathrm{Unf}(G, r) = t$, a monadic transduction $\mathcal{T}$ adds a #-arc from these vertices to a fresh vertex. It is then easy to adapt $\mathcal{I}$ to pick leaves of $\mathrm{Unf}(\mathcal{T}(G), r)$. By marking leaves targets of #-arcs and using Proposition 4.2.1, we get the required result. □

Using Propositions 2.5.4, 4.1.2 and 4.2.2, we have that there exists a deterministic prefix tree $t \in \mathrm{Tree}_n$ colored by Γ', a TWA $\mathcal{A}$ and a projection $\mu : \Gamma' \mapsto \Gamma$ such that D is isomorphic to the set of leaves of t, for all $u, v \in D$, $u <_L v$ iff $\mathcal{A}$ accepts (u, v) with a simple path and for all $u \in D$, $L(u) = \mu(t(u))$.

The construction of this section rearranges the leaves of t into a new deterministic tree $s(t)$ so that lexicographic order on $s(t)$ matches $<_L$ on D. For any leaf $u \in D$ and for $v \sqsubset u \in D$, we consider sequences $u_0 >_L u_1 >_L \ldots >_L u_k$ of leaves starting with $u_0 = u$ and such that for $i \in [0, k-1]$, $u_i \wedge u_{i+1} = v$. Intuitively, the u_i alternate from one subtree of v to the other. Using a pumping argument on $\mathcal{A}$, we can show that there exists $n_0 \geq 0$ such that for every leaf $u \in D$ and all node $v \sqsubset u$, these sequences have a length less that n_0. We may then define $s(u, v)$ as the maximal length of such a sequence.

Lemma 4.2.3. *There exists $n_0 \geq 0$ such that for any leaf x of t and for all $y \sqsubset x$, $s(x, y) \leq n_0$.*

Proof. Suppose that s is unbounded : for all n, there is x and $y \sqsubset x$ such that there is a decreasing sequence $\mathrm{seq}(x, y) = (x_0, x_1, \ldots, x_n, \ldots)$, possibly infinite. For each x_{i+1} in this sequence, there is a simple run of $\mathcal{A}$ from x_{i+1} to x_i containing (q_i, y), y being their

largest common prefix. If $n > 2|Q|$, there exist $i < j$ such that $q_{2i} = q_{2j}$. This means there are simple runs

$$(q_0, x_{2i+1}).u_i.(q_{2i}, y).v_i.(q'_i, x_{2i}) \quad \text{and}$$
$$(q_0, x_{2j+1}).u_j.(q_{2i}, y).v_j.(q'_j, x_{2j})$$

for some $u_i, v_i, u_j, v_j \in (Q \times \mathrm{Dom}(t))^*$. Then the sequence

$$(q_0, x_{2i+1}).u_i.(q_{2i}, y).v_j.(q'_j, x_{2j})$$

is also a run from x_{2i+1} to x_{2j}, so $x_{2i+1} < x_{2j}$. This is contradictory with the fact that the sequence is decreasing. □

To each node x, we associate the finite sequence $\vec{s}(x)$ of length $|x|$ of elements in $[0, n_0]$ defined by $s(x, x_0), \ldots, s(x, x_{|x|-1})$, where for all $k \in [0, |x| - 1]$, x_k denotes the prefix of x of length k.

For an illustration, consider the (uncolored) finite tree on left of Figure 4.1, where order on leaves is given along with the sequence $\vec{s}$. For instance, the leaf number 3 has maximal sequences $(3, 0), (3, 2, 1)$ and $(3, 1)$, so $\vec{s}(3) = (2, 3, 2)$.

The next lemma shows a key property : $\vec{s}$ is a morphism from $<_L$ to $<_{\mathrm{lex}}$.

Lemma 4.2.4. *For all x, y leaves of t, $x <_L y$ iff $\vec{s}(x) <_{\mathrm{lex}} \vec{s}(y)$ and $\vec{s}(x) \not\sqsubset \vec{s}(y)$.*

Proof. Let $x < Ly$ and let z be the largest prefix of x and y, with $x \neq z \neq y$ since x and y are leaves. For each $z' \sqsubset z$, there is decreasing sequence $\mathrm{seq}(x, z') = (x, x_1, \ldots, x_{s(x,z')})$, so there is a decreasing sequence $(y, x_1, \ldots, x_{s(x,z')})$ because the largest prefix of y and x_1 is z' and $x_1 <_L x <_L y$. Thus $s(x, z') \leq s(y, z')$.

For z, the sequence $\mathrm{seq}(x, z) = (x, x_1, \ldots, x_{s(x,z)})$ can be extended into a decreasing sequence $(y, x, x_1, \ldots, x_{s(x,z)})$, which means $s(x, z) < s(y, z)$ and $\vec{s}(x) <_{\mathrm{lex}} \vec{s}(y)$ with $\vec{s}(x) \not\sqsubset \vec{s}(y)$. The other direction is straightforward. □

Consider the tree $s(t)$ over the finite alphabet $[0, n_0]$ obtained by taking the prefix-closure of $\{\vec{s}(u) \mid u \text{ leaf of } t)\}$. The frontier of $s(t)$ is isomorphic to $(D, <_L)$. To ensure that colored frontier of $s(t)$ is isomorphic to L, we add the appropriate color to the leaves of $s(t)$. In practice, $\mathrm{Dom}(s(t)) := \{y \mid \exists x \in \mathrm{Leaves}(t) \wedge y \sqsubseteq \vec{s}(x)\}$ and $s(t)(x) = t(x)$ for a leaf x. A surprising result is that this tree can be built in the same class than the original tree.

Lemma 4.2.5. *If $t \in \mathrm{Tree}_n$, then $s(t) \in \mathrm{Tree}_n$.*

Proof. Let $G \in \mathrm{Graph}_{n-1}$ and $r \in V_G$ such that $\mathrm{Unf}(G, r) = t$ and G is accessible from r. There is a MSO-interpretation $\mathcal{I}$ replacing each arc labeled by $a \in \{0, 1\}$ by $n_0 + 1$ arcs labeled by $(i, a)_{0 \leq i \leq n_0}$. Let $t' = \mathrm{Unf}(\mathcal{I}(G), r)$. In particular, for any i, t' restricted to $((i, 0) + (i, 1))^*$ is isomorphic to t by the projection $\tau((i, a)) = a$.

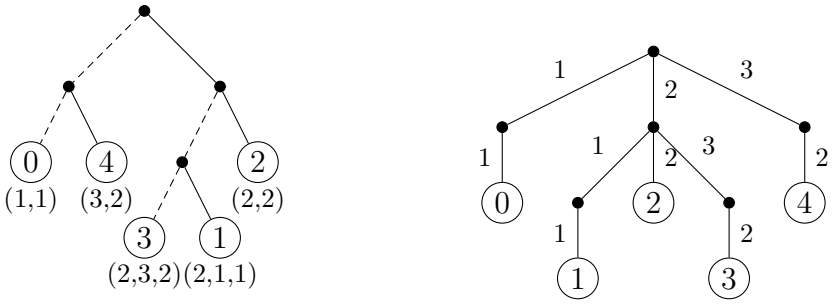


Figure 4.1: Order in a finite tree t and "arranged" tree $s(t)$.

There is a MSO-formula $\psi(x, z)$ satisfied by nodes x, z such that if $s(\tau(x), \tau(z)) = i$ then $z.(i, a) \sqsubseteq x$ for some $a \in \Sigma$ and $i < n_0$, where n_0 is defined in Lemma 4.2.3. Indeed, by Remark 4.1.3 the pairs of leaves of t accepted by $\mathcal{A}$ are recognizable by a MSO-formula. So the formula ψ finds two sets X_0, X_1 which elements have prefix respectively $z.(i, 0)$ and $z.(i, 1)$, and $x \in X_a$. The formula states that for each y (except one) in one set there is a y' in the other such that there is a simple run of $\mathcal{A}$ between y and y' up to projection by τ. Eventually ψ checks that $|X_0 \cup X_1| = i$, and that there is no such sets with $|X_0 \cup X_1| = i + 1$.

There is therefore a formula $\varphi(x)$ satisfied by nodes such that $\psi(x, z)$ is true for each prefix z of x. Let t'' be the tree restricted to the prefix closure of vertices satisfying $\psi(x)$: $t'' \in \mathrm{Tree}_n$.

By Lemma 4.2.4, if $\psi(x, z)$ and $\psi(y, z)$, then either x and y have the same prefix z, or $s(\tau(x), \tau(z)) \neq s(\tau(y), \tau(z))$; so if $z.(i, a) \in \mathrm{Dom}(t'')$, then $z.(i, 1 - a) \notin \mathrm{Dom}(t'')$. This means projecting arcs labels on their first component does not change the tree up to isomorphism. The resulting tree is $s(t)$. □

It remains to bind the lemmas and Proposition 4.2.1 to get the main result.

Theorem 4.2.6. *Any colored linear order of* Graph_n *greater than 1 is the frontier of a full prefix tree in* Tree_n.

Proof. By Proposition 4.1.2, for each linear order L in Graph_nthere is a det. binary tree $t \in \mathrm{Tree}_n$, an automaton $\mathcal{A}$ and a recoloring μ such that $\mu(\mathcal{A}(t)) = L$. By Proposition 4.2.2, $\mathcal{A}$ only works on leaves of t.

The Lemmas 4.2.4 and 4.2.5 prove that there is a tree $s(t)$ on a ranked alphabet such that $\mu(\mathbf{Fr}(s(t))) \simeq L$. By Proposition 2.5.4, the recoloring can be applied to $s(t)$ without going out of Tree_n. We can eventually apply Proposition 4.2.1 to get a full binary prefix tree. □

A particular case of this Theorem is the particular case of ω-words. We will use this result through Chapter 5.

Corollary 4.2.7. *An ω-word of* Graph_n *is the frontier of a full prefix tree in* Tree_n *with exactly one infinite branch* 1^ω.

4.3 Ordinals

In this section, we characterize the ordinals in the pushdown hierarchy. For more simplicity, orders of this section and of the following are uncolored, but the same applies to colored orders. In Section 3.1, ordinals below $\omega \uparrow\uparrow (n+1)$ were shown to be in Graph_n; here, we show that they are the only possible ordinals of Graph_n.

By Theorem 4.2.6, we only need to consider the frontier of prefix full binary trees. It is easy to see that frontier of a full binary prefix tree t is an ordinal if and only if t does not have an infinite branch with infinitely many 0's. We call such trees *well-ordered trees*.

Proposition 4.3.1. *A full prefix binary tree has an ordinal frontier if and only if it is a well-ordered tree.*

Proof. Let t be a (full binary prefix) well-ordered tree and suppose there is a infinite strictly decreasing sequence of leaves. Let t' be the restriction to prefixes of this sequence. Now t' is not full but its branches still have finitely many 0's. If t' had two distinct infinite branches, the infinitely many leaves supported by the largest would be larger than all the ones on the smallest, and they would fill the sequence. So t' has only one infinite branch, which has finitely many 0.

From the leaves of t', we can select a (decreasing) subsequence with increasing largest prefixes on the branch. Let uu' be an element of this subsequence with largest prefix on the branch u. For sufficiently large u, u has the maximum number of 0, so that the next element of the subsequence has the form $u1^kv$ for some v. Since t is deterministic, u' begins with 0, so $uu' <_{\mathrm{lex}} u1^kv$, and the sequence does not decrease.

Conversely, take a tree where one branch has infinitely many 0's. There is an infinite sequence $(u_i)_i$ such that for each i, u_i0 is in this branch; so u_i1 is not. Since the tree is prefix, there is a leaf v_i such that $u_i1 \sqsubset v_i$. Then the sequence $(v_i)_i$ is strictly decreasing. □

A non-full tree can yield an ordinal and still have a branch with infinitely many 0's, as Figure 4.2 shows. Nonetheless the Proposition 2.2.11 stands.

Let t is a well-ordered tree of Tree_n. To characterize t, we are going to look for an order α of the previous level of the hierarchy and show that t is at most ω^α. The natural first step is to consider the folded graph of t in the previous level. For $n \geq 1$, let $G \in \mathrm{Graph}_{n-1}$ and $r \in V_G$ such that $t = \mathrm{Unf}(G, r)$, and such that each vertex of V_G is accessible from r. Now we would like to find a well-order in G. Since Section 4.2, we know that it is equivalent to find a tree; so we build a well-ordered "spanning tree" into G, and we show that arcs which not in this tree cannot add too much complexity.

We start with this simple result.

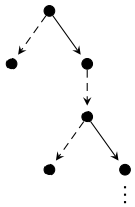


Figure 4.2: Non-full tree of frontier ω having a branch with infinitely many 0's.

Lemma 4.3.2. *For each $x \in V_G$, $\ell(x) = \min_{<_{\text{lex}}}\{w \mid r \xrightarrow[G]{w} x\}$ exists and $r \xrightarrow{\ell(x)} x$ is a simple path.*

Proof. It is a direct consequence of Proposition 2.2.11. Indeed, the set of paths from r to x is a subset of $\mathrm{Dom}(t)$ and has therefore a minimum element $\ell(x)$. If $r \xrightarrow[G]{\ell(x)} x$ is not a simple path, then there are $\ell(x) = u_1u_2u_3$ with nonempty u_2, u_3 and a vertex $y \in V_G$ such that $r \xrightarrow{u_1} y \xrightarrow{u_2} y \xrightarrow{u_3} x$. We may suppose u_1 is such that $|u_1|$ is maximal and $r \xrightarrow{u_1} y$ is a simple path. Since G is deterministic,

- either u_2 begins with 0 and u_3 with 1, but then $u_1u_2u_2u_3 <_{\text{lex}} \ell(x)$ is a path too;
- or the converse, but $u_1u_3 <_{\text{lex}} \ell(x)$.

So u_2 must be empty, and $r \xrightarrow{\ell(x)} x$ is a simple path. □

Let T be the subgraph of G defined by

$$T = \{x \xrightarrow{a} y \mid x \xrightarrow[G]{a} y \wedge \ell(y) = \ell(x) \cdot a\}.$$

T is the tree of shortest paths from r, and therefore a "spanning tree" as required.

Lemma 4.3.3. *T is a tree where each branch has finitely many 0, and $T \in \mathrm{Graph}_{n-1}$.*

Proof. For each $x \in V_G$, if $r \xrightarrow[T]{w_1} x$ and $r \xrightarrow[T]{w_2} x$, then $\min_{<_{\text{lex}}}\{w \mid r \xrightarrow{w} p\} \geq w_1$ and $\min_{<_{\text{lex}}}\{w \mid r \xrightarrow{w} x\} \geq w_2$, so $w_1 = w_2$. Since G is deterministic, so is T and thus T is a det. tree. Therefore T is a subtree of t and its branches have finitely many 0.

There is an interpretation $\mathcal{I} = \{\varphi_0, \varphi_1\}$ building $\mathcal{I}(G) = T$. Indeed, it is possible to define subsets containing the smallest path from r to a given x, and to select the smallest such subset by inclusion. Then arcs are selected to build a path. □

Note that T is not a well-ordered tree, because it is not necessarily a prefix tree. This property will be fixed later on by "completing" the tree.

Since T is a tree, we may take its tree presentation, so that its vertex set is the set of paths from the root, and thus $r = \varepsilon$. This is also the set of vertices of G.

The following lemma states a technical property on arcs of G which are not in T.

Lemma 4.3.4. *For each $x \xrightarrow{a} y \in G \setminus T$,*

- *if $x <_{\text{lex}} y$ then $x \sqsubset y$;*
- *if $a = 0$, then $y <_{\text{lex}} x$ and $x \perp y$.*

Proof. For $a \in \{0, 1\}$, if $x \xrightarrow{a} q \in G \setminus T$, $x <_{\text{lex}} y$ and $x \perp y$, then the path $r \xrightarrow{xa} y$ is smaller than $r \xrightarrow{y} y$, which is contradictory with the definition of T.

Let $x \xrightarrow{0} y \in G \setminus T$. If $y \sqsubseteq x$, then there is a loop in G containing a 0, and thus a branch with infinitely many 0 in t. If $x \sqsubset y$, then $x1 \sqsubseteq y$ and therefore the path $r \xrightarrow{x0} y$ is smaller than the path $r \xrightarrow{y} y$. In conclusion, $y <_{\text{lex}} x$ and $q \perp p$. □

As hinted above, we now tranform T to get a prefix tree. Let $\bar{T}$ be the tree such that T is a subtree of $\bar{T}$ and whenever $x \xrightarrow[G]{a} y$, then $xa \in \bar{T}$. It easy to see that $\bar{T}$ can be transducted from G.

Lemma 4.3.5. *$\bar{T}$ is a well-ordered tree of* Graph_n.

Proof. Since Comp only add leaves to T, each infinite branch of $\bar{T}$ is also in T so it may have finitely many 0. Since G is a full binary graph (each vertex has arity 0 or 2), then each vertex of $\bar{T}$ has also arity 0 or 2. The complete binary tree is not a subtree of $\bar{T}$, so each subtree of $\bar{T}$ has at least one leaf; $\bar{T}$ is prefix and therefore $\bar{T}$ is a well-ordered tree. Since Graph_n is closed by transduction, $\bar{T} \in \text{Graph}_n$. □

As a consequence, by Proposition 4.3.1, $\mathbf{Fr}(\bar{T})$ is an ordinal.

Example 4.3.6. Figure 4.3 is a example with a finite graph G and $\mathbf{Fr}(\bar{T}) = 6$. For each leaf w of $\text{Unf}(G, r)$, $\sigma(w) \in (\varepsilon + 4^*3 + 5)2^*(0 + 1)$. So we have

$$\mathbf{Fr}(\text{Unf}(G, r)) = \omega + \omega^2 + \omega = \omega^2 + \omega \leq \omega^6. \qquad \blacktriangle$$

The previous example leads to the central lemma of this section, which is the recursion mechanism.

Lemma 4.3.7. $\mathbf{Fr}(t) \leq \omega^{\mathbf{Fr}(\bar{T})}$.

Proof. Let $\alpha = \mathbf{Fr}(\bar{T})$. Recall that $V_G = \text{Dom}(T) \subseteq \text{Dom}(\bar{T})$, i.e. $r = \varepsilon$ and $r \xrightarrow[G]{w} w$. There is a natural isomorphism τ from leaves of $\bar{T}$ to α. As described in Corollary 2.2.2 and used throughout in Chapter 3, we are going to build a injective morphism from the leaves of t to the set of decreasing sequences of ordinals smaller than α in lexicographic order.

Let w be a leaf of t. There is a path $r \xrightarrow{w} p$ in G where p is a leaf of G. Let $(p_i)_{0 \leq i \leq |w|}$ be the nodes of this path, *i.e.* $r \xrightarrow[G]{w_i} p_i$ where $w_i \sqsubseteq w$, $|w_i| = i$ for $0 \leq i \leq |w|$. Let a_i

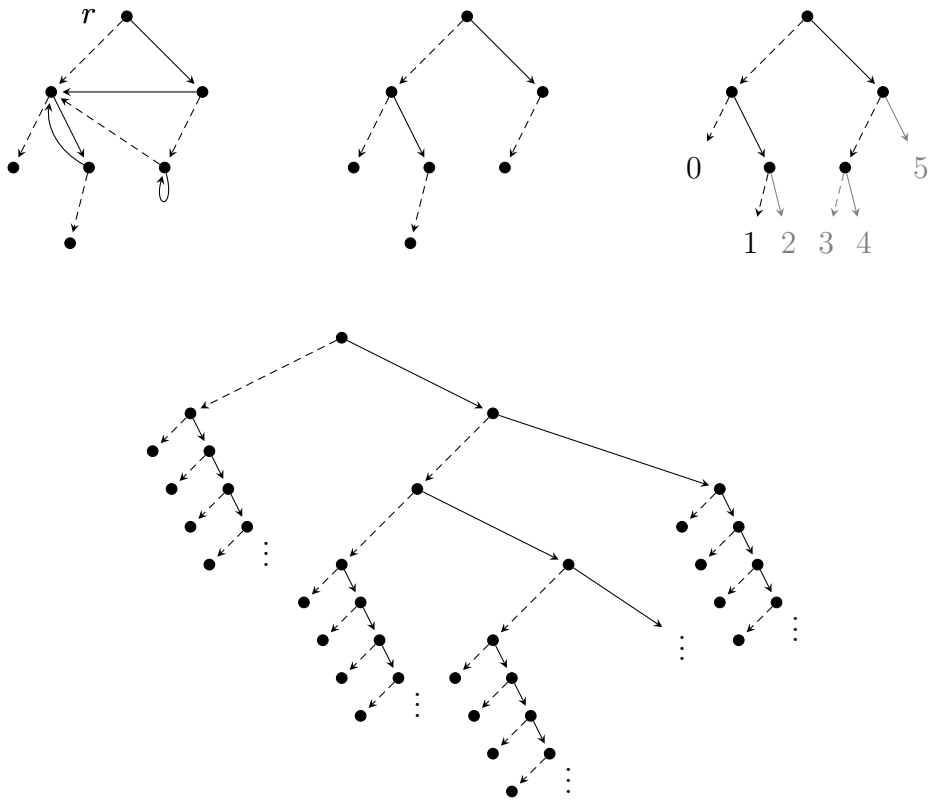


Figure 4.3: A finite graph G, "spanning tree" T, completed tree $\bar{T}$, and unfolding.

---

be the letter such that $w_{i+1} = w_i.a_i$ and $a_{|w|} = \varepsilon$. We build the following finite sequence $\sigma(w)$ associated to w :

$$\begin{array}{rcll} \sigma(w_{-1}) & = & \varepsilon \text{ (by convention)} & \\ \text{for } 0 \leq i < |w| - 1, & & & \\ \sigma(w_i) & = & \sigma(w_{i-1}) & \text{if } p_{i+1} = p_i.a_i, \text{ i.e. } p_i \xrightarrow{a_i} p_{i+1} \in T \\ \sigma(w_i) & = & \sigma(w_{i-1}), \tau(p_i.a_i) & \text{if } p_i \xrightarrow{a_i} p_{i+1} \notin T \\ \sigma(w) & = & \sigma(w_{|w|-1}), \tau(p_{|w|}) & \end{array}$$

We prove that $\sigma(w)$ is decreasing. Suppose $\sigma(w)$ has at least two elements. Let $i < j$ be the indexes of two consecutive elements, i.e. $\sigma(w_j) = \sigma(w_{i-1}) \cdot \tau(p_i.a_i) \cdot \tau(p_j.a_j)$. Then $p_i \xrightarrow{a_i} p_{i+1} \in G \setminus T$. By Lemma 4.3.4, either $p_i \geq_{\text{lex}} p_{i+1}$ or ($p_i <_{\text{lex}} p_{i+1}$ and $p_i \sqsubset p_{i+1}$). By construction, $p_{i+1} \sqsubseteq p_j$ so $p_j.a_j$ is in the subtree of $\bar{T}$ of root p_{i+1}. Several cases arise.

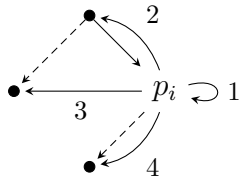
1. Either $p_i = p_{i+1}$. Then $a_i = 1$ and $p_i.1$ is the largest leaf of the the subtree of $\bar{T}$ of root p_{i+1}, so $p_i.1 \geq_{\text{lex}} p_j.a_j$.

2. Or $p_{i+1} \sqsubset p_i$. Then again $a_i = 1$, and $p_i = p_{i+1}1^k$ for some k, because a cycle in G cannot contain any 0. So $p_i.1$ is the largest of the subtree of root p_{i+1} and $p_i.1 \geq_{\text{lex}} p_j.a_j$.

3. Or $p_i >_{\text{lex}} p_{i+1}$ and $p_{i+1} \perp p_i$, then $p_i >_{\text{lex}} p_j$ and $p_i.a_i >_{\text{lex}} p_j.a_j$.

4. Or $p_i \sqsubset p_{i+1}$. Then $a_i = 1$. Since G is deterministic, $p_i.0 \sqsubset p_{i+1} \sqsubset p_j$. So $p_i.1 >_{\text{lex}} p_j.a_j$.

These cases are summed up in the following figure. Arcs 1, 2 and 4 are necessarily labeled by 1, whereas arc 3 may be labeled by 0 or 1.



To sum up, two successive elements $\tau(p_i.a_i), \tau(p_j.a_j)$ of $\sigma(w)$ are such that $p_i.1 \geq_{\text{lex}} p_j.a_j$ and so $\tau(p_i.a_i) > \tau(p_j.a_j)$. The sequence $\sigma(w)$ is decreasing.

Let $w < v$ be two leaves of t of largest common prefix u. Let $i < |u|$ be the largest such that $(p_i.a_i)$ is in $\sigma(w)$: then $\sigma(w_i) = \sigma(v_i)$. Let $j_w, j_v \geq |u|$ be the smallest such that $(p_{j_w}.a_{j_w}) \in \sigma(w)$ (resp. $(p_{j_v}.a_{j_v}) \in \sigma(v)$). Then $p_{j_w}.a_{j_w} = p_{i+1}.u_w$ with $u.u_w \sqsubseteq w.0$ (resp. $p_{j_v}.a_{j_v} = p_{i+1}.u_v$ with $u.u_v \sqsubseteq v.0$) and $u_w < u_v$, so $p_{j_w}.a_{j_w} <_{\text{lex}} p_{j_v}.a_{j_v}$. This shows that σ is a injective morphism.

We have thus proved that σ is an injective morphism from leaves of t to decreasing sequences of ordinals of α by lexicographic order. So $\mathbf{Fr}(t) \leq \omega^{\mathbf{Fr}(\bar{T})}$. □

From here, the main theorem is obtained by induction, since finite ordinals are in Graph_0.

Theorem 4.3.8. *For $n \geq 1$ and any ordinal α, $\alpha \in \text{Graph}_n$ if and only if $\alpha < \omega \uparrow\uparrow (n+1)$.*

This results give an alternative proof of the strictness of the pushdown hierarchy and shows that ε_0 does not belong to this hierarchy.

Another direct corollary concerns the interpretation of an ordinal into another. For instance, for any $\alpha < \omega^\omega$, it is possible to find an interpretation $\mathcal{I}$ such that $\alpha = \mathcal{I}(\omega)$. As a corollary of the previous theorem, we cannot interpret more[1]. At any level, if $\beta < \omega \uparrow\uparrow n \leq \alpha$ then α cannot be interpreted in β. Interestingly, the same property applies to covering graphs. It could be interesting to examine precisely the orbit of a given ordinal under monadic interpretations. Theorem 3.4.2 hints that this orbit must actually be very small, *i.e.* if α has ω-head β, then the orbit would be of the form $\omega^\omega \cup \{\beta + \delta \mid \delta < \omega^\omega\}$; for covering graphs, the orbit is larger.

[1]This result was already obtained in [BNR+10] in an stronger form : it remains true even when ω is colored with arbitrary predicates.

4.4 Scattered linear orders

In Section 3.2, we have proved that any power by $\alpha < \omega \uparrow\uparrow n$ of ζ is actually in Graph_n. In the other way around, we consider the scattered linear orders in the pushdown hierarchy. Using the result of the previous section, we characterize the Hausdorff rank of scattered orderings of the hierarchy : this is Theorem 4.4.12.

To reach this result, we begin with a general proposition on trees have a scattered frontier. Then we notice that we may recursively switch subtrees of such a tree to obtain an ordinal frontier. This does not change the mesure based on the number of infinite branches in the tree, called Cantor-Bendixson rank of the tree. It is then sufficient to prove that this CB-rank is tightly related to the Hausdorff rank, which allows us to conclude.

4.4.1 Trees with scattered frontiers

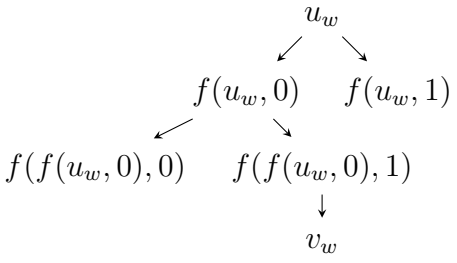
The countable scattered orders are those which are frontiers of trees with only countably many infinite branches also called *tame trees*. The following proposition is part of the folklore.

Proposition 4.4.1. *Let t be a deterministic prefix binary tree, the following propositions are equivalent:*

1. $\mathbf{Fr}(t)$ *is a scattered linear order,*
2. t *has countably many infinite branches,*
3. t *does not contain any branching subset (i.e. a non-empty subset $U \subseteq \text{Dom}(t)$ such that for all $u \in U$, $u0\{0,1\}^* \cap U \neq \emptyset$ and $u1\{0,1\}^* \cap U \neq \emptyset$).*

Proof. We will show that $1 \Rightarrow 3 \Rightarrow 2 \Rightarrow 1$.
$\underline{1 \Rightarrow 3.}$ We prove this implication by contraposition. Assume that t contains a branching subset U. There exists a mapping $f : U \times \{0,1\} \mapsto U$ s.t. for all $u \in U$ and $i \in \{0,1\}$, $f(u,i)$ belongs to $u \cdot i \cdot \{0,1\}^*$. Furthermore, as t is prefix, for each u there is a leaf v with $u \sqsubset v$.

For any $w \in \{0,1\}^*$, we define the nodes u_w and v_w. Let u_ε be any node of U. For any u_w, we set $u_{w0} = f(f(u,0),0)$, $u_{w1} = f(u,1)$ and v_w a leaf such that $f(f(u,0),1) \sqsubset v_w$. Then $v_w <_{\text{lex}} v_{w'} \iff (w \perp w' \wedge w <_{\text{lex}} w') \vee (w' \sqsubseteq w0) \vee (w1 \sqsubseteq w')$. This order is dense on $\{0,1\}^*$, so $(v_w)_{\{w \in \{0,1\}^*\}}$ is a dense suborder of $\mathbf{Fr}(t)$.



$\underline{2 \Rightarrow 1}$ In the proof of Thm. 7.7 of [KRS05], it is shown that if t has countably many branches then the Kleene-Brouwer ordering $\mathrm{KB}(t)$ is scattered (see Section 2.2.3). As $\mathbf{Fr}(t) \preceq \mathrm{KB}(t)$, we can conclude.
$\underline{3 \Rightarrow 2}$ Assume that t has uncountably many infinite branches. Consider the set of U of nodes of t such that there are uncountably many infinite branches going through $x0$ and through $x1$. We prove that if there are no point of U below some node u of t then there are only countably many infinite branches going through u. Indeed, suppose that u bears infinitely many branches, we could construct a sequence of consecutive nodes $(u_i)_{i\in\mathbb{N}}$ of t starting with u such that for all $i \in \mathbb{N}$, u_{i+1} is a son of u_i and its other son v_{i+1} (if it exists) has only countably many infinite branches going through it. The set of branches going through u is equal to the following countable union:

- the unique branch going through the u_i's,
- the set of infinite branches going through the $v_i's$ for $i \geq 1$.

A countable union of countable set is again countable. So if $x \in U$, then $x0(0+1)^* \cap U \neq \emptyset$ and $x1(0+1)^* \cap U \neq \emptyset$, and U is a branching subset. □

Remark 4.4.2. The direction $2 \Rightarrow 3$ is also easily shown by contraposition. Assume that $\mathrm{Dom}(t)$ contains a branching subset U. As U is branching, there exists a mapping $f : U \times \{0,1\} \mapsto U$ s.t. for all $u \in U$ and $i \in \{0,1\}$, $f(u,i)$ belongs to $ui\{0,1\}^*$. Let x_0 be an arbitrary element of x_0. To any infinite sequence $\delta = (\delta_i)_{i\in\mathbb{N}} \in \{0,1\}^\omega$, we associate the unique infinite branch of t going through the x_i where for all $i \geq 0$, $x_{i+1} = f(x_i, \delta_i)$. This defines an injection from $\{0,1\}^\omega$ into the set of infinite branches of t. Hence t has uncountably many infinite branches. ■

4.4.2 Permutation of subtrees

We introduce here the notion of permutation of subtrees in a tree. Intuitively, we have already seen the notion of unlabeled trees, *i.e.* the class of trees which have the same image under label projection (up to isomorphism, as usual). Given a deterministic binary tree t, what other deterministic binary trees are in the same class? And what of their frontiers?

Definition. Given two deterministic trees t and t', we write $t \equiv t'$ if there exists a bijection from $\mathrm{Dom}(t)$ to $\mathrm{Dom}(t')$ preserving the ancestor relation (i.e. for all $u, v \in \mathrm{Dom}(t)$, $u \sqsubseteq v$ iff $h(u) \sqsubseteq h(v)$).

This section is devoted to the following property of this relation.

Proposition 4.4.3. *For any prefix tame full binary tree t, there exists a well-ordered tree t' such $t \equiv t'$. Furthermore, if t belongs to* Tree_n *than t' can also be chosen in* Tree_n.

Consider the following game $\mathcal{G}$ played by two players Branch and Spoiler by moving a token on t. The two players play in turn starting with Branch. Branch moves the token to a node anywhere below the current position. Spoiler can only move the token to a son of the current position. Branch loses the game if the token reaches a leaf.

The game can easily be translated in an equivalent parity game. For a formal definition of parity games and their properties, we refer the reader to [Tho97b]. It follows from the positional determinacy of parity games [Zie98] that either Branch and Spoiler as a positional winning strategy for $\mathcal{G}$.

Lemma 4.4.4. *Branch wins the game if and only if t contains a branching subset.*

Proof. [$\Rightarrow$] Assume that Branch has a positional winning strategy for $\mathcal{G}$ which is a partial function $\Phi : \mathrm{Dom}(t) \mapsto \mathrm{Dom}(t)$. Consider the smallest set $U \subseteq \mathrm{Dom}(t)$ such that $\Phi(\varepsilon) \in U$ and for all $u \in U$, $\Phi(u0)$ and $\Phi(u1)$ also belong to U. It is easy to check that U is branching.
[$\Leftarrow$] If t contains a branching subset U, consider any partial function $\Phi : \mathrm{Dom}(t) \mapsto \mathrm{Dom}(t)$ such that $\Phi(\varepsilon) \in U$ and for all $u \in U$, $\Phi(u0) \in u0\{0,1\}^* \cap U$ and $\Phi(u1) \in u1\{0,1\}^* \cap U$. We check that Φ is a positional winning strategy for Branch on $\mathcal{G}$. □

As t is tame (and hence does not contain any branching subset), *Branch* loses the game. Hence Spoiler has a positional winning strategy. There exists a mapping $\varphi : \mathrm{Dom}(t) \mapsto \{0,1\}$ such that in any game where Spoiler choses his moves according to φ (i.e. at node u, Spoiler pick the $\varphi(u)$-son) is won by him.

Consider the tree t' obtained from t by swapping the two sons of any node u such that $\varphi(u) = 1$. Formally consider the mapping h from $\mathrm{Dom}(t)$ to $\{0,1\}$ defined for all $ui \in \mathrm{Dom}(t)$ with $u \in \mathrm{Dom}(t)$ and $i \in \{0,1\}$ by :

$$\begin{array}{lcll} h(\varepsilon) & = & \varepsilon & \\ h(ui) & = & h(u) \cdot (1-i) & \text{if } \Phi(u) = 1, \\ h(ui) & = & h(u) \cdot i & \text{if } \Phi(u) = 0. \end{array}$$

It is easy to check that $h(\mathrm{Dom}(t))$ is prefix closed and that for all $u, v \in \mathrm{Dom}(t)$, $u \sqsubseteq v$ iff $h(u) \sqsubseteq h(v)$. Let t' be the tree such that $\mathrm{Dom}(t')$ is equal to $h(\mathrm{Dom}(t))$. As h is injective, the mapping h a bijection from $\mathrm{Dom}(t)$ to $\mathrm{Dom}(t')$ preserving the ancestor relation. Hence $t \equiv t'$.

Lemma 4.4.5. *The tree t' is a well-ordered tree.*

Proof. Assume by contradiction that there exists an infinite branch B in t' containing infinitely many 0's. We are going to construct an infinite play π for $\mathcal{G}$ where Spoiler plays according to Φ. As the play is infinite, Spoiler loses this play which contradicts the fact that Φ is a winning strategy for him.

Let $u_0, \ldots, u_n, \ldots$ be the consecutive nodes of B. In particular $u_0 = \varepsilon$ and for all $i \geq 0$, there exists $k_i \in \{0,1\}$ such that $u_{i+1} = u_i k_i$. Let B' be the corresponding infinite branch in t (i.e. $B' = h^{-1}(B)$) and let $v_0, \ldots, v_n, \ldots$ be the consecutive nodes of B'.

In the play π, Branch plays so as to stay on B'. Initially this is satisfied as $v_0 = \varepsilon$. If the token is at some v_i and Branch has to play, Branch moves to some v_j with $j > i$ such that $k_j = 0$. Such a j always exists as by assumption there infinitely many k_j equal to 0. By definition of h, we know that $\Phi(v_j) = v_{j+1}$. As Spoiler plays according to Φ he moves to v_{j+1}. □

It remains to show that t' can be chosen in Tree_n if t is in Tree_n. A positional winning strategy Φ for Spoiler can be coded by two sets of vertices U_0 and U_1 respectively corresponding to set of nodes u s.t. $\Phi(u) = 0$ and $\Phi(u) = 1$. Consider an MSO-formula $\varphi(X_0, X_1)$ such that $t \models \varphi[U_0, U_1]$ if and only if U_0 and U_1 encode a positional winning strategy for Φ. The formula φ simply states that there are no infinite branch $B = b_1 \ldots b_n \cdots \in \{0,1\}^\omega$ such that for infinitely many $k \geq 0$, $b_1 \ldots b_k \in U_{b_{k+1}}$. As Branch loses $\mathcal{G}$, $t \models \exists X_0, X_1, \varphi[X_0, X_1]$.

From Prop. 2.5.7, it follows that there exists a tree $\bar{t} \in \mathrm{Tree}_n$ colored with two sets U_0 and U_1 coding a positional winning strategy for Spoiler. Let c_0 and c_1 the colors corresponding respectively to U_0 and U_1. The swapping to obtain t' can be obtained by applying an MSO-interpretation to the graph in Graph_{n-1} whose unfolding is $\bar{t}$. More formally, let G be a graph in Graph_n and let $r \in V_G$ s.t. $\bar{t} = \mathrm{Unf}(G, r)$. Consider the MSO-interpretation $\mathcal{I}$ which relabels the arcs labeled by 0 (resp. 1) by 1 (resp. 0) if their source is colored by c_1, erases the colors c_0 and c_1 and otherwise leave the graph untouched. It is easy to check that $t' \simeq Unf(\mathcal{I}(G), r)$ which concludes the proof.

4.4.3 Cantor-Bendixson rank of deterministic trees

The Cantor-Bendixson rank of a tree is an ordinal assessing *the branching complexity* of a tree. We use a definition taken from [KRS05].

Definition. For $X \subseteq \mathrm{Dom}(t)$, we write $\partial(X)$ the set of nodes $x \in X$ with at least two infinite branches from x in X. It is easy to see that if X is prefix closed then so is $\partial(X)$. Hence the operation can be iterated as follows :

$$\begin{aligned} \partial^0(X) &= X \\ \partial^{\alpha+1}(X) &= \partial(\partial^\alpha(X)) \\ \partial^\lambda(X) &= \bigcap_{\alpha<\lambda} \partial^\alpha(X) \text{ for limit } \lambda. \end{aligned}$$

The Cantor-Bendixson rank (CB-rank) of t, noted $r_{\mathrm{CB}}(t)$, is the least ordinal α such that $\partial^\alpha(\mathrm{Dom}(t)) = \partial^{\alpha+1}(\mathrm{Dom}(t))$.

Remark 4.4.6. This is an adaptation [Kec94, Exercice 6.17] of the standard notion of Cantor-Bendixson rank of an arbitrary topological space X. The Cantor-Bendixson derivative DX is the set of accumulation points, *i.e.* $DX = \{x \in X \mid x \in \overline{X \setminus \{x\}}\}$. As [KRS05, Rem. 7.2] points out, it is equivalent to the above definition if we take X as the set of infinite paths in the tree. ■

From [KRS05], t is tame if and only if there exists α s.t. $\partial^\alpha = \emptyset$. On the opposite, a tree with uncountably many branches is such that any node of $\partial^{r_{\mathrm{CB}}(t)}(t)$ belongs to a branching subset.

For tame trees t, we adopt a slightly modified version of the CB-rank, written $\tilde{r}_{\mathrm{CB}}(t)$, which is the smallest ordinal α such that $\partial^\alpha(t)$ is finite. The difference between $\tilde{r}_{\mathrm{CB}}$ and r_{CB} is at most one as $\tilde{r}_{\mathrm{CB}}(t) \leq r_{\mathrm{CB}}(t) \leq \tilde{r}_{\mathrm{CB}}(t) + 1$.

The CB-rank of tame trees and the Hausdorff rank of their frontier are tightly linked.

Proposition 4.4.7. *For every prefix and tame binary tree t,*

$$\begin{array}{rcll} \tilde{r}_{\mathrm{CB}}(t) & = & \tilde{r}(\mathbf{Fr}(t)) & \text{if } \tilde{r}_{\mathrm{CB}}(t) < \omega, \\ \tilde{r}_{\mathrm{CB}}(t) & = & \tilde{r}(\mathbf{Fr}(t)) + 1 & \text{otherwise.} \end{array}$$

Proof. Let t be a prefix tame binary tree t. As the construction of Fact 4.2.1 does not change neither the CB-rank nor the frontier of the tree, we can assume w.l.o.g. that t is also full.

For every prefix tame full binary tree t, we write $\mathrm{WF}(t)$ the smallest ordinal α such that there exists a well-ordered tree t' with $t \equiv t'$ and $\mathbf{Fr}(t') = \alpha$. Proposition 4.4.3 garantees the existence of at least one such t'.

We are going to show by transfinite induction on $\mathrm{WF}(t)$ that for every a prefix tame full binary tree t that $\tilde{r}_{\mathrm{CB}}(t) = \tilde{r}(\mathbf{Fr}(t)) + \delta$ with $\delta = 0$ if $\tilde{r}_{\mathrm{CB}}(t) < \omega$ and $\delta = 1$ otherwise. This is obvious for finite trees.

Assume that the property holds for all $\alpha < \beta$ for some β. Let t be a prefix tame full binary tree with $\mathrm{WF}(t) = \beta$. Let t' be a full well-ordered tree such that $\mathbf{Fr}(t) = \beta$ and $t \equiv t'$.

We distinguish two cases depending on whether 1^ω is an infinite branch in t'.

1^ω is not an infinite branch in t'. Let $m \geq 0$ be the maximal integer such that 1^m belongs to t'. For all $i \in [0, m-1]$, let t'_i be the subtree of t' rooted at 1^i0 et let t'_m be the tree root in 1^m consisting on only one node. As t' is full, $\mathbf{Fr}(t') = \sum_{[0,m-1]} \mathbf{Fr}(t'_1) + 1$. In particular for all $i \in [0, m-1]$, $\mathbf{Fr}(t'_i) < \mathbf{Fr}(t') = \beta$.

By definition of $\equiv$, there exists a word $w = w_1 \dots w_m \in \{0,1\}^m$ which the branch in t corresponding to to the branch 1^m in t'. Formally, for all $i \in [0, m]$, a tree t_i such that $t_i \equiv t'_i$ and t is the prefix closure of $\bigcup_{i \in [0,m-1]} w_1 \cdots w_i \cdot (1 - w_{i+1}) \cdot t_i$. Hence $\tilde{r}(\mathbf{Fr}(t)) = \max\{\tilde{r}(\mathbf{Fr}(t_i)) \mid i \in [0, m]\}$ by Proposition 2.2.8 and $\tilde{r}_{\mathrm{CB}}(t) = \max\{\tilde{r}_{\mathrm{CB}}(t_i) \mid i \in [0, m]\}$ because the subtree permutation commutes with $\tilde{r}_{\mathrm{CB}}$.

For all $i \in [0, m]$, $\mathrm{WF}(t_i) < \beta$, but by Proposition 2.2.8 again, there is an $i_{\max}$ such

that $\tilde{r}(t_{i_{\max}}) = \tilde{r}(\beta)$. So we have by induction hypothesis that $\tilde{r}_{\mathrm{CB}}(t_{i_{\max}}) = \tilde{r}(\mathbf{Fr}(t_{i_{\max}}))+\delta$. This shows that $\tilde{r}_{\mathrm{CB}}(t) = \tilde{r}(\mathbf{Fr}(t)) + \delta$.

1^ω is an infinite branch in t'. For all $i \geq 0$, let t'_i be the subtree of t' rooted at 1^i0. The frontier of t' is equal to $\sum_{i\geq 0} \mathbf{Fr}(t'_i)$. As t' is full, $\mathbf{Fr}(t'_i) \neq \mathbf{0}$ for all $i \geq 0$. Hence for all $i \geq 0$, $\mathbf{Fr}(t'_i) < \mathbf{Fr}(t') = \beta$.

By definition of $\equiv$, there exists an infinite branch $w = w_1 \dots w_m \dots \in \{0,1\}^\omega$ and for all $i \geq 0$, a tree t_i such that $t_i \equiv t'_i$ and $t = \bigcup_{i\geq 0} w_1 \cdots w_i \cdot (1 - w_{i+1}) \cdot t_i$. Hence $\tilde{r}(\mathbf{Fr}(t)) = \sum_{i\in W} \mathbf{Fr}(t_i)$ where W designates the order $\mathbb{N}$ ordered by

$$i \prec j \text{ iff } w_1 \dots w_i.(1 - w_i) <_{\mathrm{lex}} w_1 \dots w_j.(1 - w_j).$$

In particular, W has order type

$$\begin{array}{rcl} \omega + \mathbf{k} \text{ for some } k < \omega & \text{iff} & w \text{ is ultimately } 1^\omega \\ \mathbf{k} + \omega^* \text{ for some } k < \omega & \text{iff} & w \text{ is ultimately } 0^\omega \\ \omega + \omega^* & & \text{otherwise.} \end{array}$$

Assume that the type of W is $\omega + \omega^*$; the proof is only simpler for other types. By Remark 2.2.10, $I = \{i \mid \tilde{r}(\mathbf{Fr}(t_i)) = \beta\}$ is finite.

- If $\beta = \alpha + 1$, note that $\beta < \omega \iff \alpha < \omega$. By induction, for $j \notin I$, $\partial^{\alpha+\delta}(t_j)$ is finite. If $I \neq \emptyset$, $\partial^{\beta+\delta}(t)$ is reduced to the tree bearing the $\partial^{\beta+\delta}(t_i)$ for $i \in I$, which are finite. If I is empty, $\partial^{\alpha+\delta}(t)$ has exactly one infinite branch. So $\tilde{r}_{\mathrm{CB}}(t) = \beta + \delta$.
- Otherwise, β is limit and $\beta \geq \omega$. If I is empty, for $\alpha < \beta$ there always infinitely many $j \notin I$ such that $\alpha \leq \tilde{r}_{\mathrm{CB}}(t_j) < \beta$. So $\partial^\alpha(t)$ always contains the infinite branch w. Otherwise, for $i \in I$, by induction $\partial^\beta(t_i)$ is infinite. So $\partial^\beta(t)$ has always at least one infinite branch. For all $i \in I$, $\partial^{\beta+1}(t_i)$ is finite and so is $\partial^{\beta+1}(t)$. □

As an unrelated note, we may use the same proof technique to prove the equivalence between different orders on a prefix tame tree.

Proposition 4.4.8. *For every full binary prefix tame tree t,*

$$\tilde{r}(\mathbf{Fr}(t)) = \tilde{r}(\mathrm{KB}(t)) = \tilde{r}(\langle \mathrm{Dom}(t), <_{\mathrm{lex}} \rangle).$$

Proof. We note $\mathrm{Lex}(t) = \langle \mathrm{Dom}(t), <_{\mathrm{lex}} \rangle$. Let again $\mathrm{WF}(t)$ be the smallest ordinal α such that there exists a well-ordered tree t' with $t \equiv t'$ and $\mathbf{Fr}(t') = \alpha$. We prove the result by induction on $\mathrm{WF}(t)$. If $\mathrm{WF}(t) = 1$, since the tree is prefix, it is reduced to a single node and then $\mathbf{Fr}(t) = \mathrm{KB}(t) = \mathrm{Lex}(t) = \mathbf{1}$.

In the general case, it depends again whether t' has an infinite branch 1^ω or not. The t_i and t'_i trees are defined in the same way that in the previous proof; since $t'_i \equiv t_i$ and $\mathbf{Fr}(t'_i) < \mathbf{Fr}(t)$, we may apply the induction hypothesis

$$\tilde{r}(\mathbf{Fr}(t_i)) = \tilde{r}(\mathrm{KB}(t_i)) = \tilde{r}(\langle \mathrm{Dom}(t_i), <_{\mathrm{lex}} \rangle).$$

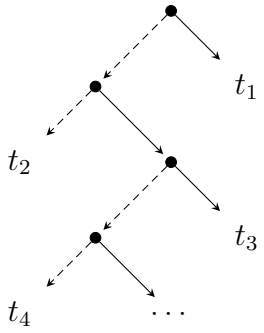


Figure 4.4: Orders in a tree.

$$\begin{array}{rcl}
\mathbf{Fr}(t) & = & \mathbf{Fr}(t_2) + \mathbf{Fr}(t_4) + \cdots + \mathbf{Fr}(t_3) + \mathbf{Fr}(t_1), \\
\mathrm{KB}(t) & = & \mathrm{KB}(t_2) + \mathrm{KB}(t_4) + \cdots + 1 + \mathrm{KB}(t_3) + \mathbf{2} + \mathrm{KB}(t_1) + \mathbf{1}, \\
\mathrm{Lex}(t) & = & \mathbf{2} + \mathrm{Lex}(t_2) + \mathbf{2} + \mathrm{Lex}(t_4) + \cdots + \mathrm{Lex}(t_3) + \mathrm{Lex}(t_1).
\end{array}$$

When 1^ω is not an infinite branch in t', it is the successor case. Then $\mathbf{Fr}(t) = \sum_{i=1}^{k} \mathbf{Fr}(t_i)$, whereas $\mathrm{KB}(t)$ and $\mathrm{Lex}(t)$ have the same form

$$\begin{array}{rcl}
\mathrm{KB}(t) & = & \sum_{i=1}^{k} (\mathrm{KB}(t_i) + c_i) \\
\mathrm{Lex}(t) & = & \sum_{i=1}^{k} (c'_i + \mathrm{Lex}(t_i))
\end{array}$$

where c_i, c'_i are integers. We may then apply Proposition 2.2.8 as before to get the result.

The limit case is when 1^ω is an infinite branch in t'. Now $\mathbf{Fr}(t) = \sum_{i \in W} \mathbf{Fr}(t_i)$ where W has order type $\omega + \omega^*$, $\omega + k$ or $k + \omega^*$. Then we have the same

$$\begin{array}{rcl}
\mathrm{KB}(t) & = & \sum_{i \in W} (\mathrm{KB}(t_i) + c_i) \\
\mathrm{Lex}(t) & = & \sum_{i \in W} (c_i + \mathrm{Lex}(t_i))
\end{array}$$

as illustrated in Figure 4.4. The Proposition 2.2.8 along with Remark 2.2.10 leads to the main result. □

Obviously, the property fails for the actual Hausdorff rank, since a full prefix tree of frontier ω has a KB-order $\omega + \omega^*$, which has Hausdorff rank 2 but $\sim$-rank 1.

It would be interesting to capture all the orders on a tame tree which share the $\sim$-rank. It may be conjectured that all orders MSO-definable on $\mathrm{Dom}(t)$ have the same $\sim$-rank; however this does not include $\mathbf{Fr}(t)$, which concerns only leaves. This raises the more general conjecture.

Conjecture 4.4.9. *Let t be a tame tree. All orders definable by a monadic formula such that any vertex is prefix of a vertex selected by the formula have the same $\sim$-rank.*

Remark 4.4.10. The proof of Theorem 7.7 in [KRS05] states that $\tilde{r}_{\mathrm{CB}}(t) = \tilde{r}(\mathrm{KB}(t))$, which seems to contradict Proposition 4.4.7. In fact this is true for finite $\tilde{r}_{\mathrm{CB}}$, which is

enough to prove the said Theorem, but false for further ordinals. Consider the following counterexample. We define the family $(t_\alpha)_{\alpha\leq\omega}$ of prefix trees by the following set of leaves.

$$\begin{aligned}
\text{Leaves}(t_1) &= 1^*0\\
\text{Leaves}(t_2) &= 1^*0\cdot\text{Leaves}(t_1)\\
&\dots\\
\text{Leaves}(t_i) &= 1^*0\cdot\text{Leaves}(t_{i-1})\\
&\dots\\
\text{Leaves}(t_\omega) &= \textstyle\bigcup_{i\in\mathbb{N}} 1^i0\cdot\text{Leaves}(t_i)
\end{aligned}$$

We prove that $\tilde{r}_{\text{CB}}(t_\omega) = \tilde{r}(\text{KB}(t_\omega)) + 1$.

For each $i > 1$, $\text{KB}(t_{i+1}) = \text{KB}(t_i).\omega + \omega^*$. In particular since $\text{KB}(t_1) = \omega + \omega^*$, for $i > 1$, by induction $\text{KB}(t_i) = \omega + \zeta.\omega^{i-1} + \omega^*$.

$$\text{KB}(t_\omega) = \left(\sum_{i\geq 0}\text{KB}(t_i)\right) + \omega^* = \omega + \zeta.\omega^\omega + \omega^*$$

By another induction, it is easy to see that $\tilde{r}_{\text{CB}}(t_i) = i$. For each $j \in \mathbb{N}$ and $i > j$, $\partial^j(t_i)$ is infinite. So in $\partial^j(t_\omega)$ there is infinitely many infinite branches. In particular the branch 1^ω is in each $\partial^j(t_\omega)$ and therefore in $\partial^\omega(t_\omega)$. To sum up,

	$\text{KB}(t)$	$\tilde{r}(\text{KB}(t))$	$\tilde{r}_{\text{CB}}(t)$
t_1	$\omega + \omega^*$	1	1
t_2	$\omega + \zeta.\omega + \omega^*$	2	2
t_3	$\omega + \zeta.\omega^2 + \omega^*$	3	3
$\dots$			
t_3	$\omega + \zeta.\omega^{i-1} + \omega^*$	i	i
$\dots$			
t_ω	$\omega + \zeta.\omega^\omega + \omega^*$	ω	$\omega + 1$

■

Proposition 4.4.11. *For any two prefix binary tame trees t and t', if $t \equiv t'$ then $\tilde{r}_{\text{CB}}(t) = \tilde{r}_{\text{CB}}(t')$ and $\tilde{r}(\mathbf{Fr}(t)) = \tilde{r}(\mathbf{Fr}(t'))$.*

Proof. Let t and t' be two tame binary trees s.t. $t \equiv t'$. Let h be a bijection from $\text{Dom}(t)$ to $\text{Dom}(t')$ which preserves the prefix relation. As h commutes with d, we have $\partial^\alpha(\text{Dom}(t')) = h(\partial^\alpha(\text{Dom}(t)))$ for every ordinal α so $\tilde{r}_{\text{CB}}(t') = \tilde{r}_{\text{CB}}(t)$. Prop. 4.4.7 implies that $\tilde{r}(\mathbf{Fr}(t)) = \tilde{r}(\mathbf{Fr}(t'))$. □

As the definition of the CB-rank does not use the relative order between the sons of a node, it follows that two prefix deterministic trees having the same underlying unordered tree have frontiers of the same $\sim$-rank.

4.4.4 Hausdorff rank of scattered orders in Graph_n

By using invariance of $\tilde{r}$ over $\equiv$, we get the main result of this section.

Theorem 4.4.12. *For all $n \geq 0$, every scattered linear order in Graph_n has an Hausdorff rank strictly less than $\omega \uparrow\uparrow n$.*

Proof. Let L be a scattered linear order in Graph_n. By Theorem 4.2.6 and Proposition 4.4.1, there exists a binary prefix tame tree $t \in \mathrm{Tree}_n$ such that $L \simeq \mathbf{Fr}(t)$. By Prop. 4.4.3 there exists a well-ordered tree $t' \in \mathrm{Tree}_n$ such that $t \equiv t'$.

By Prop. 4.4.11, we have that $\tilde{r}(\mathbf{Fr}(t)) = \tilde{r}(\mathbf{Fr}(t'))$. As t' is a well-ordered tree in Tree_n, its frontier is an ordinal in Graph_n. Hence by Theorem 4.3.8, $\mathbf{Fr}(t') < \omega \uparrow\uparrow n+1$ and hence $\tilde{r}(\mathbf{Fr}(t')) < \omega \uparrow\uparrow n$. □

Remark 4.4.13. Obviously the converse to this theorem is not true; there are uncountably many scattered orders of Hausdorff rank less than $\omega \uparrow\uparrow n$ but there are only countably many linear orderings in Graph_n.

In practical terms, consider a non-recursive sequence $(a_i)_{i\in\mathbb{N}}$ in $\{\mathbf{1}, \mathbf{2}\}^\omega$. The scattered order

$$a_0 + \zeta + a_1 + \zeta + a_2 + \dots$$

has Hausdorff rank 2. But as the sequence of a_i can be reconstructed by an MSO-formula, it has an undecidable MSO-theory, so it does not belong to the pushdown hierarchy. ■

By Proposition 4.4.7, the upper-bound of Theorem 4.4.12 directly translates the CB-rank of prefix binary tame trees in Tree_n. This leads to the following upper bound for all deterministic trees in Tree_n.

Theorem 4.4.14. *For every deterministic tree $t \in \mathrm{Tree}_n$, $r_{\mathrm{CB}}(t) \leq \omega \uparrow\uparrow n$.*

Proof. Let t be a deterministic tree in Tree_n. We can assume w.l.o.g. that t is binary.

Consider the tree t' obtained by adding a leaf to every node which is its left-most son. Clearly t' is prefix, belongs to Tree_n and $r_{\mathrm{CB}}(t) = r_{\mathrm{CB}}(t')$. The construction of Prop. 4.2.1 gives a prefix binary tree $t'' \in \mathrm{Tree}_n$ with the same CB-rank.

As the result is already established for tame trees, we can also assume that t is not tame. Note that for every deterministic non-tame tree t,

$$r_{\mathrm{CB}}(t) \leq \lambda = \sup\{r_{\mathrm{CB}}(t_{/u}) \mid u \in \mathrm{Dom}(t) \text{ and } t_{/u} \text{ tame}\}.$$

Indeed, $d^\lambda(t)$ has no tame subtrees. If t has no tame subtrees then $d(t) = t$. Hence $d(t) \leq \lambda$.

By Proposition 2.5.5 any subtree of $t \in \mathrm{Tree}_n$ is also in Tree_n. By Proposition 4.4.7 and Theorem 4.4.12 we have, for all $u \in \mathrm{Dom}(t)$ such that $t_{/u}$ is tame, $r_{\mathrm{CB}}(t_{/u}) < \omega \uparrow\uparrow n$. Hence $r_{\mathrm{CB}}(t_{/u}) \leq \omega \uparrow\uparrow n$. □

The inequality is conjectured to be strict.

4.5 Finite combs

As foretold in Section 3.4.4, known examples of graphs separating the hierarchy include the prefix tree $\mathcal{K}_f$ of well-chosen functions $f : \mathbb{N} \mapsto \mathbb{N}$, defined by $\text{Leaves}(\mathcal{K}_f) = \{1^i 0^{f(i)}\}$. In particular, let the $\exp(m, n, i)$ operation be defined by

$$\begin{aligned} \exp(m, 0, i) &= i \\ \exp(m, n+1, i) &= m^{\exp(m,n,i)}, \end{aligned}$$

then the tree $\mathcal{K}_{\exp(2,3n,\cdot)}$ was shown in [Blu08] not to belong to Graph_n, using properties on the size of stacks of n-order-pushdown automata. It was then conjectured that already $\mathcal{K}_{\exp(2,n,\cdot)} \notin \text{Graph}_n$.

This section proves this result. More generally, we consider some comb-shaped graphs of the hierarchy, *i.e.* acyclic graphs with only one simple infinite path. We state a result of the maximal size of each subgraph not intersecting this path. The proof of this result is very related to the methods employed in Section 4.3.

Recall that a *comb* is a deterministic tree t over a ordered finite alphabet with a greatest label 1 such that the only infinite branch is 1^ω. In the hierarchy, an equivalent definition specializes this label. A *#-comb* is a comb on $\Sigma \cup \{\#\}$ with a greatest label $\# \notin \Sigma$ such that each arc is labeled by # iff this arc is in the infinite branch of ω. The subtree of domain $\#^i\Sigma^* \cap \text{Dom}(t)$ is noted t_i. We alter slightly this definition to get more general graphs.

Definition. Let Σ be an alphabet and $\# \notin \Sigma$ another letter. A *#-comb-graph* over Σ is a deterministic graph G of labels $\Sigma \cup \{\#\}$ with a root r and the following properties. Let $V_i = \{x \mid r \xrightarrow{\#^i\Sigma^*} x\}$ and G_i the subgraph of support V_i :

- the subset of arcs labeled by # form an infinite path from r;
- for all i, G_i is an acyclic graph;
- the only arc between V_i and $V_{j\neq i}$ can be #.

The goal of this section is to get a upper bound for each $|V_i|$ when G is a comb-graph of the hierarchy. This is the Theorem 4.5.3.

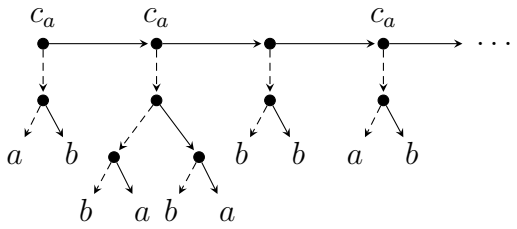
We first remark that comb-graphs of Graph_n cannot be larger than combs of Tree_n.

Lemma 4.5.1. *If $G \in \text{Graph}_n$ is a #-comb-graph, there is a comb $t \in \text{Tree}_n$ such that for all i, there is a $j \leq i$ such that $|V_i| \leq |\text{Dom}(t_j)|$.*

Proof. Let G be a #-comb-graph of support V. Since each G_i is a finite deterministic acyclic graph, there is a MSO-definable order $<$ on V_i — for instance, lexicographic order. This order can be extended to V with $x < y$ iff $x \in G_i, y \in G_j$ and $i < j \vee (i = j \wedge x < y)$. So there is an interpretation from G to the ω-word $w = \prod_{i\geq 0}(ab^{|V_i|})$: each $x \in V_i$ is colored in a when it is the smallest of V_i, and b otherwise.

By Corollary 4.2.7, there is a full binary prefix tree t in Tree_n (on the alphabet $\{0,1\}$) with only one infinite branch 1^ω such that $\mathbf{Fr}(t) \simeq w$.

For instance, t could be the following tree (the color c_a is defined later on) :



We may call t_i the subtree of root 1^i0, if any. The problem is that a subword $ab^{|V_i|}$ may be "scattered" in several t_{j_1}, t_{j_2}, ... To get the required result, we are going to merge these subtrees.

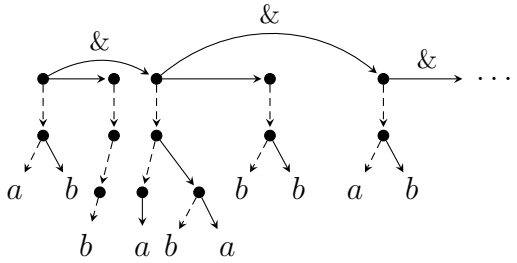
We color each node 1^i with c_a if $a \in \mathbf{Fr}(t_i)$. In particular, ε is colored. Let $H \in \mathrm{Graph}_{n-1}$ and $s \in V_H$ be such that $\mathrm{Unf}(H,s)$ is this marked tree. Let $\mathcal{I}(H)$ be the graph where there is a &-arc $x \xrightarrow{\&} y$ when

$$r \xrightarrow[H]{1^*} x \xrightarrow[H]{1^+} y, \quad (c_a,x),(c_a,y) \in H,$$
$$\text{and for all } x \xrightarrow{1^+} z \xrightarrow{1^+} y, (c_a,z) \notin H.$$

In $\mathrm{Unf}(\mathcal{I}(H),s)$, we remove the subtrees with root $\&^i u$ such that

1. there is no a-leaf $\&^i v$ with $v \leq_{\mathrm{lex}} u$,
2. or there are at least one a-leaf $\&^i 1v$ such that $1v <_{\mathrm{lex}} u$ or $v \in u0^*$.

We note t_i the subtree of domain $\&^i(0+1)^*$. Condition 1 checks that each such subtree begins with an a. Condition 2 looks for the first a-leaf in $\&^i 1(0+1)^*$ and removes subtrees of root greater or equal in lexicographic order. We note t' this new tree. Following our previous example, t' is shown below.



In short, t' is a &-comb having the same frontier than t, and such that for each i, $\mathbf{Fr}(t'_i)$ begins with the letter a. So each $ab^{|G_i|}$ is contained in some $\mathbf{Fr}(t'_j)$, which shows the result. □

If $n \geq 1$, for any $t \in \mathrm{Tree}_n$ there is a graph H in Graph_{n-1} and $s \in V_H$ such that $t = \mathrm{Unf}(H,s)$. As usual, we suppose that s is a root of H.

Lemma 4.5.2. *Let $t \in \mathrm{Tree}_n$ be a #-comb and $t = \mathrm{Unf}(H, s)$. Let $S = \{x \xrightarrow[H]{\#} y\}$.*

- *Either S is finite, and then the sequence $(t_i)_{i\geq 0}$ is ultimately periodic;*
- *or S is an infinite path from s.*

Proof. Recall that t and H are deterministic. Since there is a infinite path labeled by # in t, there is a infinite or ultimately periodic path labeled by # in H, starting from s. □

In particular, for $n = 1$, H is a finite graph, so the set S is necessarily finite, and the sequence $(|\mathrm{Dom}(t_i)|)_{i\geq 0}$ is bounded. By Lemma 4.5.1, so are the $(|V_i|)_{i\geq 0}$.

For further levels, the following result states that the bound is n-exponential in the length of the comb. For a finite tree t, let $\mathrm{depth}(t)$ be the maximal path size in t.

Theorem 4.5.3. *For $n > 0$, let t be a #-comb of Tree_{n+1} and G a #-comb-graph of Graph_{n+1}. There is a constant C such that for all i,*

$$\begin{aligned} \mathrm{depth}(t_i) &\leq \exp(2, n-1, C(i+1)), \\ |V_i| &\leq \exp(2, n, C(i+1)). \end{aligned}$$

Proof. Let t be a #-comb over Σ, and H and s as in Lemma 4.5.2. By Lemma 4.5.1, the result on G is a direct consequence of the result on t. Indeed, since t_i is a finite deterministic Σ-tree, then $|\mathrm{Dom}(t_i)| \leq |\Sigma|^{\mathrm{depth}(t_i)}$. So there is a constant C' such that $|\mathrm{Dom}(t_i)| \leq \exp(2, n, C'(i+1))$.

First note that deterministic structures of Tree_1 and Graph_1 are regular. Following notations in [MS85] and Section 2.5, they have therefore finitely many end-isomorphisms, which means that for any #-comb t in Tree_1, the sequence $(|\mathrm{Dom}(t_i)|)_{i\geq 0}$ is bounded. A similar result applies to #-comb-graphs.

In H, for $i \geq 0$, let s_i be the vertex such that $s \xrightarrow[H]{\#^i} s_i$. By Lemma 4.5.2, if $\{s_i\}_{i\geq 0}$ is finite, the result is trivial. Suppose otherwise. We call the #-level $\ell^{\#}(x)$ of a vertex $x \in V_H$ the least i such that $s_i \xrightarrow{\Sigma^*} x$. Let $V_{H_i} = \{x \mid \ell^{\#}(x) = i\}$ and H_i be the associated subgraph.

The transformation $\mathcal{I}$ removing Σ-arcs between any V_{H_i} and $V_{H_{j\neq i}}$ is a monadic interpretation. Moreover, we now that there is exactly one infinite branch in t, so there cannot be any infinite branch in any $\mathrm{Unf}(H_i, s_i)$, which means each H_i is finite and acyclic. So $\mathcal{I}(H)$ is a #-comb-graph of Graph_{n-1}. A path in t_i is smaller than $\sum_{j\leq i} |V_{H_j}|$.

- If n=2, then $|V_{H_i}|$ is bounded by some C, so

$$\mathrm{depth}(t_i) \leq \sum_{i\leq \ell^{\#}(x)} |V_{H_j}| \leq C.i.$$

- Otherwise, by induction $|V_{H_j}| \leq \exp(2, n-1, C(i+1))$ and

$$\mathrm{depth}(t_i) \leq \sum_{i\leq \ell^{\#}(x)} |V_{H_j}| \leq \exp(2, n-1, C(\ell^{\#}(x)+2)).$$

□

Chapter 5

Schemes and morphic words

In the previous chapter, we left aside the question of the coloring of orders. The first class of colored infinite graphs that comes to the mind are the infinite words, or orders of type ω. Even for such simple objects, we do not know precisely which of them inhabit the pushdown hierarchy. The aim of this chapter is to open the way to a characterization of infinite words of the hierarchy. At the first level, they are well-known.

Proposition 5.0.4. *The ω-words of* Graph_1 *are the ultimately periodic words.*

Proof. For any colored ω-order, it is easy to see that the corresponding colored infinite path is also in Graph_1. This a regular graph, and has therefore finitely end-isomorphisms (see Section 2.5) starting from the root. It follows that there are two isomorphic end-decompositions at step n and m so the structure is ultimately periodic with period $|m - n|$. □

We turn our attention on the next levels, namely Graph_2 and Graph_3. A direct consequence of [Cau02, Prop. 3.2] is that infinite words well-known as morphic words belong to Graph_2. For the other direction, by Corollary 5.1.6 we only need to consider the frontier of trees solutions of order-1 schemes whose frontier is of order type ω. Moreover, by Proposition 4.2.1, such a tree can be chosen binary, prefix and full. This tree has therefore exactly one infinite branch, which is 1^ω. As mentioned in the introduction, deterministic trees of Tree_2 can be described by the notion of recursion schemes. We may therefore plug the constrained shape of our trees into this setting to get the characterization of words.

For a slightly more powerful result, we notice that if the frontier of a deterministic tree has an initial segment of type ω, then this segment can be interpreted by an MSO-formula. If we are in the pushdown hierarchy, we can therefore find a correspond comb. For any deterministic tree, we call *ω-frontier* this initial segment when it exists.

Proposition 5.0.5. *For any safe scheme admitting a ω-frontier w, there is a safe scheme of the same order generating a comb with frontier w.*

Proof. If a deterministic tree in the pushdown hierachy has an ω-frontier w, then this order can be selected by a monadic interpretation. Indeed, consider the set of leaves by

lexicographic order. Once the smallest leaf x_0 in lexicographic order has been found, the initial segment X_0 of underlying type ω is the smallest such that for all $x \in X_0$ and $X \subseteq X_0$ with $x, x_0 \in X$, x has an immediate predecessor in X.

So w is a word of the same level of the hierarchy. By Theorem 4.2.6, there is therefore a comb in the class of trees of the same level were each node has degree 0 and 2, with frontier w. This means that there is a safe scheme generating this comb. □

If words of Graph_2 are morphic words, extending the method to the next level would hopefully bring a natural extension. Since morphic words are well-known, this is not the first attempt for such a generalisation. Already [CT02] proposes more general predicates in the framework of decidable monadic theory. Some more recent propositions tweak an automatic presentation of morphic words, for instance allow the automaton to be infinite [lG06] or generalize the underlying order [Bár08]. Another direction is more pushdown-automaton related [FS06, Mar07] and relates to the HDT0L definition of morphic words (see [AS03] for details).

This rest of this chapter is organized as follows. We first present recursion schemes and their relationship with the hierarchy. Then, we define morphic words and that frontiers of type ω of order-1 recursion schemes are exactly morphic words. Then we extend this proof to so-called hyperalgebraic trees, *i.e.* trees of Tree_3. We first introduce the notion of order-2 morphic words, and prove that they correspond exactly to ω-frontiers of order-2 recursion schemes. We therefore obtain a characterization of the ω-words of Graph_3.

5.1 Recursion schemes

We first give a definition of recursion schemes, then recall why schemes with the safety constraint are term trees of the hierarchy.

5.1.1 Definition

We borrow the definitions from [KNU02]. The notion of *recursion scheme*, or *term grammar*, gives a direct presentation of infinite tree by successive rewriting of terms.

Types and terms

First of all, these terms are constrained by their type; the first definition therefore concerns these objects. The set of *types* $\mathcal{T}$ is built from a unique *basic* type $\mathbf{o}$ and the binary operator $\to$.

$$\tau := \mathbf{o} \mid \tau_1 \to \tau_2$$

The operator $\to$ is associative to the right. The *order* or *level* $\ell(\tau)$ of a type is defined by $\ell(\mathbf{o}) = 0$ and $\ell(\tau_1 \to \tau_2) = \max(1 + \ell(\tau_1), \ell(\tau_2))$. A type $\tau_1 \to \ldots \to \tau_n \to \mathbf{o}$ is

homogeneous if $\ell(\tau_1) \geq \cdots \geq \ell(\tau_n)$. The type $\mathbf{o} \to \ldots \to \mathbf{o} \to \mathbf{o}$ with $n+1$ times $\mathbf{o}$ is written $\mathbf{o}^n \to \mathbf{o}$.

A *typed alphabet* is a set Γ of symbols with types in $\mathcal{T}$. We note $f : \tau$ when $f \in \Gamma$ is of type τ. By notation, $\ell(f) = \ell(\tau)$, and the order of the alphabet is $\ell(\Gamma) = \max_{f\in\Gamma} \ell(f)$. The set of *applicative terms* $\mathsf{AT}(\Gamma)$ is defined by $\Gamma \subseteq \mathsf{AT}(\Gamma)$ and if $f : \tau_1 \to \tau_2$ and $x : \tau_1$, then $(fx) : \tau_2 \in \mathsf{AT}(\Gamma)$. We say then that x an *operand* of f. By mirroring type association, applicative terms are associative to the left : $fxy = (fx)y$. A *term* is an applicative term of type $\mathbf{o}$; the set of terms is noted $\mathsf{T}(\Gamma)$. The *arity* ρ of a term is 0, and $\rho(f) = \rho(fx)+1$ otherwise. An *applicative subterm* is an occurence of some $t = fx_1 \ldots x_n$ where n is maximum for this occurence; it is a *subterm* if $n = \rho(f)$.

Example 5.1.1. Let $\Sigma = \{a, b, g, f\}$ with $\rho(a) = \rho(b) = 0$, $\rho(g) = 1$ and $\rho(f) = 2$. Then $f(a)$ is an applicative term of arity 1, and $f(g(a), b)$ is a term, thus of arity 0. ▲

Recursion schemes

We are now ready to define the base rules of recursion schemes. Let X be a typed alphabet of *variables*. We note $t[x := t']$ with $x : \tau \in X$ the term where each occurence of the variable x is replaced by a applicative term $t' : \tau$. A *recursion scheme* is a tuple $\mathfrak{S} = (\Sigma, N, S, E)$ where Σ is a finite typed 1-order alphabet of (lowercase) *terminals*, N is a finite typed alphabet of (uppercase) *nonterminals*, $S : \mathbf{o} \in N$ is the *starting nonterminal* and E is a set of *productions* in bijection with N, of the form

$$Fx_1 \ldots x_{\rho(F)} \Rightarrow w_F$$

where $F : \tau_1 \to \ldots \to \tau_{\rho(F)} \to \mathbf{o} \in N$; for all i, $x_i : \tau_i \in X$ and w_F is a term in $\mathsf{T}(\Sigma \cup N \cup \{x_1, \ldots, x_{\rho(F)}\})$. The order of the scheme is $\ell(\mathfrak{S}) = \ell(N)$. The set of all n-order schemes is noted $\mathcal{S}_n$.

To each scheme $\mathfrak{S}$ is associated a *rewriting relation* $\underset{\mathfrak{S}}{\Longrightarrow} \subseteq \mathsf{T}(\Sigma \cup X \cup N)^2$. Informally, a subterm which head is a nonterminal F is replaced by its related term w_F where variables are in turn replaced by the actual arguments of F. In practice,

- $Ft_1 \ldots t_{\rho(F)} \underset{\mathfrak{S}}{\Longrightarrow} w_F[\forall i, x_i := t_i]$ if there is a production $Fx_1 \ldots x_{\rho(F)} \Rightarrow w_F$ in E with $x_i : \tau_i$ and $t_i : \tau_i$ for all i.
- $t \underset{\mathfrak{S}}{\Longrightarrow} t'$, then $(st) \underset{\mathfrak{S}}{\Longrightarrow} (st')$ and $(ts) \underset{\mathfrak{S}}{\Longrightarrow} (t's)$ whenever the applicative terms exist.

Limit trees

The definition of schemes yet only produces finite terms. To reach infinity, we need the notion of limit tree. This limit in turn demands to build a converging family of trees. The *approximation* $t^\perp$ of a term t on $\Sigma \cup X \cup N$ is the term on $\Sigma \cup \{\perp\}$ where nonterminals have been pruned, that is

- if $t = ft_1 \dots t_{\rho(f)}$ with $f \in \Sigma \cup X$, $t^\perp = ft_1^\perp \dots t_{\rho(f)}^\perp$,
- if $t = Ft_1 \dots t_{\rho(F)}$ with $F \in N$, then $t^\perp = \perp$.

Let π be a projection from terms on $\Sigma \cup \{\perp\} \cup X$ to (finite) trees, where each subterm $fx_1 \dots x_{\rho(f)}$ is mapped to a subtree δ with root r, arcs $r \xrightarrow{i} \pi(x_i)$, and color $\delta(r) = f$. An approximation partial ordering can be defined on trees by $\delta' \sqsubseteq \delta$ if $\mathrm{Dom}(\delta') \subseteq \mathrm{Dom}(\delta)$ and, for each $w \in \mathrm{Dom}(\delta')$, $\delta'(w) = \delta(w)$ or $\delta'(w) = \perp$. Successive approximations are ordered by this ordering : for two terms $t_1 \underset{\mathfrak{S}}{\Longrightarrow} t_2$, we have $\pi(t_1^\perp) \sqsubseteq \pi(t_2^\perp)$. The scheme is a confluent grammar, so each term t on $\Sigma \cup N \cup X$ generates a unique tree

$$[\![t]\!] = \sup\{\pi(s^\perp) \mid t \underset{\mathfrak{S}}{\Longrightarrow} s\}.$$

For any scheme $\mathfrak{S} = (\Sigma, N, S, E)$, we note $[\![\mathfrak{S}]\!] = [\![S]\!]$ the limit tree of the scheme.

Example 5.1.2. Theorem 3.1.2 and Corollary 5.1.6 state that any ordinal smaller than $\omega \uparrow\uparrow n + 2$ is the frontier of a n-order scheme. We show schemes corresponding to the successive towers of ω.

A tree yielding ω is simply given by the 0-order scheme

$$\mathfrak{S}_0 = (\{f, a\}, \{S_0\}, S_0, \{S_0 \Rightarrow f(a, S_0)\}).$$

At the next level, a tree yielding ω^ω is given by $\mathfrak{S}_1 = (\{f, a\}, \{S_1, F_1, G_1\}, S_1, E_1)$ where E_1 is shown in Figure 5.1. Informally, the nonterminal F adds complexity whereas G propagates it. Recursively, a tree yielding $\omega \uparrow\uparrow n$ is $\mathfrak{S}_n = (\{f, a\}, \{S_n, F_n, G_1, \dots, G_n\}, S_n, E_n)$ with E_n defined in Figure 5.2.

Here the types used are

$$\begin{array}{rcl} \tau_1 & = & \mathbf{o}, \\ \text{for } n > 1, \quad \tau_n & = & \tau_{n-1} \to \dots \to \tau_1 \to \mathbf{o} \end{array}$$

and for all $n > 0$, $x_n : \tau_n$ and $F_n, G_n : \tau_{n+1}$. ▲

5.1.2 Schemes in the pushdown hierarchy

A scheme is *safe* if for any production rule $Fx_1 \dots x_n \Rightarrow w_F$, and for any applicative subterm t of w_F, there is no occurence of some x_i in t with $\ell(x_i) < \ell(t)$. The set of order-n safe schemes is noted $\mathcal{S}_n^{\mathrm{safe}}$. Note that for $n < 2$, safety is not a restriction. The safety has been previously called the *derived types* property [Dam82].

Example 5.1.3. From [KNU02], the scheme with production rule

$$\begin{array}{rcl} S & \Rightarrow & F(g, a, b) \\ F(\varphi, x, y) & \Rightarrow & f(F(F(\varphi, x), y, h(y)), f(\varphi(y), x)) \end{array}$$

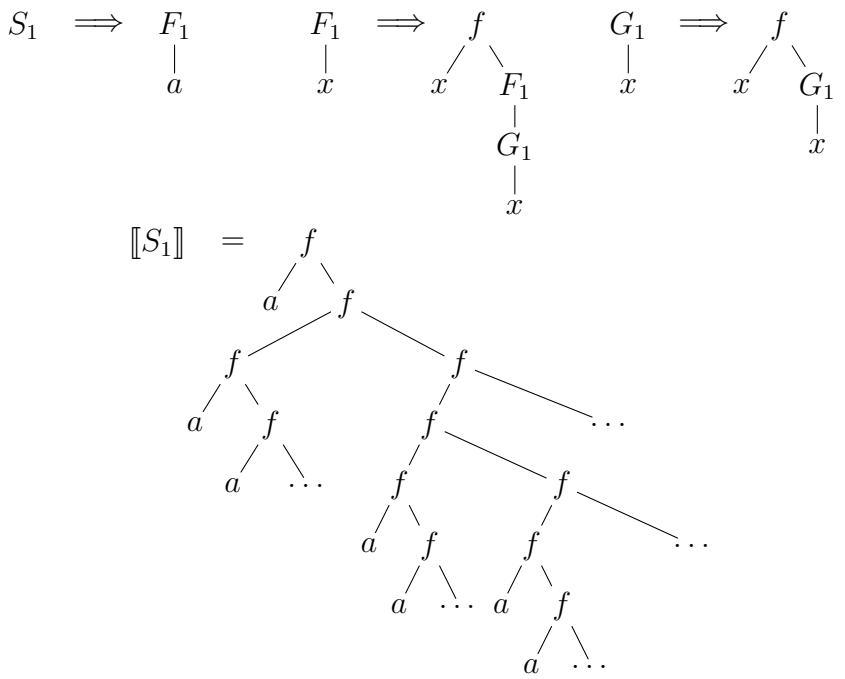


Figure 5.1: Rules for $\mathfrak{S}_1$, of frontier ω^ω.

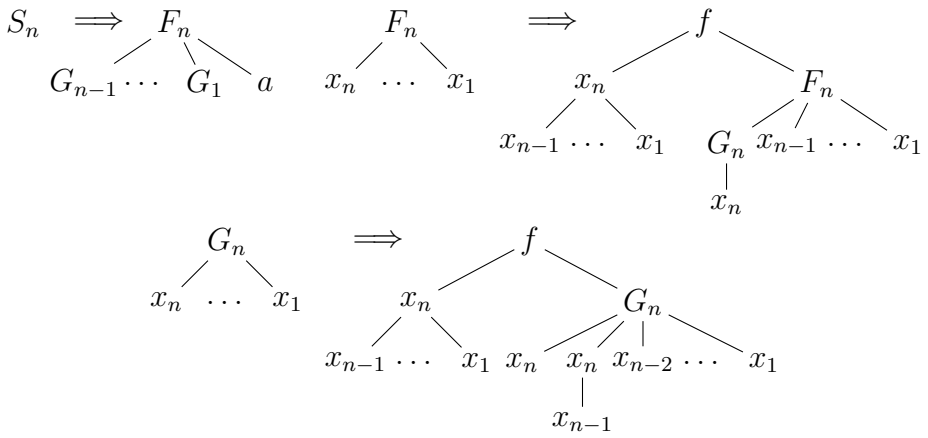


Figure 5.2: General rules for $\mathfrak{S}_n$, of frontier $\omega \uparrow\uparrow (n+1)$.

is unsafe, because there is a subterm $F(\varphi, x)$ such that $\ell(x) < \ell(F(\varphi, x))$. It is conjectured that this scheme produces a tree which cannot be produced by any safe scheme. ▲

The question whether safe schemes produce the same trees than unsafe schemes has been recently solved by the negative using a language defined by Urzyczyn. For more details, see [Par10] and [AdMO05, Prop. 7.13].

Definition. A *term tree* is a deterministic tree t colored by an alphabet Σ and labeled by $[1, k]$ where k is the maximum out-degree, and the following constraints :

- if $u \cdot i \in \mathrm{Dom}(t)$, then $i = 1$ or $u \cdot (i-1) \in \mathrm{Dom}(t)$;
- all nodes with the same color f have the same out-degree.

The relationship between schemes and the pushdown hierarchy is that trees generated by safe schemes of order n are exactly term trees on the level $n+1$ of the hierarchy. This is an application of [Cau02, Theorem 3.5].

Proposition 5.1.4. *Limit trees of safe schemes of order n are term trees of* Tree_{n+1} *up to isomorphism.*

Proof. Let h be a rational mapping as defined in Section 2.4.1 with the following constraints. Suppose that such that for each a, $h(a)$ is recognized by a finite deterministic automaton (S, i, T, η) where

- each final state (in T) is terminal, *i.e.* has no output arcs;
- for all $q \in S$, each set $P = \{q \mid q \xrightarrow{c} p \wedge c \in \Gamma\}$ is such that if $q \in P$ and $q \to p$ then $p \notin P$;
- it is strongly deterministic on arc labels : if $q \xrightarrow{c} p, q \xrightarrow{d} p'$ and $a \in \Sigma$, then $c = d$. In other words, the disjonctions are made on colors.

Such a mapping is called *deterministic rational mapping.* The set of all these mappings is called DRat.

Theorem 5.1.5 ([Cau02])**.** *For $n > 0$, $[\![\mathcal{S}^{\mathrm{safe}}_{n+1}]\!] = \mathrm{Unf} \circ \mathrm{DRat}^{-1}([\![\mathcal{S}^{\mathrm{safe}}_{n}]\!])$.*

At the first level, the recoloring of $[\![\mathcal{S}_0]\!]$ are regular trees. To get Prop. 5.1.4, it it enough to show that deterministic trees of Tree_{n+1} are the image of Tree_n under the mapping $\mathrm{Unf} \circ \mathrm{DRat}^{-1}$, and more precisely that each deterministic graph of Graph_n is $h^{-1}(t)$ for some deterministic $t \in \mathrm{Tree}_n$ and $h \in \mathrm{DRat}$.

Let $G \in \mathrm{Graph}_n$. By Prop. 4.1.2, there is a deterministic tree $t \in \mathrm{Tree}_n$ of color set Γ, a family of simple TWA $\{\mathcal{A}_a\}_{a \in \Sigma}$ and a recoloring μ such that $G = \mu(\mathcal{A}(t))$. We build

$h \in \mathrm{DRat}$ such that $G = h^{-1}(t)$. For $a \in \Sigma$, let $\mathcal{A}_a = (Q, q_0, \mu_F, \delta)$ be the automaton recognizing $h(a)$ and let $A = (S, i, T, \eta)$.

$$\begin{array}{rrcll} & S & = & Q \cup (Q \times \mathcal{P}(\Gamma)) & \\ & i & = & q_0 & \\ & T & = & (q,c) & \text{such that } q \in \mu_F(c) \\ & \eta(q,c) & = & (q,c) & \text{for each } q \in Q, c \in \mathcal{P}(\Gamma) \\ \forall a \in \Sigma, & \eta((q,c),\bar{a}) & = & p & \text{for each } \delta(q,c) = (p, \uparrow) \\ & \eta((q,c),a) & = & p & \text{for each } \delta(q,c) = (p,a) \end{array}$$

Each final state is terminal, since $\delta(q,c) = \emptyset$ if $q \in \mu_F(c)$. Each $\mathcal{P}(\Gamma)$-transition is either a final state or followed by a Σ-transition. The determinism condition is also fulfilled : if $\eta((q,c),a) = (p,0)$ and $\eta((q,c),b) = (p,0)$, then $(p,a),(p,b) \in \delta(q,c)$, and since $\mathcal{A}_a$ is deterministic, $a = b$.

For recoloring, we simply set $h(c) = \mu(c)$ for each $c \in \mathcal{P}(\Gamma)$. □

It is easy to tranform any deterministic tree into a term tree with the same frontier. This remark allows the following corollary of Theorem 4.2.6.

Corollary 5.1.6. *A linear order colored by* Γ *is in* Graph_n *if and only if it is the colored frontier of some tree limit of a safe recursion scheme of order* $(n-1)$ *with one terminal* f *of arity* 2 *and terminal of arity* 0 *for each* $c \in \Gamma$.

5.2 Morphic words

Morphic words are well-known ω-words. They can be seen as a generalisation of automatic sequences, but we prefer to skip this latter definition and give the direct presentation. For a complete introduction, see [AS03].

In this section, after the preliminaries we present a construction of the morphic words in the hierarchy. Then we prove that morphic words are the only possible words of Graph_2 by looking at leaves of trees produced by recursion schemes.

5.2.1 Definition and properties

Let Σ be an alphabet. A *morphism* on Σ^* w.r.t. concatenation is a mapping τ such that $\tau(ab) = \tau(a)\tau(b)$. Now let $\tau : \Sigma \mapsto \Sigma^*$ a morphism. Suppose there is a letter $\Delta \in \Sigma$ such that the first letter of $\tau(\Delta)$ is Δ. In this case the following sequence admits a limit :

$$\begin{array}{rcl} \tau(\Delta) & = & \Delta \cdot u \\ \tau^2(\Delta) & = & \Delta \cdot u \cdot \tau(u) \\ \cdots & & \\ \tau^\omega(\Delta) & = & \Delta \cdot u \cdot \tau(u) \cdot \tau^2(u) \cdot \ldots \end{array}$$

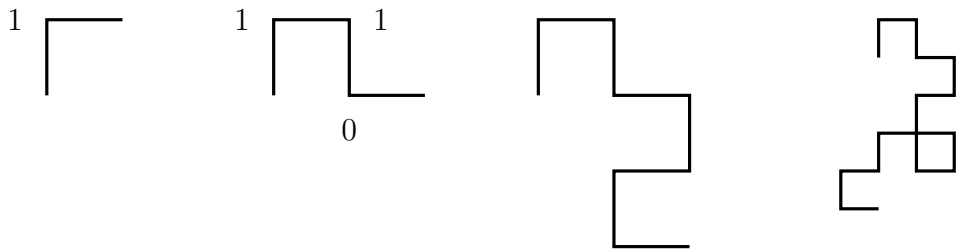


Figure 5.3: Paperfolding sequence.
When the corners are set to right angles like in this drawing, the resulting pattern is known as the dragon curve fractal.

Let $\sigma : \Sigma \mapsto \Gamma$ be another morphism. The set of *morphic words* are the words in the form $\sigma(\tau^\omega(\Delta)))$.

Example 5.2.1. The regular paperfolding sequence (see [AS03, Ex. 5.1.6]), or dragon curve sequence, is obtained by folding iteratively a piece of paper in the same direction. By unfolding the paper and looking at the direction of the corners, we get a word on two letters. For n foldings, we get the n^{th} word of the following sequence; see Figure 5.3.

$$\begin{array}{l} 1 \\ 110 \\ 1101100 \\ 110110011100100 \\ \ldots \end{array}$$

The limit word is a morphic sequence over the alphabet $\{0,1\}^2$. Define

$$\begin{array}{lcllcl} \tau(00) & = & 1000 & \tau(10) & = & 1100 \\ \tau(01) & = & 1001 & \tau(11) & = & 1101 \end{array}$$

Then $\tau^\omega(11)$ is the paperfolding sequence. ▲

We can chose σ to have a specific form. A *coding* is a morphism σ such that for all a, $|\sigma(a)| = 1$.

Theorem 5.2.2 ([AS03])**.** *If f, g are two morphisms such that $g(f^\omega(a))$ exists, then there is a letter Δ, a non-erasing morphism τ such that $\tau(\Delta)$ begins with Δ, and a coding σ such that $g(f^\omega(a)) = \sigma(\tau^\omega(\Delta))$.*

We are particularly interested by the logical properties of morphic words. The following result was obtained in [CT02] for even larger sets of ω-words.

Theorem 5.2.3 ([CT02])**.** *The monadic theory of a morphic word is decidable.*

As with uncolored ordinals, this result naturally rises the question : which morphic words are in the hierarchy? The following sections answers that they are in fact exactly words of Graph_2.

5.2.2 Construction in the pushdown hierarchy

The goal of this section is to prove that words of Graph_2 are exactly the morphic words. We begin with the easy direction. It is proved in [Cau02] that morphic words are terms of Tree_3. If it is not ultimately periodic, such a tree must be unfolded from itself, so morphic words are in Graph_2.

Theorem 5.2.4. *Any morphic word is in* Graph_2.

Proof. We reproduce here a proof by Caucal. Any morphic word can be chosen of the form $\sigma(\tau^\omega(\Delta))$ with τ be a morphism on an alphabet Σ with $\tau(\Delta) \in \Delta\Sigma^*$, and σ a coding $\Sigma \mapsto \Gamma$.

Let $n = \max_{a\in\Sigma}(|\tau(a)|)+1$. We build the following regular graph of support $\mathbb{N}\cup(\mathbb{N}\times\Sigma)\cup\Gamma$, label set $[1,n]$ and colors Γ.

$$\begin{aligned} G \;=\; & \{k \xrightarrow{n} k+1\} \cup \{k \xrightarrow{i} (k,\tau(\Delta)_i) \mid k>0, i>1\} \\ & \cup\{(k,x) \xrightarrow{i} (k-1,\tau(x)_i) \mid k>1, x\in\Sigma, i>0\} \\ & \cup\{(1,x) \xrightarrow{1} \sigma(x) \mid x\in\Sigma\} \cup \{0 \xrightarrow{1} (0,\Delta), (0,\Delta) \xrightarrow{1} \sigma(\Delta)\} \\ & \cup\{(y,y) \mid y\in\Gamma\} \end{aligned}$$

G is a regular graph, so $G \in \mathrm{Graph}_1$. Let $t = \mathrm{Unf}(G,0)$. By induction, for $k>0$, each subtree of root $n^k i$ yields the finite word $\sigma(\tau^{k-1}(\tau(\Delta)_i))$. The leaves of t in lexicographic order form the required morphic word. There is therefore a monadic interpretation building the word from t. □

Example 5.2.5. For instance, the word $abaabaaaab\ldots$ is obtained by

$$\begin{array}{rclcrcl} \tau(\Delta) &=& \Delta baa & & \sigma(\Delta) &=& a \\ \tau(a) &=& aa & & \sigma(a) &=& a \\ \tau(b) &=& b & & \sigma(b) &=& b \end{array}$$

So $\Sigma = \{\Delta, a, b\}$, $\Gamma = \{a, b\}$, $n = 5$. Figure 5.4 shows the part of G accessible from 0. The circled vertices are colored respectively by a and b. ▲

This result implies of course Theorem 5.2.3.

5.2.3 Words in Graph_2 are morphic, direct proof

The converse of Theorem 5.2.4 can be obtained by using methods similar to those used in Chapter 4. We follow here this direction. But it is limited to the first level of the

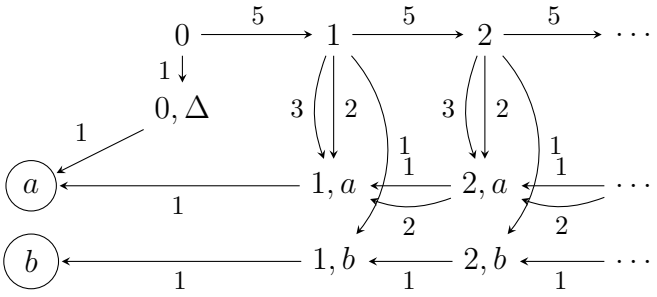


Figure 5.4: A graph which unfolding yields the morphic word $abaab \dots a^{2^i} b \dots$.

hierarchy, so the rest of this chapter will be devoted to an alternative and extensible proof using recursion schemes.

Theorem 5.2.6. *ω-words of* Graph_2 *are morphic words.*

To prove this result, we show that a tree having a frontier of type ω is unfolded from a graph having the shape of the graph of Example 5.2.5.

Let w be an ω-word on Γ of Graph_2. According to Theorem 4.2.6, we consider a prefix full Σ, Γ-tree of Tree_2 which has frontier w. Let t_w be this tree. One more step backwards, let $G \in \mathrm{Graph}_1$ and $r \in V_G$ be such that $t = \mathrm{Unf}(G, r)$. We may suppose that G is accessible from r. There is only one infinite path from r in G.

Moreover, since G is a deterministic graph, we know by [CK01] that G is in fact a regular graph[1]. We actually ignore this property to reach the result, and we prefer to follow the hierarchy backwards one more time : G is it interpreted from a tree $t \in \mathrm{Tree}_1$.

Lemma 5.2.7. *There is a* MSO*-interpretation $\mathcal{I}$ and a deterministic regular tree $t \in \mathrm{Tree}_1$ such that $G = \mathcal{I}(t)$ and t has only one infinite branch.*

Proof. Instead of a MSO-interpretation we know that we may chose a TWA-interpretation $\mathcal{A} = \{\mathcal{A}_a\}_{a \in \Sigma}$ as described in Remark 4.1.4. There is a regular tree t in Tree_1 such that G is $\mathcal{A}(t)$ up to a color projection. As usual, we may suppose that t is the prefix closure of V_G. Let $(r_i)_{i \geq 0}$ be the infinite path from r, and t_r the restriction of t to the prefix closure of $\{r_i\}_{i \geq 0}$. Suppose there are two infinite branches in t_r, with greatest common node z. Then there are infinitely many runs $r_i \xrightarrow[G]{a} r_{i+1}$ of $\mathcal{A}_a$ for some $a \in \Sigma$, where r_i and r_{i+1} have prefixes in different branches. The set of possible states on z being finite, this means that there are $r_i \xrightarrow[G]{a} r_{i+1}$ and $r_j \xrightarrow[G]{a} r_{j+1}$ with $i \neq j$ and thus $r_i \xrightarrow[G]{a} r_{j+1}$, which cannot be since G is deterministic. So t_r has only one infinite branch B.

As in Section 4.5, we may note V_i the subset of V_G such that any path from r to this subset goes through r_i, but not necessarily through r_{i+1}. Each V_i is a finite set.

[1] Given a deterministic prefix-recognizable relation $(U \to V)W$, we have necessarily that $|V| = 1$ and that a word in $U \cdot W$ must be uniquely decomposable. This satisfies exactly the property 3 of [CK01, Theorem 4.6].

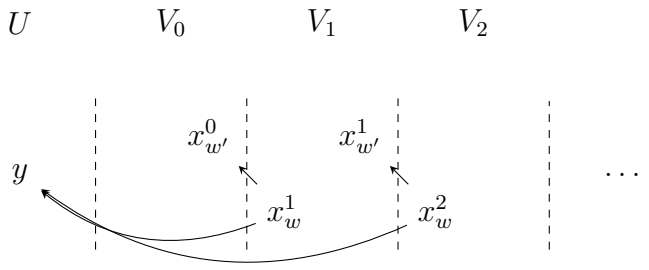


Figure 5.5: General shape of the folded graph.

Suppose there is another infinite branch B' and let $b = B \wedge B'$. Since the tree is deterministic and the V_i are finite, there are infinitely many $i \in \mathbb{N}$ such that r_i is supported by B and some vertex z_i in V_i is supported by B'. Since there is a path from r_i to z_i, we can consider the precise pair $(x_i, y_i$ such that $r_i \underset{G}{\rightarrow} x_i \underset{G}{\overset{a}{\rightarrow}} y_i \underset{G}{\rightarrow} z_i$ and $|x_i \wedge B| > |b|$ and $|y_i \wedge B'| > |b|$. Since t is deterministic and the pairs (x_i, y_i) are pairwise distinct, for all N there is a $i > 0$ and $a \in \Sigma$ such that there are infinitely many $x_i \underset{G}{\overset{a}{\rightarrow}} y_i$ and

$$\begin{aligned} |x_i \wedge B| - |b| &> N \\ |y_i \wedge B'| - |b| &> N. \end{aligned}$$

By applying the same argument than above, we see that there are $i \neq j$ such that there is also a run of $\mathcal{A}_a$ from x_i to y_j, which contradicts the determinism of G. □

So t is a regular tree with one infinite branch. This means that it is periodic in the sense that there are two words u, v with sufficiently large u such that if $t(uw) = c$ for some w, then $t(uv^k w) = c$ for all $k \geq 0$. Indeed, if d_{MS} is the graph decomposition in the sense of [MS85] (see Section 2.5), then there are n_u, n_v such that $d_{MS}^{(n_u+n_v)}(t) = d_{MS}^{(n_u)}(t)$. We may then chose u to be the prefix of size n_u of the branch and v the following factor of size n_v.

For any w such that $v \not\sqsubseteq w$, we note this vertex $x_w^k = uv^k w$. For a given k, there are only finitely many such vertices. Let $V_k = V_G \cap \{x_w^k \mid w \in \Sigma^*\}$ for $k > 0$ and $U = V_G \setminus \bigcup_{k \geq 0} V_k$. These sets are all finite. We also note G_U the subgraph of support U, $V_{\leq k} = U \cup \bigcup_{i \leq k} V_i$ and $G_{\leq k}$ the subgraph of support $V_{\leq k}$. The following technical lemma is illustrated in Figure 5.5.

Lemma 5.2.8. *There is a choice of u, v such that if $k \geq 1$ and $x_w^k \underset{G}{\overset{a}{\rightarrow}} y$, then*

- *either $y \in U$ then $x_w^{k+1} \underset{G}{\overset{a}{\rightarrow}} y$,*
- *or $y \in V_{k-1} \cup V_k \cup V_{k+1}$ so $y = x_{w'}^{k'}$ and $x_w^{k+1} \underset{G}{\overset{a}{\rightarrow}} x_{w'}^{k'+1}$.*

Proof. Suppose $y \in V_G \setminus U$ and note $y = x^{k'}_{w'}$. There is a run of $\mathcal{A}_a$ from x^k_w to $x^{k'}_{w'}$. This run goes by $x^{k'}_{w'} \wedge B$ which is of the form $x^{k'}_z$ with $z \sqsubseteq v$. By the regularity of the tree, there is also a run from x^{k+1}_w to $x^{k'+1}_{w'}$.

Consider the set of $h \in \mathbb{N}$ such that the run of $\mathcal{A}_a$ from x^k_w to $x^{k'}_{w'}$ goes by x^h_z. If $|k'-k|$ is not bounded, then this set is larger than the set of states of $\mathcal{A}_a$; so there are $h \neq h'$ such that $\mathcal{A}_a$ has the same state on $x^{h'}_z$ and x^h_z. Suppose w.l.o.g. that $k' \leq h' < h \leq k$. Then there is a run of $\mathcal{A}_a$ from $x^{k+h-h'}_w$ to $x^{k'}_{w'}$, but also from $x^{k+h-h'}_w$ to $x^{k'+h-h'}_{w'}$. This contradicts the determinism of G. So $|k'-k|$ is bounded by a constant depending on $\mathcal{A}_a$, and there is therefore a choice of v such that whenever $x^k_w \to x^{k'}_{w'}$ then $k' \in [k-1, k+1]$.

Suppose now that $y \in U$ and k is large. We use again the regularity of t on a sufficiently long run of $\mathcal{A}_a$: there is a $p_{w,y}$ such that for all $i \geq 0$ $x^{k+ip_{w,y}}_w \xrightarrow[G]{a} y$. By considering the least common multiple of all the $p_{w,y}$, we can adjust v so that $x^{k+1}_w \xrightarrow[G]{a} y$ and allow k to be simply greater than 1. □

The graph G being deterministic, the second part of this lemma can also be read backwards up to a good choice of v : if $x^{k+1}_w \xrightarrow[G]{a} x^{k'+1}_{w'}$, then $x^k_w \xrightarrow[G]{a} x^{k'}_{w'}$.

Let P be the sequence of vertices of the infinite path in G starting from r. Let $x^1_{p_1}$ be the first vertex of P in V_1. By the previous lemma, $x^2_{p_1}$ also belongs to P. Up to a good choice of u, v, the subsequence $(x^1_{p_1}, \ldots, x^2_{p_1})$ is all in $V_G \setminus V_{\leq 0}$. Let S^1 be this subsequence without the last $x^2_{p_1}$.

Let $G_{x^k_w}$ be the graph restricted to $(V_G \setminus P) \cup \{x^k_w\}$. For any $x^k_w \in V_k$, we note $\mathbf{Fr}(x^k_w) = \mathbf{Fr}(\mathrm{Unf}(G_{x^k_w}, x^k_w))$, *i.e.* the finite frontier obtained without following the infinite path. This notion can be expressed recursively. Formally, let $X = \{x_w \mid x^0_w \in V_0\}$ be a new alphabet. By the previous lemma and the fact that $G_{\leq k}$ is an acyclic graph, there is a word s_w over $\Gamma \cup X$ such that for all $k \geq 1$,

$$\mathbf{Fr}(x^k_w) = s_w[x_{w'} := \mathbf{Fr}(x^{k-1}_{w'})].$$

This allows the definition of the morphic word. The alphabet is $X \cup \Gamma$, and τ and σ are idempotent on Γ.

$$\begin{array}{rrcl} & \tau(\Delta) & = & \Delta . \prod_{x \in S} x \\ & \sigma(\Delta) & = & \mathbf{Fr}(\mathrm{Unf}(G_{\leq 0}, r)) \\ \text{for any } w, & \tau(x_w) & = & s_w \\ & \sigma(x_w) & = & \mathbf{Fr}(x^1_w) \end{array}$$

5.2.4 Words in Graph_2 are morphic, by recursion schemes

We prove again Theorem 5.2.6, *i.e.* that words of Graph_2 are necessarily morphic words, this time using recursion schemes. A first attempt of this proof appears in [Lav05].

By Corollary 5.1.6, it is enough to prove that the frontiers of prefix binary combs generated by order-1 schemes are morphic words. At this order, safety is not a constraint. In order to prove the result, a given scheme will undergo a series of transformations through the following lemmas. Eventually, we show that it is enough to consider specific simple schemes with only 2 nonterminals.

Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_1$ be a scheme. We begin by cleaning the scheme of non-productive nonterminals, where non-terminal F is *productive* if $[\![Fx_1 \dots x_{\rho(F)}]\!] \neq \bot$. It is obvious that for any $\mathfrak{S} \in \mathcal{S}_1$ generating an infinite tree, there is $\mathfrak{S}' \in \mathcal{S}_1$ with only productive nonterminals, and generating the same tree.

A nonterminal $F \in N$ is *infinite* if it generates an infinite tree, i.e. for $x_1, \dots, x_{\rho(F)} \in X$, the tree $[\![Fx_1 \dots x_{\rho(F)}]\!]$ is infinite.

Lemma 5.2.9. *For any $\mathfrak{S} \in \mathcal{S}_1$ generating an infinite tree, there is $\mathfrak{S}' \in \mathcal{S}_1$ with only infinite nonterminals, and generating the same tree.*

Proof. Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_1$ with only productive nonterminals producing an infinite tree, so S is infinite. Let $F \in N$ be a finite nonterminal and $n = \rho(F)$. There is a finite term w on $\Sigma \cup X$ such that $Fx_1 \dots x_n \underset{\mathfrak{S}}{\Longrightarrow}^* w$ and $[\![Fx_1 \dots x_n]\!] = w$. For each other nonterminal F', if there is an occurrence of F in $w_{F'}$, then in particular there is an occurence of $Ft_1 \dots t_n$ without any F in any t_i, for $1 \leq i \leq n$. Replacing $Ft_1 \dots t_n$ by $w[x_i := t_i]$ is done by rewriting. Eventually, we can replace all occurrences of F in $w_{F'}$.

The resulting scheme generates the same tree. Since N is finite, the whole process may be repeated for each finite nonterminal. □

The operand index i of a nonterminal F is *useful* if x_i appears in $[\![Fx_1 \dots x_{\rho(F)}]\!]$. A *useful nonterminal* has only useful arguments.

Lemma 5.2.10. *For any scheme in $\mathcal{S}_1$ generating an infinite tree, there is a scheme in $\mathcal{S}_1$ with infinite and useful nonterminals generating the same tree.*

Proof. Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_1$ generating a comb with only infinite nonterminals. Let F be such a nonterminal. If F is inaccessible from S (i.e. for any $S \underset{\mathfrak{S}}{\Longrightarrow} w$, F does not appear in w), we safely remove this nonterminal.

Suppose that i is a useless argument of F, then x_i cannot appear in $\pi(w)$, where w is any rewriting of a tree containing x_i only in an occurrence of $Fx_1 \dots x_{\rho(F)}$. We simply erase this argument. Formally, let $\diamondsuit$ be a new letter, $\mathfrak{S}' = (\Sigma \cup \{\diamondsuit\}, \{F'\} \cup N \setminus \{F\}, S, E')$ be the same scheme where $\rho(F') = n - 1$ and F' has production rule $F'x_1 \dots x_{i-1}x_{i+1} \dots x_n \Rightarrow w_F[x_i := \diamondsuit]$, and all occurrences of $Ft_1 \dots t_n$ in each production rule are replaced by $F't_1 \dots t_{i-1}t_{i+1} \dots t_n$. In the limit tree, this transformation can only change some subtrees into leaves labeled by $\diamondsuit$. But if it did, this means that there was a rewriting $S \underset{\mathfrak{S}'}{\Longrightarrow} w$ such that $\diamondsuit$ appears in $\pi(w)$. By the definition of F', this is contradictory with the lemma that i is a useless argument of F. So the resulting tree is the same. This operation only remove arguments, so iteration is finite. □

A scheme $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_1$ is called *simple recursive* if $N = \{S, F\}$, and $E = \{S \Rightarrow w_S, Fx_1 \dots x_{\rho(F)} \Rightarrow w_F\}$ where S cannot appear in w_S, w_F and F appears exactly once in both.

Lemma 5.2.11. *For any scheme in $\mathcal{S}_1$ generating a comb, there is a simple recursive scheme in $\mathcal{S}_1$ with only infinite and useful nonterminals generating the same comb.*

Proof. Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_1$ and $S \Rightarrow w_S \in E$. If $N = \{S\}$, the recursive form is easy to get : due to the fact that there is only one infinite branch in the limit tree and since S is infinite, there is exactly one occurence of S in w_S. Otherwise, since nonterminals are infinite, if there is two nonterminals without nonterminal ancestors in w_S, there would be two infinite paths in $[\![\mathfrak{S}]\!]$. So there only one such nonterminal subterm $Ft_1 \dots t_{\rho(F)}$, which is infinite and useful. There is a w such that

- $Fx_1 \dots x_{\rho(F)} \underset{\mathfrak{S}}{\Longrightarrow}^* w$,
- there is a nonterminal in w,
- and there is an occurence of each x_i to the left of this nonterminal.

In particular, $Ft_1 \dots t_{\rho(F)} \underset{\mathfrak{S}}{\Longrightarrow}^* w[x_i := t_i]$. This means there is no nonterminal in any t_i, or there would be two nonterminal without nonterminal ancestors in w_S. This means that in fact there is exactly one nonterminal in w_S. We reach the same conclusion for the right side of each production rule. This means there is only one w_i such that $S \underset{\mathfrak{S}}{\Longrightarrow}^i w_i$ and that w_i contains exactly one nonterminal F_i. Since there are finitely many nonterminals, the sequence $(F_i)_{i>0}$ is ultimately periodic and the simple recursive form is easy to obtain. □

We are now ready to focus on the proper result, *i.e.* Theorem 5.2.6.

Proof. Let $\mathfrak{S} = (\Sigma, \{S, F\}, S, E)$ be a simple recursive scheme of $\mathcal{S}_1$, $n = \rho(f)$ with production rules $S \Rightarrow w_S$, $Fx_1 \dots x_n \Rightarrow w_F$. Let Σ_0 be the subset of Σ of type $\mathbf{o}$. Up to renaming, suppose that $\bot \notin \Sigma$. Let $Fs_1 \dots s_n$ and $Ft_1 \dots t_n$ be the subterms starting with F respectively in w_S and w_F. Each s_i is a term on Σ and each t_i is a term on $\Sigma \cup X$. Let u_S, u_F be finite words on $\Sigma_0 \cup X$ such that $\mathbf{Fr}(\pi(w_S^{\bot})) = u_S\bot$, $\mathbf{Fr}(\pi(w_F^{\bot})) = u_F\bot$.

We set $\Sigma_0 \cup \{\Delta, x_1, \dots, x_n\}$ as the alphabet of the morphic word and τ, σ two morphisms on this alphabet. Letters from $\{\Delta, x_1, \dots, x_n\}$ are temporary and are erased by σ.

$$\begin{array}{rrcl} & \tau(\Delta) & = & \Delta \cdot u_F \\ & \sigma(\Delta) & = & u_S \\ \text{for } 1 \leq k \leq n, & \tau(x_k) & = & \mathbf{Fr}(\pi(t_k)) \\ & \sigma(x_k) & = & \mathbf{Fr}(\pi(s_k)) \\ \text{for any } a \in \Sigma_0, & \tau(a) & = & a \\ & \sigma(a) & = & a \end{array}$$

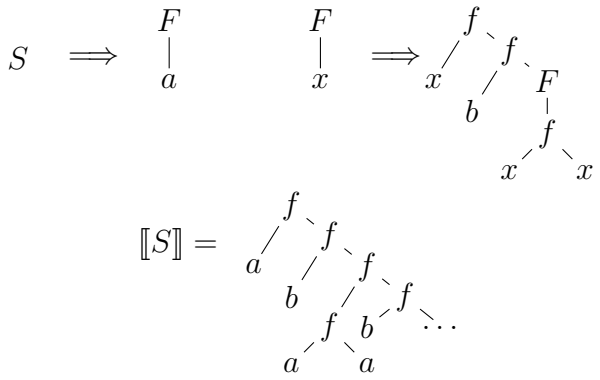


Figure 5.6: Rules of a scheme which frontier is a morphic word.

We show that the morphic word $\sigma(\tau^\omega(\Delta))$ is indeed $\mathbf{Fr}(\mathfrak{S})$. The scheme is simple recursive, which means that for each i, there is a unique term w_i such that $Fx_1 \dots x_n \underset{\mathfrak{S}}{\Longrightarrow}^i w_i$.

$$\begin{aligned}
\mathbf{Fr}(\pi(w_0^\perp)) &= \perp \\
\mathbf{Fr}(\pi(w_1^\perp)) &= u_F \cdot \perp \\
\mathbf{Fr}(\pi(w_2^\perp)) &= u_F \cdot u_F[\forall k \leq n, x_k := \mathbf{Fr}(\pi(t_k))] \cdot \perp \\
\text{for } i > 2, \mathbf{Fr}(\pi(w_i^\perp)) &= u_F \cdot \mathbf{Fr}(\pi(w_{i-1}))[\forall k \leq n, x_k := \mathbf{Fr}(\pi(t_k))] \cdot \perp
\end{aligned}$$

So for each i, $\tau^{(i)}(\Delta).\perp = \Delta.\mathbf{Fr}(\pi(w_i^\perp))$. Moreover, replacing variables by terminal terms can permuted with approximation :

$$\mathbf{Fr}(\pi(w_i[\forall k \leq n, x_k := s_k]^\perp)) = \mathbf{Fr}(\pi(w_i^\perp))[\forall k \leq n, x_k := \mathbf{Fr}(\pi(s_k)]$$

so the frontier of the $(i+1)^{th}$ iteration of S is

$$u_S \cdot \mathbf{Fr}(w_i)[\forall k \leq n, x_k := \mathbf{Fr}(\pi(s_k))] = \sigma(\tau^{(i)}(\Delta)) \cdot \perp.$$

At the limit, since $\tau^\omega(\Delta)$ is infinite, it is exactly $\mathbf{Fr}(\llbracket \mathfrak{S} \rrbracket)$. □

Example 5.2.12. Consider the scheme $(\{f, a, b\}, \{F, S\}, S, E)$ where E is given in Figure 5.6. The frontier of its limit tree is the morphic word $abaab \dots a^{2^i} b \dots$ already seen in Example 5.2.5. This scheme is simple recursive and $u_S = \varepsilon$, $u_F = xb$. We may hence deduce the following τ and σ.

$$\begin{array}{lllllll}
\tau(\Delta) & = & \Delta xb & \tau(x) & = & xx & \tau(a) = \sigma(a) = a \\
\sigma(\Delta) & = & \varepsilon & \sigma(x) & = & a & \tau(b) = \sigma(b) = b
\end{array}$$

▲

5.3 Second order

It is tempting to try and improve the method used in Section 5.2.4 in order to define an extension of the morphic words which would match exactly the words of any level of the hierarchy. In reality, higher-order schemes are not easily manipulated; in the rest of this chapter, we will only consider Graph_3. For instance, we would like to express words like the Champernowne word described below, or the Liouville word of Section 5.3.3. These two examples both express some increase of complexity compared to morphic words, respectively in terms of subword complexity and growth.

The *Champernowne word* [Cha33] (or *constant*) is the concatenation of numbers starting from 0 in some k-ary notation. Respectively in decimal and binary, it is

$$0123456789101112131415\ldots$$
$$0110111001011101111000\ldots$$

For any base, these words belong to Graph_3. To prove this fact, it is enough to find a second-order safe scheme yielding it. We present the binary case; other bases are similar. The scheme is $\mathfrak{S} = (\{f, 0, 1\}, \{S, F, G\}, S, E)$ where E is presented in Figure 5.7.

Even if there is more than two nonterminals here, this presentation is very similar to the form of schemes corresponding to (order-1) morphic words : there is one starting nonterminal S and only one "recursive" nonterminal F. We define formally this linearization form for order-2 schemes in Subsection 5.3.2.

5.3.1 Second-order morphic words

We introduce the notion of a *second-order morphic word*, or *2-morphic word*. Its definition mimics the notion of schemes : instead of letters, we use functions with operands.

Let $\rho_{max} \geq 0$ and $\Sigma = \biguplus_{i=0}^{\rho_{\max}} \Sigma_i$ where Σ_i is a set of function symbols of arity i. The symbols of Σ_0 are called letters, and there must be at least one letter. On the opposite, $\Sigma \setminus \Sigma_0$ is noted $\Sigma_{>0}$.

A *term word* θ is defined by

$$\theta := \varepsilon \mid a \in \Sigma_0 \mid f(\underbrace{\theta, \ldots, \theta}_{i}), f \in \Sigma_i \mid \theta \cdot \theta.$$

The set of term words on Σ is noted $\mathsf{TW}(\Sigma)$. We use the standard notation $f(\bar{z})$ for an arbitrary long $f(z_1, \ldots, z_n)$. Let $\mathcal{V}$ be a new set of letters called variables. We note $\Sigma_i(\bar{z})$ the set $\{f(\bar{z}) \mid f \in \Sigma_i, \bar{z} \in \mathcal{V}^i\}$ and $\Sigma(\bar{z}) = \biguplus_{i=0}^{\rho_{\max}} \Sigma_i(\bar{z})$.

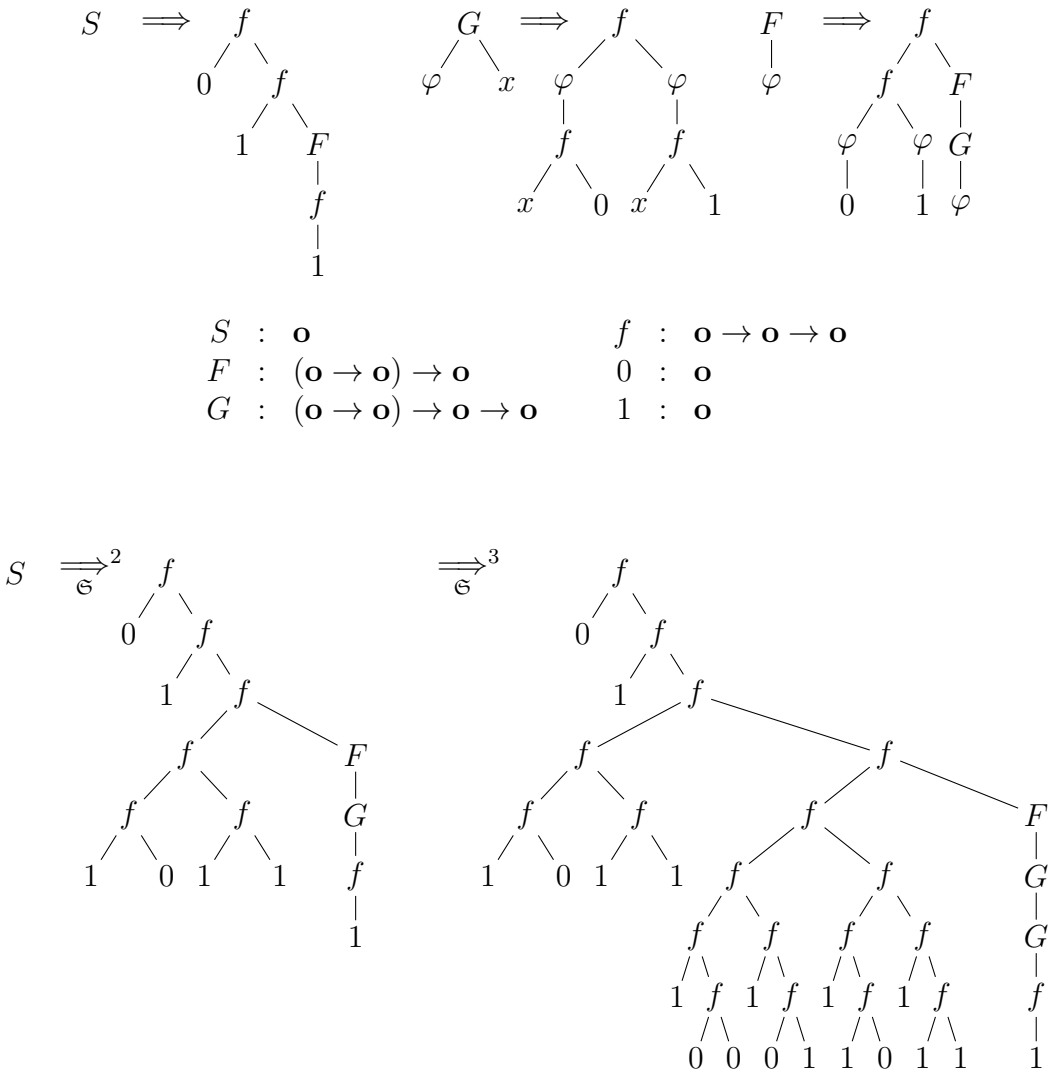


Figure 5.7: Order-2 safe scheme which frontier is the Champernowne word.

Let τ, σ be two morphisms on $\Sigma(\bar{z})^*$ w.r.t. concatenation.

$$\begin{array}{rcl} \text{for } a \in \Sigma_0, & & \\ \tau(a) & \in & \mathsf{TW}(\Sigma) \\ \sigma(a) & \in & \mathsf{TW}(\Sigma_0) \\ \text{for } f \in \Sigma_{>0} \text{ and } z_1, \ldots, z_n \in \mathcal{V}, & & \\ \tau(f(z_1, \ldots, z_n)) & \in & \mathsf{TW}(\Sigma \cup \mathcal{V}) \\ \sigma(f(z_1, \ldots, z_n)) & \in & \mathsf{TW}(\Sigma_0 \cup \mathcal{V}) \end{array}$$

This definition is extended on term words by

$$\begin{array}{rcl} \text{for } f \in \Sigma_{>0} \text{ and } t_1, \ldots, t_n \in \mathsf{TW}(\Sigma), & & \\ \tau(f(t_1, \ldots, t_n)) & = & \tau(f(z_1, \ldots, z_n))[\forall i, z_i := \tau(t_i)] \\ \sigma(f(t_1, \ldots, t_n)) & = & \sigma(f(z_1, \ldots, z_n))[\forall i, z_i := \sigma(t_i)] \end{array}$$

A *2-morphic-word* is any word of the form $\sigma(\tau^\omega(\Delta))$, where Δ is a letter in Σ such that $\tau(\Delta) \in \Delta \cdot \mathsf{TW}(\Sigma)$.

Example 5.3.1. The Champernowne word has a 2-morphic word presentation. Here $\Sigma_0 = \{0, 1\}$ and $\Sigma_1 = \{g\}$.

$$\begin{array}{rcl} \tau(\Delta) & = & \Delta g(0) g(1) \\ \tau(g(z)) & = & g(z0) g(z1) \\ \sigma(\Delta) & = & 01 \\ \sigma(g(z)) & = & 1z \end{array}$$

Implicitly, $\tau(1) = \sigma(1) = 1$ and $\tau(0) = \sigma(0) = 0$. The first steps of rewriting are shown.

$$\begin{array}{rcccccccc} \tau(\Delta) & = & \Delta & g(0) & g(1) & & & & \\ \tau^{(2)}(\Delta) & = & \Delta & g(0) & g(1) & g(00) & g(01) & g(10) & g(11) \\ \sigma(\tau^{(2)}(\Delta)) & = & 01 & 10 & 11 & 100 & 101 & 110 & 111 \end{array}$$

▲

Example 5.3.2. Just like the word $abaab \ldots a^{2^n} b \ldots$ was shown to be morphic, it is possible to build the 2-morphic word $abaaaab \ldots a^{2^{2^n}} b \ldots$. Letters a and b are copied by τ and σ.

$$\begin{array}{rcl} \tau(\Delta) & = & \Delta r(a) b \\ \tau(r(z)) & = & r(r(z)) \\ \sigma(\Delta) & = & \varepsilon \\ \sigma(r(z)) & = & zz \end{array}$$

It easy to prove that $\tau^{(n)}(r(a)) = r^{(2^n)}(a)$ and finally $\sigma(\tau^{(n)}(r(a))) = a^{2^{2^n}}$. ▲

If $\rho_{max} = 0$, all symbols are letters and we land back on the classic definition of morphic words. The converse is even more noteworthy : in our examples, the letters are

simply copied and symbols which are actually used are of non-zero arity, except for Δ. This hints the following proposition, which seems to be anecdotic but reveals itself useful in Section 5.3.2.

Proposition 5.3.3. *For a 2-morphic word w, there is $\tau, \sigma, \Sigma, \Delta$ such that*

- $w = \sigma(\tau^\omega(\Delta))$,
- Δ *appears only as the first letter of* $\tau(\Delta)$,
- *for any other letter* $a \in \Sigma_0$, $\tau(a) = a$.

Proof. Let $\tau, \sigma, \Sigma, \Delta$ such that $w = \sigma(\tau^\omega(\Delta))$. If $\tau(\Delta) = \Delta.u$, let Δ' be a new letter such that $\tau(\Delta') = \Delta'.u$ and $\sigma(\Delta') = \sigma(\Delta)$. This answers the constraint on Δ.

For any other letter, it is possible to add "fake operands". Formally, for any letter a, take a fresh symbol a_1 of arity 1 such that $\tau(a_1(z)) = \tau(a)$, $\sigma(a_1(z)) = \sigma(a)$. Then set $\tau'(a) = a$, and for each other symbol b, set

$$\tau'(b(\bar{z})) = \tau(b(\bar{z}))[a := a_1(\varepsilon)].$$

Then $w = \sigma(\tau'^\omega(\Delta'))$. □

5.3.2 Second-order scheme ω-frontiers

We adapt the method of Section 5.2.4 to prove that frontiers of combs generated by order-2 schemes are exactly the 2-morphic words. The main problem is to perform the transformation leading to a "linearized" scheme, where there are only two infinite nonterminals. Once this transformation is done, we may read the values of σ and τ as in the case of 1-order schemes.

A nice property of this method is that is does not use the safety constraint, even though 2-morphic words can be encoded as frontier of safe schemes. This means the ω-frontiers of schemes does not depend on safety.

The previous remark about productive nonterminals is still straighforward. For any $\mathfrak{S} \in \mathcal{S}_2$ generating an infinite tree, there is $\mathfrak{S}' \in \mathcal{S}_2$ with only productive nonterminals, and generating the same tree. From now on, we suppose every nonterminal is productive. The second step, deleting useless operands, is more delicate. In this case, simply deleting operands will not work, because it would lead to type mismatches — see below for an example. The solution kept here is to duplicate nonterminals when needed.

Lemma 5.3.4. *For any scheme in $\mathcal{S}_2$ generating a tree, there is a scheme in $\mathcal{S}_2$ with only useful nonterminals generating the same tree.*

Proof. Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_2$ generating an infinite tree. Let F be a nonterminal. If F is inaccessible from S (i.e. for any $S \underset{\mathfrak{S}}{\Longrightarrow}^* w$, F does not appear in w), we may safely remove it.

Let $n = \rho(F)$ and $F : \tau_1 \to \ldots \to \tau_n \to \mathbf{o}$. Suppose that i is a useless operand of F, then x_i cannot appear in w, where w is any rewriting of a tree containing x_i only in an occurrence of $Fx_1 \ldots x_{\rho(F)}$. As in the $\mathcal{S}_1$ case, we can erase this operand, but at the cost of duplication of other nonterminals.

Formally, let $\diamondsuit : \tau_i$ be a new letter, and $F' : \tau_1 \to \ldots \tau_{i-1} \to \tau_{i+1} \to \ldots \to \tau_n \to \mathbf{o}$ a new nonterminal of the same type than F without the i-operand. Let $w_{F'} = w_F[x_i := \diamondsuit]$. There are two disjoint cases depending on the order of x_i.

- If $\ell(x_i) = 1$, this case is similar to the first order. Since the scheme has order 2, each app. subterm of root F has an order at most 1; this means that this app. subterm appears as $Ft_1 \ldots t_k$ with $k \geq i$. So each occurence of F can be replaced by F' where this i-operand is deleted.

- If $\ell(x_i) = 0$, then there may be occurences of F without its i-operand in production rules of other nonterminals. In this case the app. subterm is an argument of another nonterminal H. For each nonterminal $H : (\mathbf{o}^{h_1} \to \mathbf{o}) \to \ldots \to \mathbf{o}$, and each sequence $s \in [0, h_1]^{\rho(H)}$, define the duplicate H_s. Let $(x_i)_{1 \leq x \leq \rho(H)}$ be a family of variables where

  - if $s_i = 0$, then $x_i : \mathbf{o}^{h_i} \to \mathbf{o}$;
  - otherwise, $x_i : \mathbf{o}^{h_i - 1} \to \mathbf{o}$.

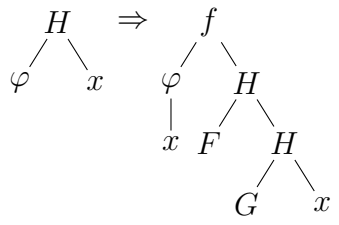
  Then let $H_s x_1 \ldots x_{\rho(H)} \Rightarrow w_{H_s}$ be a copy of w_H where types are adjusted accordingly. Formally, if $s_i \neq 0$, then each occurrence of x_i is modified. If its app. subterm has at least s_i operands, the s_i-operand is erased. Otherwise, it has order 1, so it is the j-operand of an app. subterm of root K, which is changed into the correct duplicate $K_{s'}$ where $s'_j = s_i - \rho(x_j)$.

  Each time F appears without its i-operand, it has order 1 and must appear as operand of some nonterminal H. So we can change F into F' and H into the appropriate H_s.

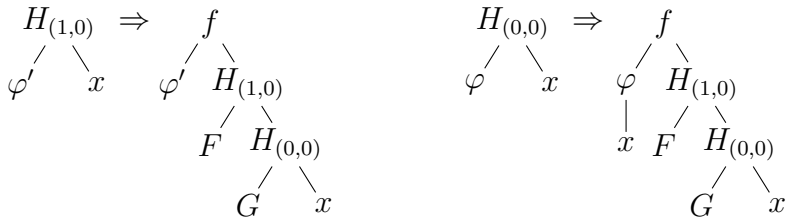
Example 5.3.5. For instance, let

$$\begin{array}{lcllcl} F & : & \mathbf{o} \to \mathbf{o} & \varphi & : & \mathbf{o} \to \mathbf{o} \\ G & : & \mathbf{o} \to \mathbf{o} & x & : & \mathbf{o} \\ H & : & (\mathbf{o} \to \mathbf{o}) \to \mathbf{o} \to \mathbf{o} & f & : & \mathbf{o} \to \mathbf{o} \to \mathbf{o} \end{array}$$

Suppose the rewriting rule for H is



Suppose that the argument of F is revealed useless and has to be deleted. Since G does not change, this implies a duplication of the nonterminal H. Note that in the rule for H_1, the type of φ' is now $\mathbf{o}$.



▲

As before, in the limit tree, this transformation can only change some subtrees into leaves labeled by $\diamondsuit$. But if it did, this means that there was a rewriting $S \underset{\mathfrak{S}'}{\Longrightarrow}^* w$ such that $\diamondsuit$ appears in w. By the definition of F', this is contradictory with the fact that i is a useless argument of F. So the resulting tree is the same.

We iterate this process for all useless operands of all nonterminals. The number of possible different duplicates of nonterminals is finite, so the whole process is also finite. □

Among other differences between $\mathcal{S}_2$ and $\mathcal{S}_1$, nonterminals generating finite trees cannot be avoided. Indeed, an applicative subterm which is not a term cannot be rewritten. To overcome this fact, it is enough to make a clear distinction between nonterminals generating finite or infinite trees.

A nonterminal G with production rule $Gx_1 \dots x_{\rho(G)} \Rightarrow w_G$ is called *semiterminal* if w_G does not contain any nonterminal or does contain only other semiterminals. As before, a nonterminal F is called *infinite* if $[\![Fx_1 \dots x_{\rho(F)}]\!]$ is an infinite tree. "Infinite nonterminal" is shortened in ∞-nonterminal.

The following property shows that these two categories (semi- and ∞-nonterminal) are actually a partition of nonterminals. Later on, we will focus only on ∞-nonterminals to build 2-morphic words.

A subterm is called *head applicative subterm* whenever its root is nonterminal and it has no other nonterminal above. Since the schemes of this section have order 2, any head applicative subterm has order 0.

Lemma 5.3.6. *Let $\mathfrak{S} = (\Sigma, N, S, E)$ be a scheme without nonproductive or useless nonterminals. A term on $\Sigma \cup N$ produces an infinite tree if and only if it contains a non-semiterminal.*

Proof. Let F be a non-semiterminal of arity n. Since F is productive, there is a w such that $Fx_1 \dots x_n \underset{\mathfrak{S}}{\Longrightarrow}^* w$ and the root of w is a terminal. Also, by definition, w contains at least one non-semiterminal nonterminal. Since the scheme has order 2, any head applicative subterm has order 0, so there is a head subterm. Since each nonterminal is useful, there is a sequence of rewriting bringing a head non-semiterminal. This nonterminal has all its operands and can be rewritten. By iterating the process, we build an infinite tree.

We prove now the opposite direction : a term only composed of terminals and semiterminals cannot be infinitely rewritten. We note sN the set of semiterminals. By definition, the graph over sN where $F \to G$ iff G is in w_F is a acyclic graph. We can provide a (topological) ordering on semiterminals so that $F \to G \iff F > G$. This ordering can be extended on $\Sigma \cup sN$ with $\Sigma < sN$. Then we translate it to $\mathsf{T}(sN \cup \Sigma)$ as follows.

For any app. subterm t, let $\mathrm{root}(t)$ be its root. We introduce the notion of lvl-1-branches of a nonterminal subterm t : a lvl-1-branch is a multiset $\{\mathrm{root}(t)\} \cup b$ where b is a lvl-1-branch of an app. subterm t_i of level 1. We call $\mathrm{lvl}^1(t)$ the multiset of lvl-1-branches of t.

For a term t, for any branch b of t we call $\mathrm{val}(b)$ the multiset of $\mathrm{lvl}^1(t')$ for any app. subterm intersecting this branch. Finally, we call $B(t)$ the multiset of $\mathrm{val}(b)$ for each branch of t. The values of app. subterms can be totally ordered by multiset ordering. Then $\{B(t) \mid t \in \mathcal{T}(sN \cup \Sigma)\}$ is therefore totally ordered by multiset ordering.

We have now to prove that (1) if $t \underset{\mathfrak{S}}{\Longrightarrow} t'$, then $B(t) > B(t')$; (2) the order on $\{B(t) \mid t \in \mathcal{T}(sN \cup \Sigma)\}$ is a well-ordering. The latter property comes from the well-known fact that the multiset operation preserves well-ordering. Yet $B(t)$ is just a chained encapsulation of 4 multisets on a finite ordering.

It remains to prove (1). Let $t_F = Ft_1 \dots t_n \underset{\mathfrak{S}}{\Longrightarrow} w = w_F[x_i := t_i]$ be the rewritten term in t with $n = \rho(F)$. Let b be a branch of t'. If b does not intersect w, then $\mathrm{val}(b)$ already exists in $B(t)$. Otherwise, b goes through w. By simply looking at w_F, we can say that there is h such that

$$b = u \cdot v_1 \cdot \mathrm{root}(t_{i_1}) \cdot v_2 \cdot \dots \cdot \mathrm{root}(t_{i_{h-1}}) \cdot v_h \cdot c_{i_h}$$

where u is the part above the rewritten subterm, $(v_k)_{k\in[1,h]}$ are (possibly empty) segments of branches in w_F without variables, and c_{i_h} is a (possibly empty) branch of t_{i_h}.

For any nonterminal G in a branch b, recall that we note t_G the associated subterm.

$$\begin{aligned}\mathrm{val}(b) = \{val(t_G) \mid\ & t_G \text{ intersects } u \\ & \text{or } t_G \text{ intersects } v_k, k \le h \\ & \text{or } t_G = t_{i_k} s_1 \dots s_l, k < h, l \le \rho(t_{i_k}) \\ & \text{or } t_G \text{ intersects } c_{i_h}\}\end{aligned}$$

Let b_i be a branch in t going through t_{i_h}, of the form $u \cdot F \cdot c$. This branch exists and $\mathrm{val}(b_i) \in B(t)$. We prove that $\mathrm{val}(b) < \mathrm{val}(b_i)$, which implies the required result $B(t) > B(t')$.

To this extent, we study the four cases above in order.

1. If t_G intersects u, since t_F and w are both of order 0, the rewriting does not affect $\mathrm{lvl}^1(t_G)$.

2. By definition, any nonterminal appearing in w_F is smaller than F. If t_G has its

root in a v_k, a lvl-1-branch of t_G is composed of at most one t_i, and nonterminals smaller than F. So it is smaller (for multiset ordering) than the corresponding branch number i in t_F. So $\mathrm{lvl}^1(t_G) < \mathrm{lvl}^1(t_F)$.

3. When t_G is an applied t_{j_k}, since $\ell(t_{j_k}) = 1$, any lvl-1-branch can be extended in a lvl-1-branch of t_F. So $\mathrm{lvl}^1(t_G) < \mathrm{lvl}^1(t_F)$.

4. As in the case of u, for any subterm t_G of a t_i, $\mathrm{lvl}^1(t_G)$ is unchanged.

To summarize, between b_i and b, for all t_G of the beginning (in u) or end (in c_{i_h}), $\mathrm{lvl}^1(t_G)$ is copied. Moreover $\mathrm{lvl}^1(t_F)$ disappears, and new values than may appear are necessarily smaller than $\mathrm{lvl}^1(t_F)$. So $\mathrm{val}(b_i) > \mathrm{val}(b)$. □

The definition of *simple recursive* still holds for order-2 schemes. A scheme $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_2$ is called simple recursive if it has no nonproductive or useless arguments, and there are only two ∞-nonterminals S, F, and $\{S \Rightarrow w_S, Fx_1 \dots x_{\rho(F)} \Rightarrow w_F\} \in E$ where S cannot appear in w_S, w_F and F can appear at most once.

Lemma 5.3.7. *For any scheme in $\mathcal{S}_2$ generating a comb, there is a simple recursive scheme in $\mathcal{S}_2$ with only useful nonterminals generating the same comb.*

Proof. Let $\mathfrak{S} = (\Sigma, N, S, E) \in \mathcal{S}_2$ be a useful scheme producing a comb and let $S \Rightarrow w_S \in E$. If there is no ∞-nonterminal in w_S, then $[\![\mathfrak{S}]\!] = [\![w_S]\!]$ is finite by Lemma 5.3.6. So there is an occurence of an ∞-nonterminal in w_S. Since all nonterminals are useful, we can suppose that there is an occurence of a head subterm with ∞-nonterminal root F, modulo some rewriting. There is only one such head subterm in a given rewriting of S, otherwise the limit tree would have two infinite branches. Let $n = \rho(F)$ and $Ft_1 \dots t_n$ be this subterm.

In order to get a contradiction, suppose that there is an ∞-nonterminal G in t_i for some i. We note $\tilde{t_i}$ as a copy of t_i where an app. subterm t_G of root G is replaced by a variable x (of the same type). Since operands are always useful, there is a term w such that $Ft_1 \dots \tilde{t_i} \dots t_n \underset{\mathfrak{S}}{\Longrightarrow}^* w$ where x occurs outside of any nonterminal (even semiterminals). Moreover, since F is infinite, there exists an ∞-nonterminal H in w. In particular, $Ft_1 \dots t_n \underset{\mathfrak{S}}{\Longrightarrow}^* w[x := t_G]$.

If $x : \mathbf{o}$, we fall on the same case than in $\mathcal{S}_1$; there would be more than one infinite branch. So the only possibility is that $\ell(t_G) = \ell(x) = 1$ and there is an occurence of x in w containing an occurence of H. Namely, x occurs in w in the form $(xu_1 \dots u_{\rho(x)})$ with $H \in u_k$ for some k. For this to happen, we must have $\ell(t_i) > 0$ so that t_i can feed x.

By applying the exact same process to G, we find that $\ell(u_k) > 0$ so $\ell(x) > 1$, which is impossible because then all app. subterms have order at most 1 in a order-2 scheme. So there cannot be any ∞-nonterminal in any t_i. As in the order-1 case, the sequence of ∞-nonterminals encountered by rewriting S is ultimately periodic. We may therefore select S and another ∞-nonterminal to get the required form. □

We are ready to prove the main result.

Theorem 5.3.8. *The following sets of ω-words are equal :*

1. *the ω-words of the third level of the pushdown hierarchy,*
2. *the ω-frontiers of safe order-2 schemes,*
3. *the frontiers of combs generated by order-2 schemes,*
4. *the 2-morphic words.*

Proof. The equality $1 = 2$ comes from Theorem 4.2.6, and $2 \subseteq 3$ comes from Proposition 5.0.5. It is therefore enough to prove that (a) frontiers of combs generated by arbitrary schemes are 2-morphic words, (b) 2-morphic words are ω-frontiers of safe schemes. For (b) it is easier to prove that 2-morphic words are frontiers of safe combs.
(a) Let $\mathfrak{S}$ be a simple recursive scheme producing a comb, and let S, F be the two ∞-nonterminals, with production rule $S \Rightarrow w_S$, $Fx_1 \dots x_{\rho(F)} \Rightarrow w_F$. For each variable $x_i : \mathbf{o}^n \to \mathbf{o}$, we define the symbol $x_i \in \Sigma_n$. Let t_i^S, t_i^F be the i-operands of F respectively in w_S and w_F. They are in $\mathsf{AT}(sN \cup \Sigma)$, and have type $\mathbf{o}^n \to \mathbf{o}$. By Lemma 5.3.6, when fed with appropriate variables, $[\![t_i^S \bar{z}]\!]$ and $[\![t_i^F \bar{z}]\!]$ are finite trees.

We simply set $\sigma(x_i(\bar{z})) = \mathbf{Fr}([\![t_i^S \bar{z}]\!])$. For τ we have to take other variables in consideration, but we will flatten the terminals. Define the mapping ξ from $\mathsf{T}(\Sigma \cup \bar{z})$ to $\mathsf{TW}(\Sigma)$:

- if t is a leaf, $\xi(t) = t$;
- if $t = ft_1 \dots t_{\rho(f)}$ where f is a terminal, $\xi(t) = \prod_{k=1}^{\rho(f)} \xi(t_k)$;
- if $t = xt_1 \dots t_{\rho(x)}$ where x is a variable, $\xi(t) = x(\xi(t_1), \dots, \xi(t_{\rho(x)}))$.

We set $\tau(x_i(\bar{z})) = \xi(\hat{t}_i^S)$ where $\hat{t}_i^S$ is the normal form of $t_i^S \bar{z}$, *i.e.* such that $t_i^S \bar{z} \underset{\mathfrak{S}}{\Longrightarrow}^* \hat{t}_i^S$ which cannot be rewritten.

The starting letter Δ is naturally associated to the part "outside of F". Formally, we can suppose as before that the app. subterm of root F in w_S is in fact a head subterm (in w_F as well). So if we replace this subterm by $\perp$ in order to approximate, we get w'_S and w'_F with only terminals and semiterminals. We set $\sigma(\Delta) \cdot \perp = \mathbf{Fr}([\![w'_S]\!])$. For τ, we set $\xi(\perp) = \varepsilon$ and we have then $\tau(\Delta) = \Delta \cdot \xi(\hat{w}'_F)$ where $\hat{w}'_F$ is the normal form of w'_F.

By construction, if $S \Rightarrow^k s_k$ only by rewriting F, then

$$\begin{aligned}
\mathbf{Fr}([\![s_k^\perp]\!]) &= \mathbf{Fr}([\![w'_S]\!]) \cdot \mathbf{Fr}([\![w'_F[x_i := t_i^S]]\!]) \cdots \mathbf{Fr}([\![w'_F \underbrace{[x_i := t_i^F] \dots}_{k-1} [x_i := t_i^S]]\!]) \\
&= \sigma(\Delta) \cdot \sigma(\xi(\hat{w}'_F)) \cdots \sigma(\xi(\hat{w}'_F[x_i(\bar{z}) := \tau(x_i(\bar{z}))])) \\
&= \sigma(\tau^{(k)}(\Delta)).
\end{aligned}$$

(b) Let w be a 2-morphic word defined by τ, σ on Σ. We chose a presentation given by Proposition 5.3.3 in order to obtain a safe scheme.

Mirroring the previous case, we set a nonterminal F which operands in the production rule are the symbols of non-zero arity. Formally, the set of variables is exactly $\Sigma_{>0}$; they have all type order 1 and same arity (for a symbol $a \in \Sigma_n$, we have the variable $a : \mathbf{o}^n \to \mathbf{o}$). The set of terminals is $\Sigma_0 \cup \{f\}$ where $f : \mathbf{o} \to \mathbf{o} \to \mathbf{o}$.

Let $a \in \Sigma_n$ such that $\tau(a(\bar{z})) = \theta \in \mathsf{TW}(\Sigma \cup \bar{z})$. Suppose a is fixed as the i-operand of F. We define the mapping $\mu : \mathsf{TW}(\Sigma \cup \bar{z}) \mapsto \mathsf{AT}(\Sigma_0 \cup \{f\} \cup \bar{z})$ as a "converse of ξ". It uses a set of semiterminals which types are as follows; the types of app. subterms of the i-operand of F are always $\mathbf{o}^n \to \mathbf{o}$.

(concatenation)	C^n	:	$(\mathbf{o}^n \to \mathbf{o}) \to (\mathbf{o}^n \to \mathbf{o}) \to \mathbf{o}^n \to \mathbf{o}$
(projection)	P_i^n	:	$\mathbf{o}^n \to \mathbf{o}$
(symbol of non-zero arity)	G_b^n	:	$(\mathbf{o}^{\rho(b)} \to \mathbf{o}) \to (\mathbf{o}^n \to \mathbf{o})^{\rho(b)} \to \mathbf{o}^n \to \mathbf{o}$
(symbol of arity 0)	G_b^n	:	$\mathbf{o}^n \to \mathbf{o}$

- if $\theta = \theta_1 \cdot \theta_2$, then $\mu(\theta) = C^n(\mu(\theta_1), \mu(\theta_2))$ where

$$C(\varphi_1, \varphi_2, \bar{z}) \Rightarrow f(\varphi_1 \bar{z}, \varphi_2 \bar{z}).$$

- if $\theta = z_i$, then $\mu(\theta) = P_i^n$ with $P_i^n(\bar{z}) \Rightarrow z_i$.
- if $\theta = b(\theta_1, \ldots, \theta_i)$ with $b \in \Sigma_{>0}$, then $\mu(\theta) = G_b^n(b, \mu(\theta_1), \ldots, \mu(\theta_{\rho(b)}))$ with

$$G_b^n(\psi, \varphi_1, \ldots, \varphi_{\rho(b)}, \bar{z}) \Rightarrow \psi(\varphi_1 \bar{z}, \ldots, \varphi_{\rho(b)} \bar{z}).$$

- if $\theta = b \in \Sigma_0$, then $\mu(\theta) = G_b^n$ with $G_b^n(\bar{z}) \Rightarrow b$.

Let θ_Δ be the term word such that $\tau(\Delta) = \Delta \cdot \theta_\Delta$. Let t_τ be the normal form of $\mu(\theta_\Delta)(a_1 \ldots a_{\rho(F)})$, *i.e.* a term on $\Sigma \cup \{f\}$ such that $\xi(t_\tau) = \theta_\Delta$. The rule for F is

$$F a_1 \ldots a_{\rho(F)} \Rightarrow f(t_\tau, F(\mu(\tau(a_1)), \ldots, \mu(\tau(a_{\rho(F)})))).$$

In the same way, the starting nonterminal S has the rule

- If $\sigma(\Delta) = \varepsilon$, then
$$S \Rightarrow F(\mu(\sigma(a_1)), \ldots, \mu(\sigma(a_{\rho(F)})))$$
- otherwise,
$$S \Rightarrow f(t_\sigma, F(\mu(\sigma(a_1)), \ldots, \mu(\sigma(a_{\rho(F)}))))$$
where t_σ is a term on $\Sigma_0 \cup \{f\}$ such that $\mathbf{Fr}(t_\sigma) = \sigma(\Delta)$.

Let $\mathfrak{S}$ be this defined scheme. Note that $\mathfrak{S}$ is not presented in a cleaned version : most semiterminals have useless arguments. This is not a requirement in this direction.

The important property is that it is safe : indeed, app. subterms of type $\mathbf{o}^k \to \mathbf{o}$ are all in w_F below F, and their sons have the same type.

It is easy to see that for $a \in \Sigma_{>0}$, if $\hat{\mu}(\theta)(\bar{z})$ is the normal form of $\mu(\theta)(\bar{z})$,

$$\begin{aligned} \xi(\hat{\mu}(\tau(a(\bar{z})))(\bar{z})) &= \tau(a(\bar{z})), \\ \xi(\hat{\mu}(\sigma(a(\bar{z})))(\bar{z})) &= \sigma(a(\bar{z})). \end{aligned}$$

By applying the method of part (a), we check that the frontier of the limit tree of $\mathfrak{S}$ is indeed $\sigma(\tau(\Delta))$. □

The above properties may sound natural, but they do not work out-of-the-box on further levels. First, at order 3 there are variables of order 2, so head nonterminal app. subterms are not necessarily terms. Consequently, a terminal term can contain nonterminals : the notion of ∞-nonterminal has to be redefined. Moreover, many proofs rely on the fact that the level of an app. subterm has order at most 1.

An immediate question about this result is whether we can transform an arbitrary scheme of ω-frontier w into a comb which frontier is w. Obviously the properties of the hierarchy developped in the previous sections are not available, so the question is open.

This result yields immediate properties on 2-morphic words.

Corollary 5.3.9. *Let w be a 2-morphic word on Σ.*

1. $\mathrm{MTh}(w)$ *is decidable.*
2. *For any* MSO*-transduction $\mathcal{T}$, if $\mathcal{T}(w)$ is an ω-word, then it is a 2-morphic word.*
3. *Let $a \in \Sigma_0$ and let a_n the index of the i^{th} occurence of a. There is a $C > 0$ such that for sufficiently large n, $a_n - a_{n-1} = \mathcal{O}(2^{2^{C.n}})$.*

Property 3 is the expected extension of [CT02, Prop. 14]; compare to the lower bound in Example 5.3.2.

Proof. Properties 1 and 2 are straightforward properties of graphs in the hierarchy. Property 3 is a corollary of Theorem 4.5.3 : there is a monadic interpretation transforming the ω-word into a comb where the infinite branch is composed of all vertices marked by a in order. The i-subtree is the rest of intermediate vertices. □

Remark 5.3.10. The subword complexity of an ω-word w is the function which maps n on the number of factors of length n in w. For morphic words, this complexity is in $O(n^2)$; see [AS03, Section 10.4] for more details. For 2-morphic words the complexity is maximal because of the Champernowne word. This was also noted for k-lexicographic words [Bár08]. ■

Remark 5.3.11. The morphism τ can be seen itself as a 1-order scheme. In this sense, it reminds of the transformation in [KNU01] using the operator @. The similarities end

here; whereas the role of @ was to study structural properties of the limit tree, the role of τ is simply to reproduce the mechanism of F and depends on the fact that there is only one recursive nonterminal. ■

5.3.3 Liouville word

The *Liouville word* or *constant* is another example of infinite words more complex than morphic words. The constant is

$$\sum_{k>0} 10^{-k!} = 0.11000100000000000000000010\ldots.$$

We only consider digits after the dot. It is the frontier of the limit tree of the safe order-2 scheme $(\{f,0,1\},\{S,F,G,H\},S,E)$ where E is described in Figure 5.8.

The associated 2-morphic word is defined by the the following morphisms on $\Sigma = \{0,1,\Delta,g,n\}$ where g,n have arity 1. As before, 0 and 1 are dumb letters : $\tau(1) = \sigma(1) = 1$ and $\tau(0) = \sigma(0) = 0$.

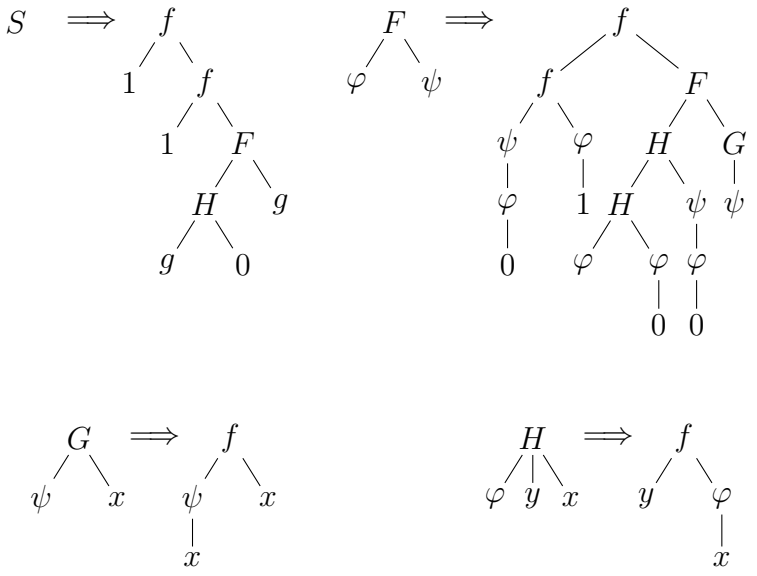
$$\begin{array}{rclcrcl}
\tau(\Delta) & = & \Delta n(g(0))g(1) & & \sigma(\Delta) & = & 11 \\
\tau(g(z)) & = & n(g(0))g(0)g(z) & & \sigma(g(z)) & = & 0z \\
\tau(n(z)) & = & zn(z) & & \sigma(n(z)) & = & z
\end{array}$$

Informally, the goal is to obtain at step k an additionnal number of letters equal to $(k+2)! - (k+1)! = (k+1)(k+1)!$. The symbols n,g are such that

$$\begin{array}{rcl}
|\sigma \circ \tau^{(k)}(n(z))| & = & k \\
|\sigma \circ \tau^{(k)}(g(z))| & = & (k+1)!
\end{array}$$

So $\sigma \circ \tau^{(k)}(n(g(z))g(z))$ is a word of length $k.(k+1)! + (k+1)! = (k+1)(k+1)!$. This is clearer when considering one iteration.

$$\begin{array}{rcccccccccc}
\tau^{(2)}(\Delta) & = & \Delta & n(g(0)) & g(1) & \tau(g(0)) & n(\tau(g(0))) & n(g(0)) & g(0) & g(1) \\
 & = & \Delta & n(g(0)) & g(1) & n(g(0))g(0)g(0) & n(n(g(0))g(0)g(0)) & n(g(0)) & g(0) & g(1) \\
\sigma(\tau^{(2)}(\Delta)) & = & 11 & 00 & 01 & 00\,00\,00 & 00\,00\,00 & 00 & 00 & 01
\end{array}$$



$$
\begin{array}{rcl}
F & : & (\mathbf{o}\to\mathbf{o})\to(\mathbf{o}\to\mathbf{o})\to\mathbf{o}\\
G & : & (\mathbf{o}\to\mathbf{o})\to\mathbf{o}\to\mathbf{o}\\
H & : & (\mathbf{o}\to\mathbf{o})\to\mathbf{o}\to\mathbf{o}\to\mathbf{o}
\end{array}
\qquad
\begin{array}{rcl}
f & : & \mathbf{o}\to\mathbf{o}\to\mathbf{o}\\
g,\varphi,\psi & : & \mathbf{o}\to\mathbf{o}\\
x,y,0,1 & : & \mathbf{o}
\end{array}
$$

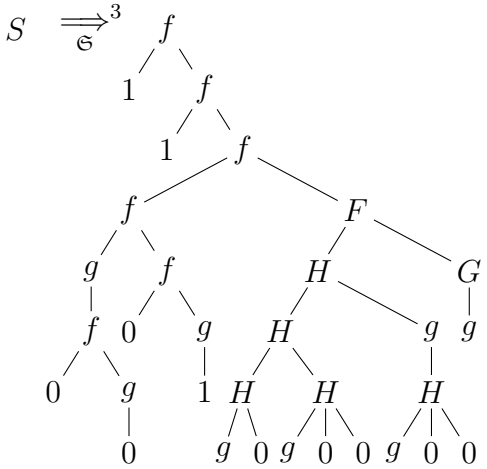


Figure 5.8: Order-2 safe scheme which frontier is the Liouville word.

List of notations

	PRELIMINARIES	
$\mathcal{P}(S)$	powerset of S	19
$[a, b]$	closed interval bounded by a and b	19
Σ^*	set of finite words on Σ	19
$<_{\text{lex}}$	lexicographic order	19
$\sqsubset, \sqsubseteq$	(strict) prefix relation	19
$\perp$	"not prefix" symmetric relation	19
$\bar{a}$	reverse arc label	34
$\langle\, U, (R_1, \ldots, R_k)\,\rangle$	logical structure	20
$\models$	satisfaction of a formula	30
$\simeq$	isomorphism relation	20
$\text{MTh}(G)$	monadic theory of the structure G	31
	ORDERS	
$\mathbf{0}, \mathbf{1}, \ldots, \mathbf{k}$	finite order types	23
ω, ζ, η	order types of $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$	23
L^*	reverse ordering of L	23
$\preceq$	suborder relation	23
$\omega \uparrow\uparrow n$	exponential tower of ω's of height n	25
ε_0	smallest ordinal such that $\varepsilon_0 = \omega^{\varepsilon_0}$	26
$\tilde{r}$	alternative version of the Hausdorff rank	27
	TREES	
$\mathbf{Fr}$	frontier	29
KB	Kleene-Brouwer ordering	29
$\equiv$	isomorphism up to subtree permutation	79
∂	Cantor-Bendixson derivative	81
$\tilde{r}_{\text{CB}}$	alternative version of the Cantor-Bendixson rank	82
$\mathcal{K}_f$	tree of the function f	62
	COVERING GRAPHS	
$\alpha[n]$	element of the fundamental sequence of α	55
$\lessdot$	covering graph relation	55
$\mathcal{G}_\alpha$	covering graph of α	56
$\mathfrak{d}$	degree word	57

Recursion schemes

$\mathbf{o}$	base type	92
ℓ	type order	93
$\mathsf{AT}(\Sigma), \mathsf{T}(\Sigma)$	set of (applicative) terms over Σ	93
ρ	arity	93
$Fx_1 \dots x_{\rho(F)} \Rightarrow w_F$	production rule	93
$\underset{\mathfrak{S}}{\Longrightarrow}$	rewriting relation	93
$t^\perp$	term approximation	93
$[\![t]\!]$	limit tree of the term t	94

Index

ω-tail, 54

ancestor, 22
approximation, 93
arc, 21
arithmetics, 23
    on ordinals, 24
arity, 20, 93

binary tree, 22
branch, 22
Büchi automaton, 32

Cantor normal form, 26
Champernowne word, 106
closed formula, 30
CNF, *see* Cantor normal form 26
coding, 98
cofinality, 54
color set, 21
colored
    deterministic tree, 22
    graph, 21
    ordering, 23
comb, 87
    #-comb, 87
    #-comb-graph, 87
complete tree, 22
configuration graph of a n-hopda, 50
crossing-free, 55

degree, 21
delabeled graph, 22
dense, 26
descendant, 22
deterministic, 21
deterministic tree, 22

finite presentation, 19
FO, 30
frontier, 29
ω-frontier, 91
full tree, 22
fundamental sequence, 55

graph, 21
Graph_n, 38

Hausdorff rank, 27
homogeneous, 92

infinite nonterminal, 103
interval, 23
inverse rational mapping, 34

Knuth notation, 25

label, 21
length, 19
linear ordering, 22
Liouville word, 117

morphic word, 98
    2-morphic word, 106
morphism, 97
MSO, 31
MSO-coloring, 33
MSO-compatible, 33
MSO-interpretation, 33
MSO-transduction, 35
MSO-transduction, 35

nonterminal, 93
  ∞-nonterminal, 111

operand, 93
order (schemes), 92
order type, 23
ordinal, 24

path, 22
powerset, 19
prefix, 19
prefix tree, 22
production, 93
productive nonterminal, 103
pushdown hierarchy, 37

RCNF, *see* Cantor normal form 26
recursion, *see* recursion
recursion scheme, 92
reverse ordering, 23
rewriting relation, 93
root, 22

safety, 94
scattered, 26
semiterminal, 111
signature, 20
simple, 22
structure, 20
subordering, 23
subterm, 93
  applicative, 93
    head, 111
successor, 24
support, 21

tame, 78
term, 93
  applicative, 93
  grammar, *see* recursion scheme 92
  tree, 96
  word, 106
terminal, 93
tree presentation, 22
Tree_n, 38
tree-walking automaton, *see* TWA 67
treegraph, 36
  Γ-treegraph, 46
TWA, 67
typed alphabet, 93
types, 92

unfolding, 35
unlabeled graph, 22
useful nonterminal, 103

well-ordered trees, 73
well-ordering, 24
word
  ω-word, 23
  finite, 19

Bibliography

[AdMO05] Klaus Aehlig, Jolie de Miranda, and Luke Ong. Safety is not a restriction at level 2 for string languages. In V. Sassone, editor, *Proc. of FoSSaCS*, volume 3441 of *Lecture Notes in Computer Science*, pages 490–504. Springer, 2005.

[AS03] Jean-Paul Allouche and Jeffrey Shallit. *Automatic Sequences*. Cambridge University Press, 2003.

[AU71] Alfred Aho and Jeffrey Ullman. Translations on a context-free grammar. *Information and Control*, 19(5):439–475, 1971.

[Bár08] Vince Bárány. A hierarchy of automatic ω-words having a decidable MSO theory. *Informatique Théorique et Applications*, 42(3):417–450, 2008.

[BC01] Stephen Bloom and Christian Choffrut. Long words: the theory of concatenation and omega-power. *Theoretical Computer Science*, 259(1-2):533–548, 2001.

[BC02] Véronique Bruyère and Olivier Carton. Hierarchy among automata on linear orderings. In *Proc. of IFIP*, pages 107–118. Kluwer, B.V., 2002.

[BC06a] Alexis Bès and Olivier Carton. A Kleene theorem for languages of words indexed by linear orderings. *International Journal of Foundations of Computer Science*, 17(3):519–542, 2006.

[BC06b] Mikolaj Bojanczyk and Thomas Colcombet. Tree-walking automata cannot be determinized. *Theoretical Computer Science*, 350(2-3):164–173, 2006.

[BC07] Véronique Bruyère and Olivier Carton. Automata on linear orderings. *Journal of Computer and System Sciences*, 73(1):1–24, 2007.

[BC08] Mikolaj Bojanczyk and Thomas Colcombet. Tree-walking automata do not recognize all regular languages. *SIAM Journal of Computing*, 38(2):658–701, 2008.

[BC10] Laurent Braud and Arnaud Carayol. Linear orders in the pushdown hierarchy. In *Proc. of ICALP*, 2010.

[BCL07] Achim Blumensath, Thomas Colcombet, and Christof Löding. Logical theories and compatible operations. *Logics and Games*, 2:75–109, 2007.

[BÉ09] Stephen Bloom and Zoltán Ésik. Scattered algebraic linear orderings. In *Proc. of FICS*, pages 25–30, 2009.

[BÉ10] Stephen Bloom and Zoltán Ésik. Algebraic ordinals. *Fundamenta Informaticae*, 99(4):383–407, 2010.

[Blu08] Achim Blumensath. On the structure of graphs in the Caucal hierarchy. *Theoretical Computer Science*, 400:19–45, 2008.

[BNR+10] Mikolaj Bojanczyk, Damian Niwiński, Alexander Rabinovich, Adam Radziwonczyk-Syta, and Michal Skrzypczak. On the Borel complexity of MSO definable sets of branches. *Fundamenta Informaticae*, 98(4):337–349, 2010.

[Boj08] Mikolaj Bojanczyk. Tree-walking automata. In C. Martín-Vide, F. Otto, and H. Fernau, editors, *Proc. of LATA*, volume 5196 of *Lecture Notes in Computer Science*, pages 1–2. Springer, 2008.

[Bra] Laurent Braud. Order-2 morphic words and recursion schemes. In preparation.

[Bra09] Laurent Braud. Covering of ordinals. In *Proc. of FSTTCS*, pages 97–108, 2009.

[Büc62] Richard Büchi. On a decision method in the restricted second-order arithmetic. *Logic, Methodology and Philosophy of science : Proc. Intern. Congr.*, pages 1–11, 1962.

[Büc73] Richard Büchi. The monadic theory of all countable ordinals. *Lecture Notes in Mathematics*, 328:1x–217, 1973.

[Bü65] Richard Büchi. Decision methods in the theory of ordinals. *Bulletin of the AMS*, 71:767–770, 1965.

[Cac06] Thierry Cachat. Tree automata make ordinal theory easy. In S. Arun-Kumar and N. Garg, editors, *Proc. of FSTTCS*, volume 4337 of *Lecture Notes in Computer Science*, pages 285–296. Springer, 2006.

[Can97] Georg Cantor. Beiträge zur begründung der transfiniten mengenlehre. *Mathematische Annalen*, 46/49:481–512/207–246, 1895/1897. Two-parts article. Translated in french by F. Marotte : Sur les fondements de la théorie des ensembles tranfinis, 1989.

[Car05] Arnaud Carayol. Regular sets of higher-order pushdown stacks. In *Proc. of MFCS*, volume 3618 of *Lecture Notes in Computer Science*, pages 168–179, 2005.

[Car06] Arnaud Carayol. *Automates infinis, logiques et langages.* PhD thesis, Université de Rennes 1, 2006.

[Cau92] Didier Caucal. On the regular structure of prefix rewriting. *Theoretical Computer Science*, 106(1):61–86, 1992.

[Cau96] Didier Caucal. On infinite transition graphs having a decidable monadic theory. In *Proc. of ICALP*, volume 1099 of *Lecture Notes in Computer Science*, pages 194–205, 1996.

[Cau02] Didier Caucal. On infinite terms having a decidable monadic theory. In *Proc. of MFCS*, volume 2420 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2002.

[Cau03] Didier Caucal. On infinite transition graphs having a decidable monadic theory. *Theoretical Computer Science*, 290(1):79–115, 2003.

[Cha33] David Champernowne. The construction of decimals normal in the scale of ten. *Journal of London Mathematical Society*, 8:254–260, 1933.

[Chu38] Alonzo Church. The constructive second number class. *Bulletin of the AMS*, 44:224–232, 1938.

[CK01] Didier Caucal and Teodor Knapik. An internal presentation of regular graphs by prefix-recognizable graphs. *Theoretical Computer Science*, 34(4):299–336, 2001.

[CK02] Bruno Courcelle and Teodor Knapik. The evaluation of first-order substitution is monadic second-order compatible. *Theoretical Computer Science*, 281(1-2):177–206, 2002.

[CL07] Thomas Colcombet and Christof Löding. Transforming structures by set interpretations. *Logical Methods in Computer Science*, 3-2(4), 2007.

[Cou78] Bruno Courcelle. Frontiers of infinite trees. *Informatique Théorique et Applications*, 12(4), 1978.

[Cou90] Bruno Courcelle. Graph rewriting: An algebraic and logic approach. In *Handbook of TCS, Volume B: Formal Models and Semantics*, pages 193–242. MIT Press, 1990.

[Cou94] Bruno Courcelle. Monadic second-order definable graph transductions: A survey. *Theoretical Computer Science*, 126(1):53–75, 1994.

[Cou11] Bruno Courcelle. *Graph structure and monadic second-order logic*. Cambridge University Press, 2011. to appear.

[CT02] Olivier Carton and Wolfgang Thomas. The monadic theory of morphic infinite words and generalizations. *Information and Computation*, 176(1):51–65, 2002.

[CW98] Bruno Courcelle and Igor Walukiewicz. Monadic second-order logic, graph coverings and unfoldings of transition systems. *Ann. Pure Appl. Logic*, 92(1):35–62, 1998.

[CW03] Arnaud Carayol and Stefan Wöhrle. The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *Proc. of FSTTCS*, volume 2914 of *Lecture Notes in Computer Science*, pages 112–123. Springer, 2003.

[Dam77] Werner Damm. Languages defined by higher type program schemes. In A. Salomaa and M. Steinby, editors, *Proc. of ICALP*, volume 52 of *Lecture Notes in Computer Science*, pages 164–179. Springer, 1977.

[Dam82] Werner Damm. The IO- and OI-hierarchies. *Theoretical Computer Science*, 20:95–207, 1982.

[Del04] Christian Delhommé. Automaticité des ordinaux et des graphes homogènes. *C. R. Acad. Sci. Paris*, Ser. I 339:5–10, 2004.

[Fra05] Séverine Fratani. *Automates à piles de piles... de piles*. PhD thesis, Université Bordeaux I, 2005.

[FS06] Séverine Fratani and Géraud Sénizergues. Iterated pushdown automata and sequences of rational numbers. *Annals of Pure and Applied Logic*, 141(3):363–411, 2006.

[Gai82] Haim Gaifman. On local and non-local properties. In J. Stern, editor, *Proceedings of the Herbrand Symposium*, volume 107 of *Studies in Logic and the Foundations of Mathematics*, pages 105 – 135. Elsevier, 1982.

[Gur85] Yuri Gurevich. Monadic second-order theories. *Model-Theoretic Logic*, pages 479–506, 1985.

[Hau08] Felix Hausdorff. Grundzüge einer theorie der geordnete mengen. *Math. Ann.*, 65:435–505, 1908.

[Hei80] Stephan Heilbrunner. An algorithm for the solution of fixed-point equations for infinite words. *Informatique Théorique et Applications*, 14(2):131–141, 1980.

[Ian60] Iu Ianov. The logical schemes of algorithms. *English translation in Problems of Cybernetics*, 1:82–140, 1960.

[Kec94] Alexander S. Kechris. *Classical Descriptive Set Theory.* Springer-Verlag, 1994.

[Kle38] Stephen Kleene. On notation for ordinal numbers. *Journal of Symbolic Logic*, 3(4):150–155, 1938.

[Knu76] Donald Knuth. Coping with finiteness. *Science*, 194(4271):1235–1242, 1976.

[KNU01] Teodor Knapik, Damian Niwiński, and Paweł Urzyczyn. Deciding monadic theories of hyperalgebraic trees. In *Proc. of TLCA*, pages 253–267, 2001.

[KNU02] Teodor Knapik, Damian Niwiński, and Paweł Urzyczyn. Higher-order push-down trees are easy. In *Proc. of FoSSaCS*, Lecture Notes in Computer Science, pages 205–222, 2002.

[KNUW05] Teodor Knapik, Damian Niwinski, Pawel Urzyczyn, and Igor Walukiewicz. Unsafe grammars and panic automata. In L. Caires, G. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *Proc. of ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 1450–1461. Springer, 2005.

[KRS05] Bakhadyr Khoussainov, Sacha Rubin, and Frank Stephan. Automatic linear orders and trees. *ACM Trans. Comput. Log.*, 6(4):675–700, 2005.

[Lav05] Thomas Lavergne. *Prédicats algébriques d'entiers.* Master thesis, Université de Rennes, 2005.

[lG06] Marion le Gonidec. *Sur la complexité des mots q^∞-automatiques.* PhD thesis, Université de la Méditerranée, 2006.

[Mar07] Nathalie Marin. *Suites de mots et automates.* Master thesis, Université de Bordeaux 1, 2007.

[Mas74] A. N. Maslov. The hierarchy of indexed languages of an arbitrary level. *Soviet Math. Dokl.*, 15:1170–1174, 1974.

[MS85] David Muller and Paul Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, 37(1):51–75, 1985.

[Niv72] Maurice Nivat. Langages algébriques sur le magma libre et sémantique des schémas de programme. In *Proc. of ICALP*, pages 293–308, 1972.

[NP82] Maurice Nivat and Dominique Perrin. Ensembles reconnaissables de mots biinfinis. In *Proc. of STOC*, pages 47–59. ACM, 1982.

[Ong06] Luke Ong. On model-checking trees generated by higher-order recursion schemes. In *Proc. of LICS*, pages 81–90. IEEE Computer Society, 2006.

[Par10] Paweł Parys. Collapse operation increases expressive power of deterministic higher order pushdown automata. Accepted at ICALP, 2010.

[Pil04] Julien Pillot. Produits de graphes infinis et logique monadique. Master's thesis, IRISA — ENST Bretagne, 2004.

[Rab69] Michael Rabin. Decidability of second-order theories and automata on infinite trees. *Transaction of the AMS*, 141:1–35, 1969.

[Rog87] Hartley Rogers. *Theory of recursive functions and effective computability.* MIT Press, Cambridge, MA, USA, 1987.

[Roi90] Judith Roitman. *Introduction to Modern Set Theory.* John Wiley and Sons, 1990.

[Ros82] Joseph G. Rosenstein. *Linear orderings.* Academic Press Inc., 1982.

[RS08] Alexander Rabinovich and Amit Shomrat. Selection in the monadic theory of a countable ordinal. *Journal of Symbolic Logics*, 73(3):783–816, 2008.

[See91] Detlef Seese. The structure of the models of decidable monadic theories of graphs. *Annals of pure and applied logic*, 53(2):169–195, 1991.

[Sem84] Alexei Semenov. Decidability of monadic theories. In M. Chytil and V. Koubek, editors, *Proc. of MFCS*, volume 176 of *Lecture Notes in Computer Science*, pages 162–175. Springer, 1984.

[She75] Saharon Shelah. The monadic theory of order. *Annals of Mathematics*, 102(3):379–419, 1975.

[Tho86] Wolfgang Thomas. On frontiers of regular trees. *Informatique Théorique et Applications*, 20(4):371–381, 1986.

[Tho97a] Wolfgang Thomas. Ehrenfeucht games, the composition method, and the monadic theory of ordinal words. In J. Mycielski, G. Rozenberg, and A. Salomaa, editors, *Structures in Logic and Computer Science*, volume 1261 of *Lecture Notes in Computer Science*, pages 118–143. Springer, 1997.

[Tho97b] Wolfgang Thomas. Languages, automata, and logic. In *Handbook of Formal Language Theory*, volume III, pages 389–455. Springer-Verlag, 1997.

[Tho08] Wolfgang Thomas. Model transformations in decidability proofs for monadic theories. In M. Kaminski and S. Martini, editors, *Proc. of CSL*, volume 5213 of *Lecture Notes in Computer Science*, pages 23–31. Springer, 2008.

[Veb08] Oswald Veblen. Continuous increasing functions of finite and transfinite ordinals. *Transactions of the American Mathematical Society*, 9(3):280–292, 1908.

[Wal02] Igor Walukiewicz. Monadic second-order logic on tree-like structures. *Theoretical Computer Science*, 275(1-2):311–346, 2002.

[Zie98] Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.