Volumetry of timed languages and applications

Nicolas Basset

UNIVERSITÉ PARIS-EST

THESE

Pour obtenir le grade de
DOCTEUR de l'UNIVERSITE PARIS-EST

Spécialité
Informatique
Ecole Doctorale : Mathématiques et Sciences et Technologies de l'Information et de la communication

Présentée par

Nicolas Basset

Volumetry of timed languages and applications

sous la co-direction de
Eugène Asarin (LIAFA) *et* Dominique Perrin (LIGM)

soutenue le 5 décembre 2013 devant le jury composé de :

M. Rajeev Alur	Rapporteur
M. Eugène Asarin	Co-directeur de thèse
Mme Christel Baier	Examinatrice
M. Jean Mairesse	Directeur du jury
M. Dominique Perrin	Co-directeur de thèse
Mme Brigitte Vallée	Rapporteur

Remerciements

Je tiens à remercier Eugene Asarin et Dominique Perrin pour m'avoir encadré pendant cette thèse ainsi qu'Aldric Degorre pour son aide et ses conseils aussi bien sur le plan humain que scientifique et orthographique.

Je remercie Marie-Pierre Béal et Ahmed Bouajjani pour m'avoir tous deux conseillé de soumettre le chapitre 4 de cette thèse à ICALP et je remercie le comité de programme d'ICALP 2013 track B pour m'avoir décerné le "best student paper award". Merci aussi à Peter Habermehl pour des conseils similaires.

Merci à Marta Kwiatkowska et son équipe pour m'avoir accueilli en postdoc à Oxford deux mois avant même que je soutienne ma thèse.

Je remercie aussi ceux qui m'ont aidé pour ma mission d'enseignement: Nicolas Bedon, Cyril Nicaud, Didier Caucal, Gregory Kutcherov...

Je remercie mon père et ma mère pour m'avoir transmis leur gout de la science et des maths respectivement. Et surtout merci à Sonia, Gaël, Délia nés pendant la thèse où juste après pour avoir égayé mes temps libres et merci à leur maman Eve pour s'en être bien occupé, m'avoir soutenu et aidé.

Contents

Résumé introductif en français. **9**

1 Introduction **19**
1.1 Contributions, extended outline 21
1.2 Related work 28
1.2.1 Classical results lifted to the timed case. 28
1.2.2 Timed automata related works 29
1.2.3 Combinatorics 30
1.3 Past and ongoing publications 30

2 Preliminaries **33**
2.1 Basics definitions 33
2.1.1 Timed languages 33
2.1.2 Bounded Deterministic Timed Automata 34
2.1.3 Timed region graphs 36
2.2 Advanced preliminaries 37
2.2.1 Paths, polytopes and point to point reachability. 37
2.2.2 Closed version of a timed region graph 39
2.2.3 SCC decomposition 39
2.2.4 Volume and entropy of runs 40
2.3 Link between BDTA and TRGs (technical section). 41
2.3.1 The region-splitting of [AD09a] 42
2.3.2 BDTAs and TRGs have the same entropy 42
2.3.3 Proof of Proposition 5 43
2.3.4 Recurrent equations on volume functions [AD09a] 44

3 Thin and Thick languages **47**
3.1 Preliminaries 48
3.1.1 Thinness, simplices and examples 48
3.1.2 Point to point reachability: algebraic characterization 49
3.1.3 Linear Lyapunov functions and sub-exponential volume. 51
3.2 Main section 54
3.2.1 Pumping lemma for long thick paths 54

3.2.2 Characterizing thick languages . . . 56
3.2.3 Thin and thick SCC . . . 57
3.3 Conclusion and perspectives . . . 58

4 The maximal entropy SPOR. 59
4.1 Maximal entropy Markov chain on a graph . . . 60
4.1.1 Markov chain on a graph . . . 60
4.1.2 Ergodic stochastic processes . . . 60
4.1.3 Entropies . . . 61
4.1.4 The asymptotic equipartition property for Markov-chain . . . 62
4.1.5 The Shannon-Parry Markov chain . . . 62
4.2 Stochastic processes on timed region graphs . . . 63
4.2.1 SPOR of a timed region graph . . . 64
4.2.2 Entropy . . . 65
4.3 The maximal entropy SPOR . . . 67
4.3.1 Technical assumptions . . . 67
4.3.2 Main theorems . . . 68
4.3.3 Definition and properties of ρ, v and w . . . 69
4.3.4 Examples . . . 78
4.3.5 Proof of the maximal entropy theorem (Theorem 14) . . . 80
4.4 Conclusion and perspectives . . . 86
4.4.1 Technical challenges . . . 86

5 Timed symbolic dynamics 87
5.1 Preliminaries . . . 88
5.1.1 Words and factors . . . 88
5.1.2 Topology . . . 88
5.1.3 Shift spaces . . . 89
5.1.4 ε-entropies and topological entropy . . . 90
5.2 Classical symbolic dynamics . . . 90
5.2.1 Characterization with finite factors . . . 91
5.2.2 Edge and sofic shifts . . . 91
5.2.3 The language point of view. . . . 91
5.3 Compact alphabet shift space . . . 92
5.3.1 Factor based characterization of shift spaces. . . . 92
5.3.2 An infinite topological entropy . . . 93
5.3.3 Entropy of a mesurable shift . . . 93
5.3.4 Keeping the ε-entropy . . . 94
5.4 Timed edge shift and timed sofic shift . . . 95
5.4.1 Definitions . . . 96
5.4.2 The timed language point of view . . . 97
5.4.3 Discretization . . . 100
5.4.4 Metric mean dimension . . . 106

5.5 Sliding block codes . . . 111
5.6 Conclusion and perspectives . . . 112
5.6.1 Open problems . . . 112

6 Toward a Timed Theory of Channel Coding 115
6.1 Theory of channel coding for finite alphabet languages . . . 115
6.1.1 Terminology . . . 115
6.1.2 Coding: the basic case . . . 116
6.1.3 Other coding settings . . . 118
6.2 Timed coding . . . 118
6.2.1 Timed source, discrete channel, approximate transmission . . . 119
6.2.2 Timed source, timed channel, exact transmission . . . 121
6.2.3 A variant: scaling allowed . . . 124
6.2.4 A speedup and a slowdown lead to a collapse . . . 126
6.3 Conclusion and perspectives . . . 127

7 Generating functions of timed languages 129
7.1 Preliminaries . . . 129
7.1.1 Clock languages and timed languages . . . 129
7.1.2 From timed automata to triplet, clock and timed languages . . . 130
7.1.3 Volume(s) of timed and clock languages . . . 132
7.2 Generating functions . . . 136
7.2.1 Definitions . . . 136
7.2.2 Analytic characterization . . . 136
7.2.3 Volumes, generating functions and functional analysis . . . 137
7.2.4 Inductive characterization of generating functions . . . 138
7.3 Computing generating functions . . . 140
7.3.1 Generating functions for particular classes of automata . . . 140
7.4 Conclusion and perspectives . . . 146

8 Combinatorics using timed languages 147
8.1 Two problem statements . . . 147
8.2 A timed and geometric approach . . . 149
8.2.1 Order sets of a language of signatures $(\mathcal{O}_n(L))_{n\geq 1}$ . . . 149
8.2.2 Timed semantics of a language of signatures $(\mathbb{L}'_n)_{n\in\mathbb{N}}$ . . . 150
8.2.3 Volume preserving transformation between $\mathbb{L}'_n$ and $\mathcal{O}_n(L)$. . . . 152
8.2.4 The S-T (timed) language encoding. . . . 154
8.3 Solving the two problems . . . 155
8.3.1 Characterization of the VGF of an S-T-automaton. . . . 155
8.3.2 An algorithm for Problem 2 . . . 156
8.4 Examples . . . 160
8.4.1 The alternating permutations . . . 160
8.4.2 The up-up-down-down permutations . . . 161

8.4.3 Permutations without two consecutive descents 162
8.5 Conclusion and perspectives 163

9 Conclusion and perspectives **165**

Bibliography **173**

List of Figures **175**

Abstract and Résumé **176**

Résumé introductif en français.

Deux théories pour un objet d'étude. La théorie des automates et celle de la dynamique symbolique ont été développées en parallèle depuis le milieu du $20^{\text{ème}}$ siècle. Ces deux théories partagent le même objet d'étude : l'ensemble des chemins d'un graphe étiqueté fini. Pour la première théorie, cet ensemble est un langage régulier (et le graphe étiqueté est appelé automate) tandis que pour la deuxième théorie c'est l'ensemble des blocs autorisés d'un ensemble de décalage sofique (et le graphe étiqueté est appelé présentation de l'espace de décalage). Nous conseillons au lecteur de lire le chapitre [BBEP10] pour une exposition de la dynamique symbolique dans le contexte de la théorie des automates.

Une analyse quantitative donnée par la dynamique symbolique. La dynamique symbolique apporte une analyse quantitative des langages réguliers grâce à la notion d'entropie. Celle-ci mesure le taux de croissance du langage par rapport à la taille des mots considérés. Sa définition formelle est détaillée ci-après. Soit L_n un ensemble de mots de taille n d'un langage régulier L. Dans la majorité des cas, son cardinal $|L_n|$ se comporte à peu près comme une exponentielle ρ^n multipliée à un terme sous-exponentiel de telle sorte que $\lambda^{-n} << (|L_n|\rho^{-n}) << \lambda^n$ pour tout $\lambda > 1$. L'entropie $h(L)$ est définie comme le logarithme en base 2 de ρ, en d'autres termes $|L_n|$ se comporte comme $2^{nh(L)}$ et formellement

$$h(L) = \limsup_{n\to\infty} \frac{1}{n} \log_2 |L_n|.$$

L'entropie $h(L)$ s'interprète naturellement comme le nombre de bits par symbole pour coder un mots de L.

La dynamique symbolique considère les langages réguliers comme des systèmes dynamiques appelés espaces de décalage sofiques et offre un aspect topologique. L'entropie des espaces de décalage sofiques est un cas particulier de l'entropie topologique des systèmes dynamiques.

Du cas discret au cas continu. Les notions de décalage et de temps dans le contexte de la dynamique symbolique sont discrets : le décalage d'une lettre (par la suite appelé événement) vers la droite permet d'avancer d'un pas dans le futur tandis que le décalage d'un événement vers la gauche permet de reculer d'un pas dans le passé. De la même façon, dans le contexte de la vérification, les mots d'un langage régulier représentent l'histoire possible

d'un système. Chaque lettre représente un événement et les événements se produisent les uns après les autres sans quantifier le temps entre eux.

Cependant, il est souvent pertinent de quantifier de façon continue les délais entre les événements. Les automates temporisés ont été introduits pour cette raison dans le années 90 par Alur et Dill [AD94]. Les automates temporisés sont rapidement devenus populaires comme un modèle simple mais expressif vérifiant beaucoup de résultats de décidabilités tels que la vérification de modèle ou les problèmes d'accessibilité (équivalent au problème du vide pour les langages). Depuis les automates temporisés ont été minutieusement étudiés d'un point de vue théorique.

Volumétrie des langages temporisés. La notion de temps continu entre les événements n'a pas encore été étudiée dans le contexte de la dynamique symbolique. Dans la théorie des automates temporisés, une analyse quantitative de la taille des langages temporisés et l'information contenue dans leur éléments ont seulement été introduites récemment par deux de mes collègues Eugene Asarin et Aldric Degorre. Dans ce but, ils ont défini dans [AD09a, AD09b] les notions de volume et d'entropie (inspirées par la dynamique symbolique) des langages temporisés et construit une nouvelle théorie que nous appelons la *volumétrie de langages temporisés*. Dans le paragraphe suivant, nous rappellerons brièvement la définition de volume et d'entropie des langages temporisés.

Un *mot temporisé* $t_1 w_1 \cdots t_n w_n$ est un mot formé d'une alternance de *délais* de valeur réelles positives $t_i \geq 0$ et d'événements a_i. Un *langage temporisé* est un ensemble de mots temporisés. D'un point de vue géométrique, un mot temporisé est un vecteur de délais $(t_1, \ldots, t_n) \in \mathbb{R}^n$ étiqueté par un mot d'événements $w = w_1 \cdots w_n \in \Sigma^n$ où Σ représente un alphabet d'événements. Nous adoptons la convention d'écriture suivante : $(\vec{t}, w)$ est le mot temporisé $t_1 w_1 \cdots t_n w_n$ avec $\vec{t} = (t_1, \cdots, t_n)$ et $w \in \Sigma^n$ $(n \geq 1)$. En continuant avec la même convention, étant donné un langage temporisé $\mathbb{L} \subseteq (\mathbb{R}^+ \times \Sigma)^*$, le langage temporisé restreint aux mots de taille n, $\mathbb{L}'_n$ peut être vu comme une union formelle d'ensembles $\biguplus_{w \in \Sigma^n} \mathbb{L}_w \times \{w\}$ où $\mathbb{L}_w = \{\vec{t} \in \mathbb{R}^n \mid (\vec{t}, w) \in \mathbb{L}'\}$ est l'ensemble des vecteurs de délais qui avec w forme un mot temporisé de $\mathbb{L}$. Dans la thèse, nous considérons uniquement des langages $\mathbb{L}$ tel que chaque $\mathbb{L}_w$ a un volume mesurable. Pour un langage temporisé $\mathbb{L}$, on peut définir sa séquence de volumes :

$$\mathtt{Vol}(\mathbb{L}_n) = \sum_{w \in \Sigma^n} \mathtt{Vol}(\mathbb{L}_w)$$

et son entropie volumétrique :

$$\mathcal{H}(\mathbb{L}) = \limsup_{n \to +\infty} \frac{1}{n} \log_2 \mathtt{Vol}(\mathbb{L}_n).$$

Ainsi, la définition de l'entropie d'un langage temporisé imite celle de l'entropie des langages réguliers en remplaçant l'énumération des mots de taille n par la mesure du volume des mots de taille n (qui est indénombrable).

Dans la thèse, nous utilisons comme exemple principal l'automate $\mathcal{A}^{ex}$ représenté sur la Figure 1, pour illustrer les notions de volume, d'entropie, de processus d'entropie maximale

ou de fonctions génératrices. Le langage temporisé reconnu par l'automate $\mathcal{A}^{ex}$ est

$$L = \{t_1 a t_2 b t_3 a \dots | \forall i \ (t_i + t_{i+1} \leq 1)\}.$$

Pour tout nombre d'événements n nous avons un polytope dans $\mathbb{R}^n$:

$$L_n = \{(t_1, t_2, t_3, \dots, t_n) | \forall i \ (t_i + t_{i+1} \leq 1)\},$$

la séquence de volumes V_n de ces polytopes est :

$$1; 1; \frac{1}{2}; \frac{1}{3}; \frac{5}{24}; \frac{2}{15}; \frac{61}{720}; \frac{17}{315}; \frac{277}{8064} \dots,$$

et il est démonté dans [AD09a] que cette séquence se comporte asymptotiquement comme $(2/\pi)^n$ (l'entropie du langage est $\log_2(2/\pi)$).

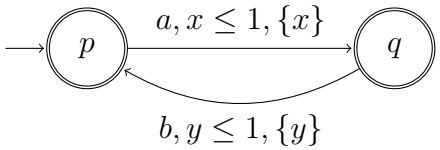

Figure 1: Un automate temporisé $\mathcal{A}^{ex}$

Dans cette thèse nous construisons de nouveaux développements à cette théorie (que nous appelons volumétrie des langages temporisés) et l'appliquons a plusieurs problèmes apparaissant dans divers domaine de recherche tel que la théorie de l'information, la vérification, la combinatoire énumérative.

Entre autre nous

- développons une théorie de la dynamique symbolique temporisées ;
- caractérisons une dichotomie entre automates temporisés se comportant bien ou mal ;
- définissons pour un automate temporisé donné, un processus stochastique d'entropie maximale le moins biaisé possible ;
- développons une version temporisé de la théorie des codes sur canal contraint ;
- énumérons et générons aléatoirement des permutations dans une certaine classe ;

Notre travail améliore considérablement la volumétrie des langages temporisés car nous

- fournissons les applications énumérées ci-dessus ;
- éliminons les conditions restrictives imposées dans [AD09b, AD09a] ;
- caractérisons algébriquement (et algorithmiquement) les langages d'entropie fini ;

- caractérisons précisément la suite de volumes d'un langage temporisé grâce a sa fonction génératrice des volumes ;

- prouvons et améliorons la convergence de la méthode de discrétisation pour le calcul de l'entropie proposé dans [AD09b].

- fournissons plusieurs définitions équivalentes de l'entropie, chacune ayant ses propres avantages selon le contexte.

Nous donnons le cadre de dynamique symbolique temporisé dans le chapitre central (Chapitre 5). Même si nous aurions pu écrire une bonne partie de la thèse en terme de dynamique symbolique temporisé nous avons préféré dans la plupart des cas utiliser le vocabulaire des automates temporisés. Ceci afin de simplifier la réutilisation de nos contributions, qui sont plus proches de la théorie des automates temporisés.

Ci-dessous nous décrivons chapitre par chapitre nos contributions dans l'ordre dans lequel elles apparaissent dans la thèse.

Chapitre 2. Préliminaires.

Dans ce chapitre de préliminaires, nous rappelons les définitions de volume et d'entropie des automates temporisés introduites par Asarin et Degorre. Nous introduisons le modèle de graphe de régions temporisés qui est utilisé dans la plupart des chapitres de cette thèse. Chaque automates temporisé a un graphe de région temporisé sous-jacent. Nous réduisons donc l'étude de l'entropie des automates temporisés à celle des graphes de régions temporisées. Nous introduisons une caractérisation de l'entropie basé sur la mesure des calculs (Proposition 5) qui

- coïncide dans le cas déterministe avec la définition originale basée sur le volume des mots temporisé du langage,

- est simple à comparer avec l'entropie d'un processus stochastique sur les calculs introduite plus tard au Chapitre 4.

Chapitre 3. L'alternative gras et maigre.

Les automates temporisés utilisant des horloges analogiques exactes, des gardes exactes et des mises à 0 sont de beaux objet mathématiques et sont des modèles utiles pour les systèmes temps-réels. Cependant, dès le tout début des recherches sur les automates temporisés, il parut évident qu'ils sont, sur plusieurs aspects, trop précis, qu'ils ont parfois d'étranges artéfacts, des pathologies mathématiques ou encore mènent à des modèles irréalistes. De nombreuses lignes de recherches ont expliqués en parties ces questions.

L'espace d'états d'un automate temporisé étant infini, des calculs longs (ou infinis) ne permettront jamais de revisiter le même état. Pour cette raison, comme stipulé dans [Bea98], le lemme d'itération habituel ne marche pas et doit plutôt être remplacé par un analogue

plus complexe. Dans un calcul, beaucoup d'événements peuvent se produire un nombre infini de fois durant une portion de temps fini, ou deux événements peuvent se produire encore et encore avec un intervalle de temps entre eux tendant vers 0. De tels calculs rappellent les paradoxes de Zénon et sont souvent appelé calculs Zénon ("Zeno runs" en anglais), voir [GB07] et ces références.

Nous introduisons la notion clé de cycles *d'oubli*, un critère pour distinguer les langages *maigres*, ayant une entropie $-\infty$, et les langages *gras*, ayant une entropie $> -\infty$.

Cette notion est primordiale pour repérer et éliminer les automates temporisés aux comportements pathologiques, comme nous le faisons dans plusieurs chapitres de cette thèse (Chapitres 4,5 et 6).

La dichotomie entre maigre et gras se résume ainsi :

Dans les automates maigres toutes les trajectoires infinies sont, en un sens faible, Zénon; la discrétisation des longues trajectoires est difficile, puisque qu'elle nécessite de très petits pas de discrétisation.

Dans les automates gras la majorité des trajectoires sont non-Zénon et se comportent bien pour la discrétisation; de tels automates ont des cycles d'oubli et la plupart des trajectoires visite de tels cycles.

Le principal résultat de ce chapitre (Théorème 5) établit que la propriété d'être gras (c'est à dire $\mathcal{H} > -\infty$) est équivalente au autres propriétés décrites brièvement ci-après : une bonne discrétisation, l'existence d'un cycle d'oubli, etc.). Nous montons dans le même temps que tout long chemin d'un automate temporisé gras, contient nécessairement un cycle d'oubli (Théorème 3), qui peut être assimilé à un lemme de pompage pour chemins gras.

La preuve du Théorème 5 est assez technique. Elle utilise un mélange d'outils "temporisés" inspirés de [Pur00, AKY10] ainsi que le théorème de factorisation de forêt de Simon [Sim90] (qui constitue un raffinement de la théorie de Ramsey pour les monoïdes).

Etre gras, une condition necessaire et suffisante au calcul de l'entropie. Dans le travail original d'Asarin et Degorre, deux approches sont étudiées pour calculer l'entropie, l'approche par discrétisation [AD09b] et l'approche par opérateur[AD09a]. Cependant, la convergence de ces algorithmes pour ces deux approches n'est pas prouvée et une condition restrictive sur les mises à 0 des horloges au cours de chaque cycle est nécessaire. Dans cette thèse, nous nous débarrassons de cette condition restrictive et résolvons le problème de convergence de l'approche par discrétisation. Nous montrons aussi la convergence de l'approche par opérateur dans [ABD13]. Ces deux solutions sont basés sur la notion des cycles d'oubli introduites dans ce chapitre. Plus précisément, le cas $\mathcal{H} = -\infty$ (équivalent à l'absence de cycle d'oubli) peut être détecté avec une procédure PSPACE décrite dans ce chapitre. Quand ce cas est éliminé, le fait d'être gras ($\mathcal{H} > -\infty$, équivalent à la présence d'un cycle d'oubli) devient une condition nécessaire et suffisante pour que les deux méthodes convergent (voir [BA11] et Chapitre 5 pour l'approche par discrétisation, [ABD13] pour l'approche par opérateurs).

Chapitre 4. Un processus stochastique d'entropie maximale pour un graphe de région temporisé.

Dans le contexte de vérification de système temps réel, plusieurs paramètres probabilistes ont été ajoutés aux automates temporisés (voir les références ci-après). Il y a de nombreuses raisons d' ajouter des probabilités : cela permet (i) un meilleur reflet des systèmes physiques qui se comporte aléatoirement, (ii) une réduction de la taille du modèle en émondant les comportement de probabilités nulles [BBB+07], (iii) de résoudre le non-déterminisme lié aux composition parallèle d'automates[DLL+11, KBM13].

Dans la majorité des travaux précédents sur le sujet (voir e.g. [BA07, BAM06, BBBM08, DLL+11]), les distributions de probabilité sur des transitions continues et discrètes sont données en même temps que les paramètres de temps. Dans ces travaux, la distribution de probabilité est donné par le concepteur du modèle. Alors que, il peut vouloir fournir uniquement un automate temporisé et s'interroger sur le "meilleur" choix de distribution de probabilité pour l'automate temporisé considéré? Un tel "meilleur" choix doit transformer l'automate temporisé en un générateur aléatoire de calculs le moins biaisé possible, c'est à dire qu'il devrait générer des calculs aussi uniformément que possible pour couvrir avec une grande probabilité le maximum des comportements du système modélisé. Plus précisément, la probabilité qu'un calcul généré appartienne à un ensemble donné devrait être proportionnelle à la taille (volume) de cet ensemble. Nous formalisons cette question et proposons une réponse basée sur la notion d'entropie des automates temporisés introduite dans [AD09a].

La théorie développée par Shannon [Sha48] et ses successeurs permets de résoudre le problème analogue de la génération aléatoire quasi-uniforme des chemins d'un graphe fini. Ce problème peut être formulé comme suit : étant donné un graphe fini G, comment peut-on trouver une chaîne de Markov stationnaire sur G permettant de générer des chemins de la façon la plus uniforme? La réponse est en deux points (voir Chapitre 1.8 de [Lot05] et aussi la section 13.3 de [LM95]) : (i) Il existe une chaîne de Markov stationnaire sur G avec entropie maximale : la chaîne de Markov de Shannon et Parry ; (ii) cette chaîne de Markov stationnaire permet de générer des chemins quasi uniformément.

Dans ce chapitre nous étendons cette théorie au cadre des automates temporisés. Nous définissons les processus stochastiques sur les calculs des graphes de régions temporisées et leur entropie (continue). Cette généralisation des chaînes de Markov pour les automates temporisés a un intérêt en soit (c'est la première fois par exemple qu'une distribution de probabilité sur les états initial est donnée). Un tel graphes de régions temporisées permets de générer des calculs aléatoires pas à pas en temps linéaire. Comme résultat principal, nous décrivons un graphe de régions temporisées d'entropie maximal stationnaire ergodique et généralisant la chaîne de Markov de Shannon et Parry pour les automates temporisés (Théorème 14). Les concepts d'entropie maximale, stationnarité et ergodicité peuvent être intéressants en eux-même. Ici, nous les utilisons comme des hypothèses clés afin de garantir la quasi-uniformité de la génération aléatoire (Théorème 15). Plus précisement, le résultat que nous prouvons est une variante du célèbre théorème de Shannon-McMillan-Breiman aussi connu sous le nom de propriété d'équipartition asymptotique (AEP).

Chapitre 5. Dynamique symbolique temporisée.

Dans ce chapitre, nous établissons le cadre de théorie de la dynamique symbolique temporisé a la volumétrie des langages temporisés. Nous introduisons les espaces de décalages sofique qui sont des espace de décalage sur un alphabet compact. Du coté négatif, l'entropie topologique d'un tel espace de décalage tends vers l'infinie. Du coté positif, trois mesures de la taille des espace de décalages sofique peuvent néanmoins être décrites : la dimension métrique moyenne **mdim** de [LW00], l' ε-entropie h_ε et l'entropie volumétrique $\mathcal{H}$ inspirés de [AD09b, AD09a]. Nous expliquons le sens de ces trois mesures ainsi que leur lien commun. En particulier, nous résolvons le problème resté ouvert dans [AD09b] de l'approximation de l'entropie volumétrique par discrétisation. Formellement, nous montrons le développement asymptotique suivant qui constitue l'un des résultats principaux du chapitre :

$$h_\varepsilon = \mathcal{H} + \log_2(1/\varepsilon) + o(1).$$

Une interprétation de cette formule en terme de complexité de Kolmogorov est donné dans [AD09b]. Cette formule a aussi une interprétation naturelle en terme d'informations contenues dans un mot temporisé que nous expliquons et utilisons dans notre théorie des codes temporisés sur canal contraint (voir Chapitre 6).

Les codes à fenêtres glissantes (en anglais "sliding block code") sont des objets clés de la dynamique symbolique. Ils sont exactement, en vertu du théorème de Curtis-Hedlund-Lyndon, les morphismes des espaces de décalage (c'est à dire les fonctions continues qui commute avec le décalage). Dans le cas temporisé, il n'y a pas de correspondance entre morphismes et codes à fenêtres glissantes (Proposition 29). Cependant, nous montrons que tout morphisme vers un espace de décalages total est la limite uniforme d'une suite de codes à fenêtre glissantes continus (Théorème 30).

Chapitre 6. Vers une théorie temporisé des codes sur canal contraint.

La théorie développé dans ce chapitre est la première tentative de généralisation de la théorie classique des codes sur canal contraint aux langages temporisés.

Soit un langage, S représentant tous les messages possibles que peut généré une *source*, et C tous les messages qui peuvent être transmis par un *canal*. Les problèmes typiques de théorie des codes sont :

- Est-il possible de transmettre n'importe qu'elle message généré par la source via un canal?
- Quelle sera la vitesse de transmission?
- Comment coder le message avant et le décoder après la transmission?

Les réponses données par la théorie des codes sur canal contraint sont les suivantes : à chaque langage L est associé un réel positif $h(L)$, appelé son entropie, qui caractérise la quantité

d'information en bits par symbole. Dans le but de transmettre une information en temps réel (resp. avec vitesse α) l'entropie de la source ne doit pas excéder celle du canal (aussi nommé capacité) : $h(S) \leq h(C)$ (resp. $\alpha h(S) \leq h(C)$). Pour les langages réguliers (ou plus précisément sofique), Chaque fois que l'inégalité d'information précédente est stricte, la théorie des codes sur canal contraint fournie un protocole de transmission avec un encodage et décodage simple (réalisé par un transducteur à état fini). Dans la pratique, quand $S = \Sigma^*$ et $h(S) < h(C)$, le décodage peut être fait de façon simples (par fenêtre glissante). Un exemple typique est l'EFMPlus code [Imm95] permettant l'écriture de n'importe quel fichier binaire (i.e. la source $\{0,1\}^*$, avec entropie 1) sur un DVD (le canal $C = (2,10) - \mathrm{RLL}$ qui contient tous les mots sans facteurs 11, 101 et 0^{11}, son entropie est de 0.5418, voir [Bla90]) avec un taux presque optimal $\alpha = 1/2$.

La théorie classique des codes sur canal contraint traite de messages discrets. Il est cependant important de considérer des mots de donnés c'est à dire des mots discrets augmentés de donnés (par exemple des nombre réels). Dans ce chapitre, nous développons la théorie des codes sur canal contraint pour la classe de langage de donnés la plus étudiée : les langages temporisés. Plusieurs modèles de transmission d'informations sont alors possibles :

- La source est un langage temporisé ; le canal est un langage discret. Dans ce cas, un encodage avec perte minimale est impossible, et il faut considérer l'encodage avec une précision ε;
- La source et le canal sont des langages temporisés, nous pouvons faire un encodage exact (perte minimale);
- La source et le canal sont des langages temporisés, et les changement d'échelles sur les données temporisées sont autorisées.

L'entropie des langages temporisés est la notion clé pour résoudre ces problèmes. Pour beaucoup de modèles de transmission de données temporisées nous écrivons une *inegalité d'information* reliant l'entropies des sources et des canaux aux paramètres des encodages (taux, précision, graduation, voir ci-après). Une telle inégalité est une condition nécessaire à l'existence d'un encodage. D'un autre côté, Pour chaque inégalité d'information (dans sa forme stricte) si les langages sont réguliers (sofique), nous donnons une construction explicite d'une fonction simple d'encodage-décodage temporisé.

Techniquement, nous nous appuyons fortement sur une discrétisation quantitative des langages temporisés développé dans le Chapitre 5.

Chapitre 7. Fonctions génératrices des langages temporisés.

L'entropie est une mesure grossière de la taille des langages temporisés. Dans ce chapitre, nous faisons une analyse plus précise de la taille des langages temporisés acceptés par un automate temporisé déterministe. Nous associons à un langage L la séquence de ces volumes $\mathtt{Vol}(L_n)$, et sa fonction génératrice $f(z) = \sum_n \mathtt{Vol}(L_n) z^n$. Ainsi la fonction $f(z)$ contient une information complète sur le "profile de taille" $\mathtt{Vol}(L_n)$ en fonction de n. La fonction

génératrice $f(z)$ peut être exprimer en termes de résolvante de l'opérateur Ψ de Asarin et Degorre [AD09a], et l'entropie du langage temporisé dépend uniquement du rayon de convergence $f(z)$.

Les méthodes développées dans ce chapitre produisent par exemple une formule close pour la fonction génératrice des volumes de l'exemple courant (automate $\mathcal{A}^{ex}$ de la Figure 1.1) :

$$f(z) = \tan z + \sec z. \tag{1}$$

Le rayon de convergence de la série, $\pi/2$, est l'inverse du taux de croissance de la séquence V_n. Cette série décrit précisément la séquence des volumes, et une formule close pour V_n peut en être déduite :

$$V_{2n-1} = B_{2n}(-4)^n(1-4^n)/(2n)!\ ;\ V_{2n} = (-1)^n E_{2n}/(2n)!\ ,$$

où les B_s sont les nombres de Bernoulli et E_s les nombres d'Euler.

Les fonctions génératrices se comportent de façon naturelle par rapport aux opérations simples sur les langages temporisés (union disjointe, concaténation non-ambiguë, et étoile non-ambiguë). Cependant afin d'obtenir une caractérisation exacte pour les fonctions génératrices des langages temporisés réguliers, une analyse plus approfondie est requise. Une telle analyse constitue la principale contributions de ce chapitre. Des formules close pour les fonctions génératrices des volumes peuvent être calculées pour des sous-classe d'automates temporisés incluant ceux utilisés dans le Chapitre 8. Ces dernier permettent d'exprimer de façon combinatoire certaines classes de permutations.

Chapitre 8. Compter et générer des permutations en utilisant les langages temporisés.

La signature d'une permutation $\sigma = \sigma_1 \cdots \sigma_n$ est le mot $w = w_1 \cdots w_{n-1} \in \{\mathbf{a}, \mathbf{d}\}^{n-1}$ ou quand il y a une descente à la position i ($\sigma_i > \sigma_{i+1}$) $w_i = \mathbf{d}$ et $w_i = \mathbf{a}$ pour s'il y a une montée ($\sigma_i < \sigma_{i+1}$).

Générer toutes les permutations pour une signature donnée où simplement les compter sont deux sujets classiques de la combinatoire (voir e.g. [Szp01] et ses références).

Un exemple très étudié de permutations données par leur signature est la célèbre classe de permutation alterné (aussi appelé en zig-zag) (voir [Sta10] pour une vue d'ensemble des travaux sur cette classe). Leurs signatures appartient au langage exprimé par les expressions régulières $(\mathbf{da})^*(\mathbf{d} + \epsilon)$ (i.e. ils satisfont $\sigma_1 > \sigma_2 < \sigma_3 > \sigma_4 ...$).

Une telle définition de classe de permutations en terme de langage de signatures est en fait une nouveauté. Pour un langage $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$, nous associons la classe $\mathtt{sg}^{-1}(L)$ de permutations dont la signature est dans L. Plusieurs classes de permutations peuvent être exprimé de cette façon (e.g. les permutations en alternance, celles sans 2 descentes consécutives, celles avec un nombre pair de descentes, etc.). Nous appelons régulière ces classes de permutations.

Nous posons les deux problèmes de dénombrement et de génération aléatoire quand le langage de signature est régulier.

Nous proposons l'Algorithme 1 qui prend en entrée un langage régulier L et qui retourne une formule close pour la fonction génératrice exponentielle (EGF) de $\mathtt{sg}^{-1}(L)$ i.e. une série formelle $\sum a_n \frac{z^n}{n!}$ où le n^{eme} coefficient a_n compte les permutations de taille n avec une signature dans L. Avec une telle EGF, il est facile de retrouver le nombre a_n et de faire une estimation du taux de croissance de a_n (voir [FS09] pour une introduction complète à l'analyse combinatoire).

La génération aléatoire est faite par un algorithme décrit dans le Théorème 38. Le langage régulier de signature L associée à n la taille de la permutation à générer est donné en entrée, et les sorties sont des permutations aléatoires de taille n dont les signatures sont dans L, chaque permutation ayant une probabilité égale d'être retournée.

Ici, nous trouvons une interprétation combinatoire des fonctions génératrices de volumes des langages temporisés introduits dans le Chapitre 7. Nous développons dans ce chapitre une théorie plus pointu (et auto-contenue) car nous restreignons notre attention a des classes de langages temporisés particulières qui permettent d'encoder les classes de permutations régulières.

Le passage d'une classe de permutations à un langage temporisé se fait en deux temps. D'abord nous associons les polytopes d'ordre et de chaîne aux signatures (ceux sont des cas particulier de polytopes de Stanley associés aux ensemble partiellement ordonnés [Sta86]). Puis, nous interprétons le polytope de chaîne d'une signature w comme l'ensemble des vecteur de délais qui avec w forment un mot temporisé dans un langage temporisé bien choisi.

L'outil théorique principal est une transformation préservant les volumes entre l'union des polytopes d'ordre des mots du langage régulier considéré et le langage temporisé associé (Théorème 36). Ainsi le calcul de la fonction génératrice exponentielle des permutations se ramène a un calcul de fonction génératrice des volumes d'un langage temporisé (étudié dans le chapitre précédent). Par exemple A^{ex} schématisé sur la Figure 1.1 encode le cas bien étudié des permutations alternantes dont la fonction génératrice exponentielle est $\tan(z) + \sec(z)$. Plusieurs preuves différentes de ce résultat peuvent être trouvées dans [Sta10]. Ici, nous l'obtenons en identifiant la fonction génératrice exponentielle des permutations alternantes a la fonction génératrice des volumes de l'exemple courant décrite en (1).

Nous fournissons aussi un générateur aléatoire uniforme de mots temporisé dans un langage temporisé associé a un langage régulier de permutations (Algorithme 2). Ce générateur de mots temporisé combiné à la transformation préservant les volumes évoqué précédemment fournit un générateur de permutations aléatoires dans la classe de permutations associé au langage régulier considéré (Théorème 38).

Chapter 1

Introduction

Two theories, one object of studies Automata theory and symbolic dynamics are two well-established theories developed in parallel since the middle of the 20^{th} century, and sharing roughly the same main object of study: the sets of labels of paths in a finite labelled directed graph. In the former theory the object is a regular language and the labeled graph is called finite state automaton while in the latter theory this is known as the set of allowed blocks of a sofic shift (and the labeled graph is called a presentation of the shift). We refer the reader to the handbook chapter [BBEP10] for an exposition of symbolic dynamics in the context of automata theory.

A quantitative analysis provided by symbolic dynamics. Symbolic dynamics provides a quantitative analysis of regular languages with the notion of entropy. Entropy measures the growth rate of the languages w.r.t. the size of words considered. Its formal definition and meaning is sketched as follows. Let L_n be the set of words of length n of a regular language L. In most cases, its cardinality $|L_n|$ roughly behaves like an exponential ρ^n multiplied by a sub-exponential term. This means $\lambda^{-n} << (|L_n|\rho^{-n}) << \lambda^n$ for all $\lambda > 1$. The entropy $h(L)$ is defined as the logarithm in base 2 of ρ, in other words $|L_n|$ behaves like $2^{nh(L)}$ and formally $h(L) = \limsup_{n\to\infty} 1/n \log_2 |L_n|$. The entropy $h(L)$ has a natural interpretation as the number of bits per symbol of words of L.

Symbolic dynamics considers regular languages as dynamical systems called shift spaces and provides a topological point of view, e.g. the entropy of a sofic shift is a particular case of the so-called topological entropy defined for general dynamical systems.

From discrete to continuous time. The notion of shift and of time in the context of symbolic dynamics is discrete: shift of one letter (called one event in the following) to the right permits to go a step forward in the future while shift of one event to the left permits to go a step backward in the past. In the same way, in the verification context, words of a regular language represent the possible history of a system. Each letter represents an event and events occur one after the other without quantifying time between them.

However it is often relevant to quantify continuous time delays between events. Timed automata (TA) were introduced for this purpose in the early 90s by Alur and Dill [AD94].

TAs quickly became popular as a simple yet expressive model with a lot of decidability results such as for the model checking problem or the reachability problem (equivalent to the language emptiness problem). Since then TA have been thoroughly explored from a theoretical standpoint.

Volumetry of timed languages The notion of continuous time between events has not been explored yet in the context of symbolic dynamics. In timed automata theory a quantitative analysis of the size of timed languages and information content of timed words has only been introduced recently by two of my colleagues Eugene Asarin and Aldric Degorre. For this purpose they defined in [AD09a, AD09b] the notion of volume and entropy (inspired by symbolic dynamics) of timed languages and built a new theory that we call the *volumetry of timed languages*. In the following paragraph we briefly recall the definition of volume and entropy of timed languages.

A *timed word* is an alternating sequence $t_1 w_1 \cdots t_n w_n$ of non-negative real-valued *delays* $t_i \geq 0$ and events a_i. A *timed language* is a set of timed words. From a geometric point of view, a timed word is a vector of delays $(t_1, \ldots, t_n) \in \mathbb{R}^n$ together with a word of events $w = w_1 \cdots w_n \in \Sigma^n$ where Σ denotes the alphabet of events. We adopt the convention to write $(\vec{t}, w)$ the timed word $t_1 w_1 \cdots t_n w_n$ with $\vec{t} = (t_1, \cdots, t_n)$ and $w \in \Sigma^n$ $(n \geq 1)$. Continuing with the same convention, given a timed language $\mathbb{L} \subseteq (\mathbb{R}^+ \times \Sigma)^*$, the timed language restricted to words of length n, $\mathbb{L}'_n$ can be seen as a formal union of sets $\biguplus_{w \in \Sigma^n} \mathbb{L}_w \times \{w\}$ where $\mathbb{L}_w = \{\vec{t} \in \mathbb{R}^n \mid (\vec{t}, w) \in \mathbb{L}'\}$ is the set of delay vectors that together with w form a timed word of $\mathbb{L}$. In the sequel we will only consider languages $\mathbb{L}$ for which every $\mathbb{L}_w$ is volume measurable. To a timed language $\mathbb{L}$ one can associate its sequence of volumes:

$$\mathtt{Vol}(\mathbb{L}_n) = \sum_{w \in \Sigma^n} \mathtt{Vol}(\mathbb{L}_w)$$

and its volumetric entropy:

$$\mathcal{H}(\mathbb{L}) = \limsup_{n \to +\infty} \frac{1}{n} \log_2 \mathtt{Vol}(\mathbb{L}_n).$$

Thus the definition of the entropy of a timed language mimics that of the entropy of a regular language by replacing the counting of n-length words by the measurement of volume of n-length timed words (which are uncountable).

Throughout the thesis we will use as a running example the automaton $\mathcal{A}^{ex}$ depicted in Fig. 1.1, to illustrate the notions of volume, entropy, maximal entropy stochastic process or generating function. The timed language recognized by automaton $\mathcal{A}^{ex}$ is

$$L = \{t_1 a t_2 b t_3 a \ldots | \forall i \ (t_i + t_{i+1} \leq 1)\}.$$

For any number of events n we have a polytope in $\mathbb{R}^n$:

$$L_n = \{(t_1, t_2, t_3, \ldots, t_n) | \forall i \ (t_i + t_{i+1} \leq 1)\},$$

the sequence of volumes V_n of these polytopes is

$$1; 1; \frac{1}{2}; \frac{1}{3}; \frac{5}{24}; \frac{2}{15}; \frac{61}{720}; \frac{17}{315}; \frac{277}{8064} \ldots,$$

and it was shown in [AD09a] that this sequence behaves asymptotically like $(2/\pi)^n$ (the entropy of the language is $\log_2(2/\pi)$).

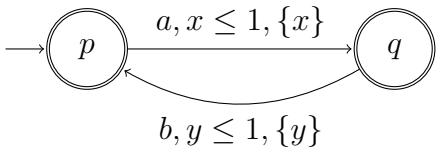

Figure 1.1: A timed automaton $\mathcal{A}^{ex}$

1.1 Contributions, extended outline

In this thesis we use the volumetry of timed language to solve several problems occurring in various domains of theoretical computer science such as verification, enumerative combinatorics or information theory. Among other we

- develop a theory of timed symbolic dynamics;
- characterize a dichotomy between bad behaving and well behaving TA;
- define a least biased stochastic process for a timed automaton;
- develop a timed theory of constrained channel coding;
- count and generate randomly and uniformly permutations in certain classes.

Our work also clearly improves the volumetry of timed languages by

- providing applications enumerated above;
- getting rid of restrictive conditions assumed in [AD09b, AD09a];
- characterizing algebraically (and algorithmically) the languages of finite entropy;
- characterizing precisely the volume sequence of a timed language with its volume generating functions;
- proving and improving the convergence of the discretization method for the computation of entropy proposed in [AD09b].
- providing several equivalent definitions of the entropy, each one having its own advantages depending on the context.

We give the framework of timed symbolics dynamics in the central chapter (Chapter 5). Even if we could have written a broad part of this thesis in terms of timed symbolic dynamics we preferred to use in most places a timed automata vocabulary. This simplifies the re-use of our contributions which are more timed automata oriented.

We summarize below our contributions in the order they appear in the thesis.

Chapter 2. Preliminaries.

In this preliminary chapter we recall the definition of volume and entropy of timed languages introduced by Asarin and Degorre. We introduce the model of timed region graph which used in most chapter of this thesis. They are the underlying structure of timed automata. We reduce the study of entropy of timed automata to that of their underlying structures called *timed region graphs*.

We introduce a machine based characterization of entropy (Proposition 5) which

- matches the original "language based" definition in case of determinism,
- is easy to compare with the entropy of a stochastic process over runs introduced latter in Chapter 4.

Chapter 3. The thin and thick alternative.

An amazing theoretical application of the entropy of timed languages is related to a well-known, but not yet sufficiently understood issue of "pathological" and "normal" behaviours of timed automata. Indeed, timed automata using exact continuous clocks, exact guards and resets are a beautiful mathematical object and a useful model of real-time systems. However, from the very beginning of research on timed automata, it was clear that they are in several aspects too precise, which leads sometimes to strange artifacts, mathematical pathologies or unrealistic models. Several lines of research have partially elucidated these issues (see the related work section below).

The state space of a timed automaton being infinite, some long (or infinite) runs never revisit the same state. For this reason, as stated in [Bea98], usual **pumping lemmata** do not hold, and should be replaced by rather involved analogues. In a run, infinitely many events can happen during a finite amount of time, or two events can happen again and again with the time interval between them tending to 0. Such a run reminds of Zeno's aporias and is often called a **Zeno run**, see [GB07] and references therein.

We introduce the key notion of *forgetful* cycles, the existence of which is a criterion for distinguishing between *thin* languages, with entropy $-\infty$, and *thick* languages, with entropy $> -\infty$.

This notion is primordial to rule out pathological TA, like we do in several chapters of this thesis (Chapters 4,5 and 6).

It turns out that:

In thin automata all the infinite trajectories, are, in some weak sense Zeno; the discretization of long trajectories is difficult, since it requires a very small discretization step.

In thick automata most of trajectories are non-Zeno and behave well under discretization; such automata have a forgetful cycle, and most trajectories visit such a cycle.

The main result of this chapter (Theorem 5) establishes that thickness of a language (i.e. that $\mathcal{H} > -\infty$) is equivalent to many other nice properties briefly described above (good discretization, existence of forgetful cycle etc.). We state at the same time that any path in a timed automaton which is thick and long, necessarily contains a forgetful cycle (Theorem 3), which can be seen as a pumping lemma for thick paths.

The proof of Theorem 5 is rather technical, and uses together with "timed" techniques inspired by [Pur00, AKY10], the monoid version of Ramsey's theory, namely Simon's factorization forests theory [Sim90].

Thickness, a necessary and sufficient condition for entropy computation. In the original work of Asarin and Degorre, two approaches were explored to compute the entropy, the discretization approach [AD09b] and the operator approach [AD09a]. However the convergence of the algorithms for both approaches was not proved and a restrictive condition on the reset of clocks along each cycle was used. In the thesis we get rid of the restrictive condition and solve the problem of convergence for the discretization approach. We also show the convergence of the operator approach in [ABD13]. Both solutions are based on the notion of forgetful cycle introduced in this chapter. More precisely the case $\mathcal{H} = -\infty$ (equivalent to the absence of a forgetful cycle) can be detected with a PSPACE procedure described in this chapter. When this case is ruled out, thickness ($\mathcal{H} > -\infty$, equivalent to the presence of a forgetful cycle) turns out to be a sufficient and necessary condition for the two methods to converge (see [BA11] and Chapter 5 for the discretization approach and [ABD13] for the operator approach).

Chapter 4. A maximal entropy stochastic process for a timed region graph.

In the context of verification of real-time system, several probability settings have been added to TA (see references below). There are several reasons to add probabilities: this permits (i) to reflect in a better way physical systems which behave randomly, (ii) to reduce the size of the model by pruning the behaviors of null probability [BBB+07], (iii) to resolve undeterminism when dealing with parallel composition [DLL+11, KBM13].

In most of previous works on the subject (see e.g. [BA07, BAM06, BBBM08, DLL+11]), probability distributions on continuous and discrete transitions are given at the same time as the timed settings. In these works, the choice of the probability functions is left to the designer of the model. Whereas, she or he may want to provide only the TA and ask the following question: what is the "best" choice of the probability functions according to the TA given? Such a "best" choice must transform the TA into a random generator of runs

the least biased as possible, i.e it should generate the runs as uniformly as possible to cover with high probability the maximum of behaviours of the modeled system. More precisely the probability for a generated run to fall in a set should be proportional to the size (volume) of this set (see [KBM13] for a same requirement in the context of job-shop scheduling). We formalize this question and propose an answer based on the notion of entropy of TA introduced in [AD09a].

The theory developed by Shannon [Sha48] and his followers permits to solve the analogous problem of quasi-uniform path generation in a finite graph. This problem can be formulated as follows: given a finite graph G, how can one find a stationary Markov chain on G which allows one to generate the paths in the most uniform manner? The answer is in two steps (see Chapter 1.8 of [Lot05] and also section 13.3 of [LM95]): (i) There exists a stationary Markov chain on G with maximal entropy, the so-called Shannon-Parry Markov chain; (ii) This stationary Markov chain allows to generate paths quasi uniformly.

In this chapter we lift this theory to the timed automata setting. We work with timed region graphs which are to timed automata what finite directed graphs are to finite state automata i.e. automata without labeling on edges and without initial and final states. We define stochastic processes over runs of timed region graphs (SPOR) and their (continuous) entropy. This generalization of Markov chains for TA has its own interest (for instance it has the nice feature to provide a continuous probability distribution on starting states). Such SPOR permits to generate step by step random runs. As a main result we describe a maximal entropy SPOR which is stationary and ergodic and which generalizes the Shannon-Parry Markov chain to TA (Theorem 14). Concepts of maximal entropy, stationarity and ergodicity can be interesting by themselves, here we use them as the key hypotheses to ensure a quasi uniform sampling (Theorem 15). More precisely the result we prove is a variant of the so-called Shannon-McMillan-Breiman theorem also known as asymptotic equipartition property (AEP).

Chapter 5. Timed symbolic dynamics.

In this chapter we provide a timed symbolic dynamics framework to the volumetry of timed languages. We introduce timed sofic shifts which are shift spaces on a compact alphabet. On the negative side, the topological entropy of such a shift space turns out to be infinite. On the positive side three measures of the size of sofic shifts can nevertheless be described: the metric mean dimension **mdim** of [LW00], the ε-entropy h_ε and volumetric entropy $\mathcal{H}$ adapted from [AD09b, AD09a]. We describe the meaning of these three measures as well as their common link. In particular we solve the problem left open in [AD09b] of the approximation of the volumetric entropy by discretization: formally we show that $h_\varepsilon = \mathcal{H} + \log_2(1/\varepsilon) + o(1)$. An interpretation of this formula in terms of Kolmogorov complexity was given in [AD09b]. This formula has also a natural interpretation in terms of information content of a timed word that we explain and use in our theory of timed channel coding (see Chapter 6).

The so-called sliding block codes are key objects in symbolic dynamics. They are by virtue of the Curtis-Hedlund-Lyndon theorem exactly the morphisms of shift spaces (i.e. continuous functions that commute with the shift). In the timed case, there is no more correspondence

between morphisms and sliding block codes (Proposition 29). However we show that every morphism to a full shift space is the uniform limit of a sequence of continuous sliding block codes (Theorem 30).

Chapter 6. Toward a timed theory of channel coding.

The theory developed in this chapter is the first attempt of lifting the classical theory of constrained-channel coding to timed languages.

Let a language S represent all the possible messages that can be generated by a *source*, and C all the messages that can transit over a *channel*. Typical problems addressed by coding theory are:

- Is it possible to transmit any source generated message via the channel?
- What would be the transmission speed?
- How to encode the message before and to decode it after transmission?

The answers given by the theory of channel coding are as follows: to each language L is associated a non-negative real number $h(L)$, called its entropy, which characterizes the quantity of information in bits per symbol. In order to transmit information in real-time (resp. with speed α) the entropy of the source should not exceed the one of the channel (also called its capacity): $h(S) \leq h(C)$ (resp. $\alpha h(S) \leq h(C)$). For regular (or more precisely sofic) languages, whenever the information inequalities above are strict, the theory of channel coding provides a transmission protocol with a simple encoding and decoding (realized by a finite-state transducer). For the practically important case when $S = \Sigma^*$ and $h(S) < h(C)$, the decoding can be made even simpler (sliding-window). A typical example is EFMPlus code [Imm95] allowing writing any binary file (i.e. the source $\{0,1\}^*$, with entropy 1) onto a DVD (the channel $C = (2,10) - \text{RLL}$ admits all the words without factors 11, 101 and 0^{11}, its entropy is 0.5418, see [Bla90]) with almost optimal rate $\alpha = 1/2$.

Classical theory of channel coding deals with discrete messages. It is, however, important to consider data words, i.e. discrete words augmented with data, e.g. real numbers. In this chapter, we develop the theory of channel coding for the most studied class of data languages: timed languages. Several models of information transmission are possible for the latter:

- the source is a timed language; the channel is a discrete language. In this case, lossless encoding is impossible, and we will consider encoding with some precision ε;
- the source and the channel are timed languages, we are interested in exact (lossless) encoding;
- the source and the channel are timed languages, some scaling of time data is allowed.

Entropy of timed languages is the key notion to address these problems. For several models of transmission of timed data we will write *information inequalities* relating entropies of

sources and channels with parameters of encodings (rate, precision, scaling, see below). Such an inequality is a necessary condition for existence of an encoding. On the other hand, whenever the information inequality holds (in its strict form) and the languages are regular (sofic), we give an explicit construction for simple timed encoding-decoding functions.

Technically, we strongly build on the quantitative discretization of timed languages done in Chapter 5.

Chapter 7. generating functions of timed languages.

Entropy is a rough size measure of a timed language. In this chapter, we make a much more precise size analysis of timed languages accepted by deterministic timed automata. We associate to such a language L the sequence of its volumes $\mathtt{Vol}(L_n)$, and the generating function $f(z) = \sum_n \mathtt{Vol}(L_n) z^n$. Thus the function $f(z)$ contains a complete information on the "size profile" of $\mathtt{Vol}(L_n)$ as a function of n. To relate it with the operator Ψ of Asarin and Degorre [AD09a], we show that $f(z)$ can be expressed in terms of its resolvent, and that the entropy of a timed language depends only on the convergence radius of $f(z)$.

The methods developed in this chapter yield for instance a closed-form expression for the generating function of volumes for the running example (automaton $\mathcal{A}^{ex}$ of Figure 1.1):

$$f(z) = \tan z + \sec z. \tag{1.1}$$

The convergence radius of the series, $\pi/2$, is the inverse of the growth rate of the sequence V_n. This series describes precisely the sequence of volumes, and a closed-form formula for V_n can be deduced:

$$V_{2n-1} = B_{2n}(-4)^n(1-4^n)/(2n)!\ ;\ V_{2n} = (-1)^n E_{2n}/(2n)!$$

where B_s stand for Bernoulli numbers and E_s for Euler numbers.

Generating functions behave in a natural way with respect to simple operations on timed languages (disjoint union, unambiguous concatenation, and unambiguous star). However in order to obtain an exact characterization for generating function of timed regular languages a more involved analysis is needed. Such an analysis constitutes the main contribution of the chapter. Closed-form expressions for the generating function of volumes can be computed for subclasses of timed automata including those used in Chapter 8 which permit to express combinatorics of some class of permutations.

Chapter 8. counting and generating permutations using timed languages.

The signature of a permutation $\sigma = \sigma_1 \cdots \sigma_n$ is the word $w = w_1 \cdots w_{n-1} \in \{\mathbf{a}, \mathbf{d}\}^{n-1}$ where in the i^{th} position $w_i = \mathbf{d}$ when σ has a descent ($\sigma_i > \sigma_{i+1}$), and $w_i = \mathbf{a}$ when it has an ascent ($\sigma_i < \sigma_{i+1}$).

Generating all the permutations with a prescribed signature or simply counting them are two classical combinatorial topics (see e.g. [Szp01] and reference therein).

A very well studied example of permutations given by their signatures are the so-called alternating (or zig-zag, or down-up) permutations (see [Sta10] for a survey). Their signatures belong to the language expressed by the regular expression $(\mathbf{da})^*(\mathbf{d}+\epsilon)$ (i.e. they satisfy $\sigma_1 > \sigma_2 < \sigma_3 > \sigma_4 ...$).

Such a definition of a class of permutations in terms of a language of signatures is in fact a novelty of the present work. To a language $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$, we associate the class $\mathtt{sg}^{-1}(L)$ of permutations whose signature is in L. Many classes of permutations can be expressed in that way (e.g. alternating permutations, those without 2 consecutive downs, those with an even number of downs, etc.).

We state and address the two problems of counting and randomly generating when the language of signatures is regular.

We propose Algorithm 1 which takes as its input a regular language L and returns a closed form formula for the exponential generating function (EGF) of $\mathtt{sg}^{-1}(L)$ i.e. a formal power series $\sum a_n \frac{z^n}{n!}$ where the n^{th} coefficient a_n counts the permutations of length n with signature in L. With such an EGF, it is easy to recover the number a_n and some estimation of the growth rate of a_n (see [FS09] for an overview of analytic combinatorics).

The random generation is done by an algorithm described in Theorem 38. The regular language of signatures L together with n the size of permutation to generate are given in input, and the output are random permutations of size n whose signatures are in L with equal probability to be returned.

Here we find a combinatorial interpretation to the volume generating function of timed languages introduced earlier in Chapter 7. We develop in this chapter a self contained and more acute theory since we restrict our attention to a simple class of timed languages that permit to encode regular classes of permutations. The passage from a class of permutations to a timed language is in two steps. First we associate order and chain polytopes to signatures which are particular cases of Stanley's poset polytopes [Sta86]. Then we interpret the chain polytopes of a signature w as the set of delays which together with w form a timed word of a well chosen timed language.

The main theoretical tool is a volume preserving transformation between the union of order polytopes of the words in the regular language considered and the timed language associated (Theorem 36). Hence the computation of the exponential generating function of permutations reduce to that of the generating function of volumes of a timed language (studied in the previous chapter). For instance, the running example A^{ex} depicted on Figure 1.1 encodes the well studied class of alternating permutations whose exponential generating function is known to be $\tan(z) + \sec(z)$. Several different proofs of this results can be found in [Sta10]. Here we get another proof by identifying the exponential generating function of the alternating permutations to the generating function of volumes of the running example (already described in 1.1).

Last but not least, we provide a uniform sampler of timed word for any regular timed language associated to a regular language of signature (Algorithm 2). This sampler of timed word combined with the volume preserving transformation mentioned above provide a uniform sampler of permutations in the class associated to the regular language considered.

1.2 Related work

We have already cited several times the two pioneering articles on volumetry of timed language [AD09a, AD09b]. We should also mention the work [AD10] done by the same authors where a mean dimension and an entropy were introduced for timed language for which the usual volume is not defined.

1.2.1 Classical results lifted to the timed case.

A broad part of our work consists in lifting to the timed case of results coming from information theory, symbolic dynamics and coding theory. A lot of results concerning entropy were already presented in the seminal paper of Shannon [Sha48].

Symbolic dynamics

A well established reference for a general introduction of symbolic dynamics and coding is [LM95]. We refer also the reader to the more recent yet unpublished chapter "Symbolic dynamics" of the handbook of automata theory [BBEP10] where an effort is made to present symbolic dynamics in the context of automata theory.

Mean dimension and compact alphabet shift spaces

In [LW00], Lindenstrauss and Weiss introduced the two concepts of metric mean dimension and topological mean dimension to measure shift spaces on compact alphabet such as $[0,1]^{\mathbb{Z}}$. Here we study metric mean dimension for timed sofic shift. As mentioned above, a mean dimension has already been introduced in the context of volumetry of timed language [AD10]. Yet it does not capture phenomena explained in Chapter 5. We briefly explain the differences with our approach in the conclusion of Chapter 5.

Constrained channel coding

Constrained-channel coding theory for finite alphabets is a well-established domain; we refer the reader to monographs [LM95, Bla90, BPR09], handbook chapters [MRS98, BBM$^+$10] and references therein.

Shannon-Parry maximal entropy Markov chain

One of the main contributions of the thesis (Chapter 4) is the lifting to the timed setting of the theory of maximal entropy Markov chain of a finite graph developed by Shannon and Parry.

In the discrete case the theory of Perron and Frobenius is used as a key tool to define the maximal entropy Markov chain of a graph (see e.g. [LM95, Lot05]). In the continuous case, we use its generalization to positive operators on functions defined on a "continuous" state space. All the results on positive operators we use can be found in [KLS89].

1.2.2 Timed automata related works

Avoiding pathological behaviour in TAs

In order to rule out bad behaviors, restricted classes of timed automata, and alternative semantics were considered by several authors. Thus, in [GHJ97, HR00], a **tube language** semantics is introduced. In a pioneering paper [Pur00] a **robust semantics**, based on small imprecisions is considered. It reappears in a different flavour as **implementability**, see [WDR05, WDMR08], and in another version in [AKY10].

With the same objective to rule out bad behaviors, restrictions are often put on all the cycles in the automaton, by requiring that each cycle takes at least one time unit (strongly non-Zeno condition), or resets all the clocks (progress cycle condition), or even resets all the clocks at one and the same transition (regeneration condition). This kind of conditions intervenes in most of the cited literature – and will be replaced in several places of this thesis (as we have already mentioned) by the somewhat subtler condition of existence of a forgetful cycle.

The notion of forgetfulness we introduced in [BA11], has been used by other authors in the context of frequency analysis [Sta12] and robust controller synthesis for timed automaton [SBMR13]. In the latter work the existence of a forgetful cycle (equivalent to thickness) is a necessary and sufficient condition for a robust controller synthesis to be achievable.

Probabilistic settings for real time systems

The models of stochastic real-time systems we introduce in Chapter 4 can be related to numerous previous works. Almost-sure model checking for probabilistic real-time systems based on generalized semi Markov processes GSMPs was presented in [ACD91] at the same time as the timed automata theory and by the same authors. This work was followed by [BAM06, BA07] which address the problem of quantitative model checking for GSMPs under restricted hypotheses. The GSMPs have several differences with TA; roughly they behave as follows: in each location, clocks decrease until a clock is null, at this moment an action corresponding to this clock is fired, the other clocks are either reset, unchanged or purely canceled. Our probability setting is more inspired by [BBB$^+$07, BBJM12, DLL$^+$11] where probability densities are added directly on the TA. Here we add the new feature of an initial probability density function on states.

In [DLL$^+$11], a probability distribution on the runs of a network of priced timed automaton is implicitly defined by a race between the components, each of them having its own probability. This allows a simulation of random runs in a non deterministic structure without state space explosion. There is no reason that the probability obtained approximates uniformness and thus it is quite incomparable to our objective.

Polytopes and discretization

For volume definition and discretization of timed language we use the fact that constraints linking successive delays that can be read along a path lies in a polytopes. This fact is quite

folklore in timed automata community and was related to discretization in several previous works [MP04, HMP92, AMP98, Krc09, AD09b].

Polytopes and discretization has been also studied in a combinatorics context. In particular, we will use the beautiful theorem of Erhrhart for counting discrete points in polytopes (see [BR07, BLD+05]).

1.2.3 Combinatorics

The generating functions of Chapter 7 generalize those of regular languages, thoroughly studied and applied, [BR11, FS09, SS78].

The link between geometry and permutations used in Chapter 8 is not new and can be found in several articles including [EJ12, Mar12, Szp01] that state integral equations similar to ours. Our use of timed languages provides a new tool necessary to catch the dynamic of the regular languages of signatures. Particular regular languages of signatures are considered in [EJ12] under the name of consecutive descent pattern avoidance.

Numerous other works treat more general cases of (consecutive) pattern avoidance (see e.g. the monograph [Kit11]) and are quite incomparable to our works. Indeed, certain classes of permutations avoiding a finite set of patterns cannot be described as a language of signatures while some class of permutations involving regular languages cannot be described by finite pattern avoidance (e.g. the permutations with an even number of descents).

The random sampler of timed words we introduce at the end of Chapter 8 is an adaptation to the timed case of the so-called recursive method of [NW78] and developed by [FZVC94]. It has been successively improved for generation of words in a regular language in [BG12].

Last but not least, we refer the reader to [Sta10] for a broad survey on combinatorics of the class of alternating permutations (treated as an example in the present thesis). In [Mar12], the author exposes a method to generate uniformly alternating permutations in time $O(n \log_2 n)$ which is faster than ours. However our method is more generic as it works for every regular languages.

1.3 Past and ongoing publications

A broad part of this thesis has been published in several conferences or is under submission for conference or journal.

Published conference papers

- [BA11] corresponds to Chapter 3 with the discretization of the entropy (Theorem 25) of Chapter 5;
- [ABDP12] corresponds to Chapter 7;
- [ABB+12] corresponds to Chapter 6;

- [Bas13b] corresponds to Chapter 4. This article has obtained the Best student paper award at ICALP 2013.
- [Basar] corresponds to Chapter 8. A long version is available here [Bas13a].

Papers under submission

- A shorten version of Chapter 5 is under submission to a conference.
- Chapter 4 corresponding to [Bas13b] is under submission for the ICALP 2013 special issue of *Information and Computation*.

Chapter 2

Preliminaries

Outlines of the chapter

In this thesis we continue the work of Asarin and Degorre on volumetry of timed languages [AD09a, AD09b]. This theory has been designed for bounded deterministic timed automata (BDTA). In section 2.1, we recall first the definitions of timed languages and BDTAs and then define a simplified version of BDTA called timed region graph. We will consider this model in most chapters of this thesis (Chapter 3, 4, 5, 6) while in the two remaining chapter (Chapter 7 and 8) we give a self contained definition of timed automata based on clock languages [BP02].

In section 2.2 we give advanced preliminaries. In section 2.3 we explain how to pass from BDTA to TRGs without changing the entropy. This last technical section can be skipped in a first reading as its material is not used in the rest of the thesis.

2.1 Basics definitions

2.1.1 Timed languages

An alphabet of *timed events* is the product $\mathbb{R}^+ \times \Sigma$ where Σ is a finite alphabet. The meaning of a timed event (t_i, w_i) is that t_i is the time *delay* before the occurrence of the *event* w_i. A *timed word* is a sequence of timed events with the intuition that they occur one after the other. From a geometric point of view, a timed word is a vector of delays $(t_1, \dots, t_n) \in \mathbb{R}^n$ together with a word of events $w = w_1 \cdots w_n$. We will often abuse the notation and write $(\vec{t}, w)$ the timed word $(t_1, w_1) \cdots (t_n, w_n)$ with $\vec{t} = (t_1, \cdots, t_n)$ and $w \in \Sigma^n$ $(n \geq 1)$. A *timed language* is just a set of timed words.

Volume and entropy For every timed language $\mathbb{L} \subseteq \Sigma^*$ and word of events $w \in \Sigma^*$, we define $\mathbb{L}_w = \{\vec{t} \mid (\vec{t}, w) \in \mathbb{L}\}$. A timed language $\mathbb{L}$ can be viewed as a formal sum over $w \in \Sigma^*$ of sets $\mathbb{L}_w \subseteq \mathbb{R}^{|w|}$. We denote by $\mathbb{L}_n$ the restriction of $\mathbb{L}$ to n-length timed words: $\mathbb{L}_n \cong \sum_{w \in \Sigma^n} \mathbb{L}_w \times \{w\}$. Its n^{th} *volume* $V_n(\mathbb{L})$ is the sum over $w \in \Sigma^n$ of $\texttt{Vol}(\mathbb{L}_w)$ where $\texttt{Vol}$

stands for volume measure[1] (also known as Lebesgue measure):

$$V_n(\mathbb{L}) = \sum_{w \in \Sigma^n} \mathtt{Vol}(\mathbb{L}_w).$$

The volumetric entropy of $\mathbb{L}$ is

$$\mathcal{H}(\mathbb{L}) = \limsup_{n \to +\infty} \frac{1}{n} \log_2 V_n(\mathbb{L}). \tag{2.1}$$

Timed languages considered in this thesis are mostly those recognized by bounded deterministic timed automata we define now.

2.1.2 Bounded Deterministic Timed Automata

A good introduction to timed automata theory is its founding paper [AD94]. Here we give self contained definitions and explanations. We fix a natural constant M which upper bounds all the constants in the automaton. A *clock* is a variable ranging over $\mathbb{R}_{\geq 0}$ (non-negative reals). A *clock constraint* over a set of clocks C is a conjunction of finitely many inequalities of the form $x^i \sim c$ or $x^i \sim x^j + c$, where x^i and x^j are clocks, $\sim \in \{<, \leq, =, \geq, >\}$ and $c \in \{0, \ldots, M\}$. A subset of $\mathbb{R}^C$ defined by a clock constraint is called a *zone*.

A timed automaton (TA) is a tuple $\mathcal{A} = (Q, \Sigma, C, \Delta, I, F)$ where

- C is a finite set of clocks;
- Q is a finite set of *locations*;
- The *states* of $\mathcal{A}$ are couples of a location and a clock vector: $\mathbb{S}_{\mathcal{A}} =_{\text{def}} Q \times [0, M]^C$;
- Σ is a finite alphabet of events;
- Δ is a finite set of *transitions*. Any transition $\delta \in \Delta$ goes from a *starting location* $\delta^- \in Q$ to an *ending location* $\delta^+ \in Q$; δ has
  - a set $\mathfrak{r}(\delta)$ of indices of clocks to reset when firing it,
  - a guard $\mathfrak{g}(\delta)$ (which is a clock constraint) to satisfy to fire it,
  - a label $\mathtt{Lab}(\delta)$;
- I is the set of initial states described for each location by a zone I_q: $(q, \vec{x}) \in F$ iff $\vec{x} \in I_q$.
- F is the set of final states described for each location by a zone F_q: $(q, \vec{x}) \in F$ iff $\vec{x} \in F_q$.

[1] Such a volume measure exists when $\mathbb{L}_w$ is a union of polytopes. This will always be the case for timed languages considered in this thesis.

A *timed transition* is an element (t, δ) of $\mathbb{A} =_{\text{def}} [0, M] \times \Delta$ (the *delay* t represents the time before firing the transition δ). A timed word $(t_1, a_1) \cdots (t_n, a_n)$ is the label of $(t_1, \delta_1) \cdots (t_n, \delta_n) \in \mathbb{A}^n$ (another timed word, sequence of timed transitions) if for all $i \in \{1, \ldots, n\}$, $\mathtt{Lab}(\delta_i) = a_i$.

Given a state $s = (q, \vec{x}) \in \mathbb{S}_{\mathcal{A}}$ and a timed transition $\alpha = (t, \delta) \in \mathbb{A}$, the *successor* of s by α is denoted by $s \triangleright \alpha$ and defined as follows. Let $\vec{x}'$ be the clock vector obtained from $\vec{x} + (t, \ldots, t)$ by resetting clocks in $\mathfrak{r}(\delta)$ ($x'_i = 0$ if $i \in \mathfrak{r}(\delta)$, $x'_i = x_i + t$ otherwise). If $\delta^- = q$ and $\vec{x} + (t, \ldots, t)$ satisfies the guard $\mathfrak{g}(\delta)$ then $s \triangleright \alpha = (\delta^+, \vec{x}')$ else $s \triangleright \alpha = \perp$. Here and in the rest of the thesis $\perp$ represents every undefined state. Timed words of $\mathbb{A}^*$ act on states $\mathbb{S} \cup \{\perp\}$ as follows $\perp \triangleright \alpha = \perp$, $s \triangleright \varepsilon = s$ and $s \triangleright (\alpha \vec{\alpha}') = (s \triangleright \alpha) \triangleright \vec{\alpha}'$ for all $s \in \mathbb{S}$, $\alpha \in \mathbb{A}$, $\vec{\alpha}' \in \mathbb{A}^*$.

A *run* of the TA $\mathcal{A}$ is a word $s_0 \alpha_0 \cdots s_n \alpha_n \in (\mathbb{S}.\mathbb{A})^{n+1}$ such that $s_0 \triangleright \alpha_0 \alpha_1 \cdots \alpha_n \neq \perp$ (i.e. $s_{i+1} = s_i \triangleright \alpha_i \neq \perp$ for all $i \in \{0, \ldots, n-1\}$ and $s_n \triangleright \alpha_n \neq \perp$). Its label is that of $\alpha_0 \cdots \alpha_n$. The run and its label are called accepted by $\mathcal{A}$ if $s_0 \in I$ and $s_n \triangleright \alpha_n \in F$. The timed language of $\mathcal{A}$ denoted by $\mathbb{L}(\mathcal{A})$ is the set of its accepted time words. We denote by $\mathbb{L}_n(\mathcal{A})$ the language $\mathbb{L}(\mathcal{A})$ restricted to its n-length timed words and by $\mathcal{R}_n(\mathcal{A})$ the set of its n-length runs ($n \geq 1$).

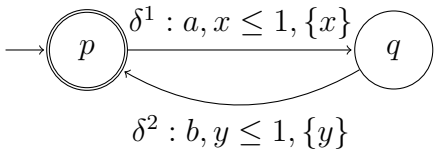

Figure 2.1: A bounded deterministic timed automaton $\mathcal{A}$

Example 1. *Consider the timed automaton depicted in figure 2.1 with* $I_p = (0, 0)$, $I_q = \emptyset$, $F_p = [0, 1] \times \{0\}$, $F_q = 0$. *An accepted run of this timed automaton is*

$$(p, (0, 0))(0.3, \delta_1)(q, (0, 0.3))(0.6, \delta_2)(p, (0.6, 0))(0.2, \delta_1)(q, (0, 0.2))(0.7, \delta_2)$$

The language of this timed automaton can be simply describe as follows. For every $n \in \mathbb{N}$*:*

$$\mathbb{L}_{2n}(\mathcal{A}) = \{(t_1, a)(t_2, b) \cdots (t_{2n-1}, a)(t_{2n}, b) \mid t_i + t_{i+1} \leq 1 \text{ for } i \leq 2n - 1\}$$

and $\mathbb{L}_{2n+1}(\mathcal{A}) = \emptyset$.

A TA is *deterministic* if for any two transitions with the same source and the same label the guards are disjoint.

The volumetry of timed languages ([AD09a, AD09b]) has been developed for deterministic timed automata with bounded clocks (BDTA). If some guards in the automaton were unbounded, the volume would be infinite, which is beyond the reach of our approach. Determinism was motivated as follows in [AD09a, AD09b]:

Remark 1. *Most of known techniques to compute entropy of untimed regular languages work on deterministic automata. In fact, these techniques count paths in the automaton, and only in the deterministic case their number coincides with the number of accepted words. The same is true for volumes in timed automata.*

2.1.3 Timed region graphs

Smallest (by inclusion) zones are called *regions* (e.g. the set of points (x_1, x_2, x_3, x_4) which satisfy the constraints $0 = x_2 < x_3 - 4 = x_4 - 3 < x_1 - 2 < 1$). Regions of $[0,1]^2$ are depicted in Figure 2.2.

Informally, timed region graph is to a timed automaton what a graph is to an automaton: an automaton without initial, final states as well as labels on transitions. Timed region graphs admit also a useful decomposition in regions, making them a timed version of the so-called region graphs [AD94].

Formally, a *timed region graph* (TRG) is a tuple $\mathcal{G} = (C, Q, \mathbb{S}, \Delta)$ such that

- C is a finite set of clocks.
- Q is a finite set of *locations*.
- $\mathbb{S}$ is the set of *states* which are couples of a location and a clock vector ($\mathbb{S} \subseteq Q \times [0, M]^C$). It admits a region decomposition $\mathbb{S} = \cup_{q \in Q} \{q\} \times \mathbf{r}_q$ where for each $q \in Q$, $\mathbf{r}_q$ is a region called *entry* region of q.
- Δ is a finite set of *transitions*. Any transition $\delta \in \Delta$ goes from a *starting location* $\delta^- \in Q$ to an *ending location* $\delta^+ \in Q$; it has a set $\mathfrak{r}(\delta)$ of clocks to reset when firing δ and a guard $\mathfrak{g}(\delta)$ to satisfy to fire it. Moreover, the set of clock vectors that satisfy $\mathfrak{g}(\delta)$ is projected on the region $\mathbf{r}_{\delta^+}$ when the clocks in $\mathfrak{r}(\delta)$ are resets.

It can be seen as a timed automaton for which $I_q = F_q = \mathbf{r}_q$ for every $q \in Q$ and the labelling function `Lab` is the identity on Δ.

Runs of timed region graphs are defined as for timed automata. The timed language of a timed region graph is just the set of labels of all its runs.

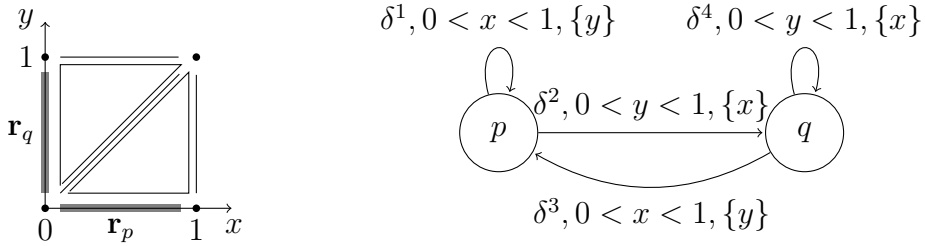

Figure 2.2: Right: $\mathcal{G}^{\texttt{ex1}}$; left: Its state space (in gray).

Example 2. *Let $\mathcal{G}^{\texttt{ex1}}$ be the timed region graph depicted in Figure 2.2 with $\mathbf{r}_p$ and $\mathbf{r}_q$ the region described by the constraints $0 = y < x < 1$ and $0 = x < y < 1$ respectively. Successor action is defined by $[p, (x, 0)] \triangleright (t, \delta^1) = [p, (x+t, 0)]$ and $[p, (x, 0)] \triangleright (t, \delta^2) = [q, (0, t)]$ if $x + t < 1$; $[q, (0, y)] \triangleright (t, \delta^3) = [p, (t, 0)]$ and $[q, (0, y)] \triangleright (t, \delta^4) = [q, (0, y + t)]$ if $y + t < 1$. An example of run of $\mathcal{G}^{\texttt{ex1}}$ is $(p, (0.5, 0))(0.4, \delta^1)(p, (0.9, 0))(0.8, \delta^2)(q, (0, 0.8))(0.1, \delta^3)(p, (0.1, 0))$.*

A transition is *fleshy* if its guard $\mathfrak{g}(\delta)$ has no constraints of the form $x = c$ in its definition. A timed region graph is *fleshy* if so does its transitions. For instance, the timed region graph depicted in figure 2.2 is fleshy.

Proposition 1 (adapted from [AD09a] to TRGs)**.** *Given a TRG $\mathcal{G}$, a fleshy TRG $\mathcal{G}'$ accepting a language $\mathbb{L}(\mathcal{G}') \subset \mathbb{L}(\mathcal{G})$ with $V_n(\mathbb{L}(\mathcal{G}')) = V_n(\mathbb{L}(\mathcal{G}))$ and $\mathcal{H}(\mathbb{L}(\mathcal{G}')) = \mathcal{H}(\mathbb{L}(\mathcal{G}))$ can be constructed.*

Most of the time, TRGs considered in this thesis are fleshy and we will sometimes consider their closed version $\overline{\mathcal{G}}$ (see section 2.2.2 below).

2.2 Advanced preliminaries

Reduced version of runs

The reduced version of a run $s_0\alpha_0 \cdots s_n\alpha_n \in (\mathbb{S} \times \mathbb{A})^{n+1}$ is $[s_0, \alpha_0 \dots \alpha_n] \in \mathbb{S} \times \mathbb{A}^{n+1}$ (for all $i > 0$ the state $s_i = s_{i-1} \triangleright \alpha_{i-1}$ is determined by its preceding states s_{i-1} and timed transition α_{i-1} and thus is a redundant information). In the following we will use without distinction extended and reduced version of runs.

2.2.1 Paths, polytopes and point to point reachability.

Throughout the thesis we will use geometrical ideas involving polytopes (especially those associated to paths).

polytopes A *closed (resp. open) affine half-space* of $\mathbb{R}^n$ is a set of vectors of the form $\{\vec{t} | \left\langle \vec{h}, \vec{t} \right\rangle \geq a\}$ where $\vec{h} \in \mathbb{R}^n$, $a \in \mathbb{R}$ and $\langle , \rangle$ denotes the standard scalar product of $\mathbb{R}^n$. An affine hyperspace of $\mathbb{R}^n$ is a set of the form $\{\vec{t} | \left\langle \vec{h}, \vec{t} \right\rangle = a\}$. We recall that a *polytope* of $\mathbb{R}^n$ is a bounded set which is the intersection of a finite number of affine closed half-spaces. When the half-spaces are open we speak of *open polytope*. We call an *open non full dimensional polytope* the intersection of an open polytope with several affine sub-spaces.

A polytope can equivalently be defined as the convex hull of a finite number of points in $\mathbb{R}^n$. Worth mentioning examples of polytopes are simplices which are the convex hull of $n + 1$ affinely independent points in $\mathbb{R}^n$. In particular we call a simplex of type 1 a set of the following form $\{\vec{t} \in \mathbb{R}^n_{\geq 0} \mid t_1 + \ldots + t_n \leq 1\}$. It is the convex hull of the origin $(0, \ldots, 0)$ and of the standard basis of $\mathbb{R}^n$: $(1, \ldots, 0), \ldots, (0, \ldots, 1)$. A simplex of type 2 is a set of the form $\{\vec{T} \in \mathbb{R}^n_{\geq 0} \mid 0 \leq T_1 \leq \ldots \leq T_n \leq 1\}$. It is the convex hull of $(0, \ldots, 0), (0, \ldots, 0, 1)(0, \ldots, 0, 1, 1), \ldots, (1, \ldots, 1)$. Polytopes are n-dimensional generalization of convex polyhedra e.g. simplices generalize tetrahedra.

Polytopes associated to a path Here we describe for a path $\pi = \pi_1 \cdots \pi_n \in \Delta^n$ $(n \geq 1)$ the geometrical shape of constraints on clocks and delays involved in runs along π. For

$j \in \{1, \ldots, |C|\}$, we denote by x_i^j the value of the j^{th} clock before the $(i+1)^{th}$ transition of the run. It is easy to see that x_i^j is of the following form:

$$\begin{array}{ll} x_i^j & = x_0^j + \sum_{k=1}^{i} t_k \qquad \text{if } x^j \text{ is not reset during } \pi_1 \cdots \pi_i; \\ & = \sum_{\mathtt{lr}(x^j,i)}^{i} t_i \qquad \text{if } \mathtt{lr}(x^j, i) \text{ is the index of the last reset of } x^j \text{ before } i. \end{array}$$

Let π be a path from a location q to a location q', $\vec{t} \in [0, M]^n$. We denote by $\vec{x} \triangleright_\pi \vec{t} = \vec{x}'$ the fact that $(q, \vec{x}) \triangleright (\vec{t}, \pi) = (q', \vec{x}')$.

Lemma 1. *For every path π the set $\{\vec{x}_0, \vec{t}, \vec{x}_n \mid \vec{x}_0 \triangleright_\pi \vec{t} = \vec{x}_n\}$ is an open non full dimensional polytope (and so it is convex).*

Fixing the starting and ending clock vectors $\vec{x}_0, \vec{x}_n$ as parameters we get an open polytope $P_\pi(\vec{x}_0, \vec{x}_n) = \{\vec{t} \mid \vec{x}_0 \triangleright_\pi \vec{t} = \vec{x}_n\}$. If we are not interested in clock values at the end of the path, we get an open polytope depending only on the path and clock values at the beginning of the path.

$$P_\pi(\vec{x}_0) = \{\vec{t} \mid \exists \vec{x}_n \text{ s.t. } \vec{x}_0 \triangleright_\pi \vec{t} = \vec{x}_n\}$$

By projecting also the clock values at the beginning of the path we obtain the open polytope:

$$P_\pi = \{\vec{t} \mid \exists \vec{x}_0, \vec{x}_n \text{ s.t. } \vec{x}_0 \triangleright_\pi \vec{t} = \vec{x}_n\}.$$

The other way around, if we do not care about timing, we get the reachability predicate:

$$\mathtt{Reach}(\pi) = \{(\vec{x}_0, \vec{x}_n) \mid \exists \vec{t} \text{ s.t. } \vec{x}_0 \triangleright_\pi \vec{t} = \vec{x}_n\}.$$

A *contiguous polytope* is a bounded subset of $\mathbb{R}^n$ which is composed by all the points satisfying a conjunction of inequalities of the form $\sum_{i=j}^{k} t_i \in (A, B)$ for some $1 \leq i \leq j \leq n$ and $A, B \in \mathbb{N}$. We say that the polytope is *d-contiguous* if the length of all sums in the inequalities is bounded by d. They are associated to *d-progressive* paths: the paths along which each clock is reset at least every d transitions.

Proposition 2. *For each path $\pi \in \Delta^*$, P_π is a contiguous polytope. If π is d-progressive then P_π is a d-contiguous polytope.*

Polytopes associated to paths and language of a TRG

The language of a timed region graph $\mathcal{G}$ can simply be decomposed as a formal union of open polytopes P_π $\mathbb{L}(\mathcal{G}) = \cup_{\pi \in \Delta^*} P_\pi \times \{\pi\} = \{(\vec{t}, \pi) \mid \vec{t} \in P_\pi\}$.

In the following V_π denotes the volume of the polytope P_π: $V_\pi =_{\text{def}} \mathtt{Vol}(P_\pi)$. By summing over all paths π of length n we obtain the volume of $\mathbb{L}_n(\mathcal{G})$: $V_n(\mathcal{G}) =_{\text{def}} \sum_{\pi \in \Delta^n} V_\pi$.

2.2.2 Closed version of a timed region graph

We sometimes need to work with closed sets of delays (e.g. to ensure compactness in Chapter 5). Hence we also consider the closed version $\overline{\mathcal{G}}$ of a timed region graph $\mathcal{G}$, which is constructed by replacing every region $\mathbf{r}$ appearing in the definition of $\mathcal{G}$ by its topologic closure (in $\mathbb{R}^d$) $\overline{\mathbf{r}}$.

If we consider $\mathcal{G}$ and $\overline{\mathcal{G}}$ as labelled graphs, it is easy to see that this transformation is an isomorphism, and thus both automata have exactly the same discrete paths. For a path π of $\mathcal{G}$, we denote by $\overline{\pi}$ its isomorphic image in $\overline{\mathcal{G}}$. Then the following holds:

Proposition 3. *For a timed region graph $\mathcal{G}$,*

1. *for every $n \in \mathbb{N}$, $V_n(\mathcal{G}) = V_n(\overline{\mathcal{G}})$ and thus $\mathcal{H}(\mathbb{L}(\mathcal{G})) = \mathcal{H}(\mathbb{L}(\overline{\mathcal{G}}))$;*
2. *for every path π of $\mathcal{G}$, $\mathtt{Reach}(\overline{\pi}) = \overline{\mathtt{Reach}(\pi)}$.*

2.2.3 SCC decomposition

Given a timed region graph $\mathcal{G}$, it can be split into (non-trivial) strongly-connected components (SCC) $\mathcal{G}_i$ and acyclic pathways between them. The entropy of a TRG depends in a very natural way on the entropies of its SCC.

Proposition 4. *The entropy of $\mathcal{G}$ equals the maximal entropy of its SCC.*

Proof. Let us denote by $\mathcal{H}_{\max}$ the maximal entropy of the SCC and let it be reached on $\mathcal{G}_{i_{\max}}$. As $\mathbb{L}(\mathcal{G}_{i_{\max}})$ is a sublanguage of $\mathbb{L}(\mathcal{G})$, we can conclude that $\mathcal{H}_{\max} \leq \mathcal{H}(\mathcal{G})$.

For the converse inequality let us fix some $\sigma > 0$. By definition of entropy, there exists an $A > 0$ such that in each subautomaton $\mathcal{G}_i$ we have for all n:

$$\mathtt{Vol}(\mathbb{L}_n(\mathcal{G}_i)) \leq A 2^{n(\mathcal{H}_{\mathcal{G}_i}+\sigma)} \leq A 2^{n(\mathcal{H}_{\max}+\sigma)}.$$

One can decompose $\mathbb{L}_n$ as a finite disjoint union of languages of the form

$$\mathbb{L}_{n_1}(\mathcal{G}_{i_1}) \cdot \mathbb{L}^{i_1 \to i_2} \cdot \mathbb{L}_{n_2}(\mathcal{G}_{i_2}) \cdots \mathbb{L}_{n_m}(\mathcal{G}_{i_m}). \tag{2.2}$$

Languages $\mathbb{L}^{i_j \to i_{j+1}}$ correspond to all paths going from $\mathcal{G}_{i_j}$ to $\mathcal{G}_{i_{j+1}}$. There are finitely many such paths, thus their lengths and volumes are globally bounded (respectively by some L and D. In particular, in the decomposition (2.2) only a bounded length is spent outside the SCCs: $n - c \leq n_1 + \cdots + n_m \leq n$ (for $c = mL$).

Thus, the volume V of a language described in (2.2) satisfies

$$V \leq A' 2^{(n_1+\cdots+n_m)(\mathcal{H}_{\max}+\sigma)} \leq A'' 2^{n(\mathcal{H}_{\max}+\sigma)}$$

Summing over elements of the finite disjoint union of languages gives the inequality $\mathtt{Vol}(\mathbb{L}_n(\mathcal{G})) \leq A''' 2^{n(\mathcal{H}_{\max}+\sigma)}$ and then, since σ is arbitrary, $\mathcal{H}_{\max} \geq \mathcal{H}(\mathcal{G})$. □

2.2.4 Volume and entropy of runs

One of the main contributions of the thesis is the design of a maximal entropy stochastic process over runs of a timed region graph (Chapter 4). Definition of such a process and its entropy requires to know how to integrate (probability density) functions over states and runs. Moreover to compare the entropy of the process with that of the graph, it is necessary to give a run based characterization of this latter entropy (Proposition 5).

Integrating over states and runs; volume of runs.

It is well known (see [AD94]) that a region is uniquely described by the integer parts of clocks and by an order on their fractional parts, e.g. in the region $\mathbf{r}^{\text{ex}}$ given by the constraints $0 = x_2 < x_3 - 4 = x_4 - 3 < x_1 - 2 < 1$, the integer parts are $\lfloor x_1 \rfloor = 2, \lfloor x_2 \rfloor = 0, \lfloor x_3 \rfloor = 4, \lfloor x_4 \rfloor = 3$ and fractional parts are ordered as follows $0 = \{x_2\} < \{x_3\} = \{x_4\} < \{x_1\} < 1$. We denote by $\gamma_1 < \gamma_2 < \cdots < \gamma_d$ the fractional parts different from 0 of clocks of a region $\mathbf{r}_q$ (d is called the dimension of the region). In our example the dimension of $\mathbf{r}^{\text{ex}}$ is 2 and $(\gamma_1, \gamma_2) = (x_3 - 4, x_1 - 2)$. We denote by Γ_q the simplex $\Gamma_q = \{\vec{\gamma} \in \mathbb{R}^d \mid 0 < \gamma_1 < \gamma_2 < \cdots < \gamma_d < 1\}$. The mapping $\phi_{\mathbf{r}} : \vec{x} \mapsto \vec{\gamma}$ is a natural bijection from the d dimensional region $\mathbf{r} \subset \mathbb{R}^{|X|}$ to $\Gamma_q \subset \mathbb{R}^d$. In the example the pre-image of a vector (γ_1, γ_2) is $(\gamma_2 + 2, 0, \gamma_1 + 4, \gamma_1 + 3)$. We will often abuse the notation and write $\vec{\gamma}$ instead of $\phi^{-1}_{\mathbf{r}_q}(\vec{\gamma})$ e.g. in polytopes $P_\pi(\vec{\gamma}), P_\pi(\vec{\gamma}, \vec{\gamma}')$.

Example 3 (Continuing example 2)**.** *The region* $\mathbf{r}_p = \{(x, y) \mid 0 = y < x < 1\}$ *is 1-dimensional,* $\phi_{\mathbf{r}_p}(x, y) = x$ *and* $\phi^{-1}_{\mathbf{r}_p}(\gamma) = (\gamma, 0)$.

Now, we introduce simplified notation for sums of integrals over states, transitions and runs. We define the integral of an integrable[2] function $f : \mathbb{S} \to \mathbb{R}$ (over states):

$$\int_{\mathbb{S}} f(s)ds = \sum_{q \in Q} \int_{\Gamma_q} f(q, \phi^{-1}_{\mathbf{r}_q}(\vec{\gamma}))d\vec{\gamma}.$$

where $\int .d\vec{\gamma}$ is the Lebesgue integral (wrt. the Lebesgue measure). We define the integral of an integrable function $f : \mathbb{A} \to \mathbb{R}$ (over timed transitions):

$$\int_{\mathbb{A}} f(\alpha)d\alpha = \sum_{\delta \in \Delta} \int_{[0,M]} f(t, \delta)dt$$

and the integral of an integrable function $f : \mathcal{R}_n \to \mathbb{R}$ (over runs) with the convention that $f[s, \vec{\alpha}] = 0$ if $s \triangleright \vec{\alpha} = \bot$:

$$\int_{\mathcal{R}_n} f[s, \vec{\alpha}]d[s, \vec{\alpha}] = \int_{\mathbb{S}} \int_{\mathbb{A}} \dots \int_{\mathbb{A}} f[s, \vec{\alpha}]d\alpha_1 \dots d\alpha_n ds$$

To summarize, we take finite sums over finite discrete sets Q, Δ and take integrals over dense sets Γ_q, $[0, M]$. More precisely, all the integrals we define have their corresponding

[2] A function $f : \mathbb{S} \to \mathbb{R}$ is integrable if for each $q \in Q$ the function $\vec{\gamma} \mapsto f(q, \phi^{-1}_{\mathbf{r}_q}(\vec{\gamma}))$ is Lebesgue integrable. A function $f : \mathbb{A} \to \mathbb{R}$ is integrable if for each $\delta \in \Delta$ the function $t \mapsto f(t, \delta)$ is Lebesgue integrable.

measures[3] which are products of counting measures on discrete sets Σ, Q and Lebesgue measure over subsets of $\mathbb{R}^m$ for some $m \geq 0$ (e.g. Γ_q, $[0, M]$). We denote by $\mathfrak{B}(\mathbb{S})$ (resp. $\mathfrak{B}(\mathbb{A})$) the set of measurable subsets of $\mathbb{S}$ (resp. $\mathbb{A}$).

The volume of the set of n-length runs is defined by:

$$\mathtt{Vol}(\mathcal{R}_n) = \int_{\mathcal{R}_n} 1 d[s, \vec{\alpha}] = \int_{\mathbb{S}} \int_{\mathbb{A}^n} \mathbf{1}_{s \triangleright \vec{\alpha} \neq \perp} d\vec{\alpha} ds$$

Remark 2. *The reduced version of runs is necessary when dealing with integrals (and densities in the following). Indeed the following integral on the extended version of runs is always null since variables are linked ($s_{i+1} = s_i \triangleright \alpha_i$ for $i = 0..n-2$):*

$$\int_{\mathbb{A}} \int_{\mathbb{S}} \dots \int_{\mathbb{A}} \int_{\mathbb{S}} 1_{s_0 \alpha_0 \cdots s_{n-1} \alpha_{n-1} \in \mathcal{R}_n} ds_0 d\alpha_0 \dots ds_{n-1} d\alpha_{n-1} = 0.$$

Entropy of a TRG

Now, there are two ways of defining entropy for TRGs. Either with the volume sequence of $\mathbb{L}_n(\mathcal{G}) = \sum_{\pi \in \Delta^*} \pi \times P_\pi$ or with that of its runs $\mathcal{R}_n$. Fortunately these two definitions coincide and moreover they involve true limit instead of the lim sup of (2.1).

Proposition 5. *The entropy of a timed region graph satisfies:*

$$\mathcal{H}(\mathcal{G}) = \lim_{n \to \infty} \frac{1}{n} \log_2 \mathtt{Vol}(\mathbb{L}_n(\mathcal{G})) = \lim_{n \to \infty} \frac{1}{n} \log_2 \mathtt{Vol}(\mathcal{R}_n(\mathcal{G})) \in [-\infty, \log_2(M\Sigma)]$$

The proof of this proposition is quite technical and will be one of the purposes of the next section.

2.3 Link between BDTA and TRGs (technical section).

The content of this section will not be used in the rest of the thesis. It just explains how to pass from BDTA to its underlying TRG without changing the entropy. The proof of Proposition 5 is also given here. This section can be skipped by a reader not interested by technical developments.

This section goes as follows: firstly we recall the region-splitting transformation of a BDTA [AD09a]. The region-split form of a BDTA $\mathcal{A}$ is a BDTA $\mathcal{A}'$ with the same entropy which admits a region decomposition of the state space. Secondly, we define the underlying TRG of a region-split BDTA and show that entropy of both objects coincide. Thirdly we recall the equations of timed languages and volume functions [AD09a] for region-split BDTA (which fit also TRGs). Finally we prove that the different notions of entropies associated to a TRG coincide.

[3] We refer the reader to [Bil12] for an introduction to measure and probability theory.

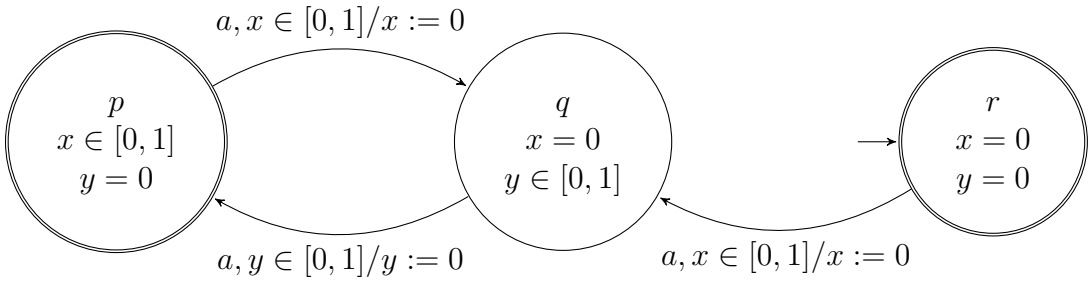

Figure 2.3: Fleshy region-split forms of automata $\mathcal{A}$ from Figure 2.1. Entry regions are given by condition in nodes.

2.3.1 The region-splitting of [AD09a]

We say that a BDTA $\mathcal{A} = (Q, \Sigma, C, \Delta, I, F)$ is in a *region-split form* if the following properties hold:

B 1. Each location and each transition of $\mathcal{A}$ is visited by some accepting run.

B 2. For every location $q \in Q$ a unique clock region $\mathbf{r}_q$ (called its *entry region*) exists, such that the set of clock values with which q is entered is exactly $\mathbf{r}_q$. For the initial location q_0, its entry region is the singleton $\{\mathbf{0}\}$.

B 3. For every $\delta \in \Delta$ the guard $\mathfrak{g}(\delta)$ is just one region. All the clock values satisfying $\mathfrak{g}(\delta)$ are time-reachable from $\mathbf{r}_q$.

Notice, that B2 and B3 imply that $\mathfrak{r}(\mathfrak{g}) = \mathbf{r}_{q'}$ for every δ.

Proposition 6. *[AD09a] Given a deterministic BDTA $\mathcal{A}$, a region-split TA $\mathcal{A}'$ accepting the same language can be constructed.*

Given a region-split BDTA $\mathcal{A}$ its underlying timed region graph $\mathcal{G}$ is obtained by deleting I, F and `Lab` in its definition (see section 2.1.2). In particular, for every $n \in \mathbb{N}$, $\mathcal{R}_n(\mathcal{G}) = \mathcal{R}_n(\mathcal{A})$.

2.3.2 BDTAs and TRGs have the same entropy

We show in this section that the passage from a region-split BDTA $\mathcal{A}$ to its underlying TRG $\mathcal{G}$ is safe wrt. the entropy:

Proposition 7. $\mathcal{H}(\mathbb{L}(\mathcal{G})) = \mathcal{H}(\mathbb{L}(\mathcal{A}))$

We need first a technical lemma to prove the Proposition. This lemma permits to upper bound the volume of any polytope associated to a path π' by the volume of the polytope of a two-side extension of this path $\pi\pi'\pi''$ which is accepting for $\mathcal{A}$ (i.e. goes from the initial location to a final location).

This lemma is based on the following fact: for any path π the set of *date* vectors $P_\pi^{date} = \{(T_1, \dots, T_n) \mid (T_1, T_2 - T_1, \dots, T_n - T_{n-1}) \in P_\pi\}$, corresponding to delay vectors in P_π, is a zone (with volume $\texttt{Vol}(P_\pi^{date}) = V_\pi$). Indeed, to show this fact, it suffices to remark that every condition $\sum_{k=i}^{j} t_k \in (A, B)$ of the contiguous polytope P_π yields a condition $T_j - T_i \in (A, B)$. Therefore P_π^{date} is a zone as defined by a conjunction of conditions of the form $T_j \sim T_i + c$ where $\sim \in \{<, >\}$ and $c \in \mathbb{N}$.

Lemma 2. *Let π, π', π'' be three consecutive paths, $n = |\pi\pi'\pi''|$ and $k = |\pi| + |\pi''|$ then*

$$\frac{(n-k)!}{n!} V_{\pi'} \leq V_{\pi\pi'\pi''}.$$

Proof. $P^{date}_{\pi\pi'\pi''}$ and $P^{date}_{\pi'}$ are zones (and thus unions of regions). We denote by $\mathbf{Reg}(A)$ the set of regions included in a zone A with maximal dimension $\dim(A)$. We have $\texttt{Vol}(A) = \sum_{\mathbf{r} \in \mathbf{Reg}(A)} \texttt{Vol}(\mathbf{r}) = \frac{1}{\dim(A)!} |\mathbf{Reg}(A)|$. It remains to prove that $|\mathbf{Reg}(P^{date}_{\pi'})| \leq |\mathbf{Reg}(P^{date}_{\pi\pi'\pi''})|$. This is true since every region of $\mathbf{Reg}(P^{date}_{\pi'})$ is the projection of several regions of $\mathbf{Reg}(P^{date}_{\pi\pi'\pi''})$. □

Proof of Proposition 7. The inequality $\mathcal{H}(\mathbb{L}(\mathcal{A})) \leq \limsup_{n\to\infty}(\log_2 V_n(\mathcal{G}))/n$ is straightforward since $V_n(\mathcal{A}) = \sum_{q_0 \xrightarrow{\pi} F} V_\pi \leq \sum_{\pi \in \Delta^n} V_\pi = V_n(\mathcal{G})(\mathcal{G})$. We show the converse inequality. Let k be a constant greater than twice the diameter of $\mathcal{A}$. We will show that for every $n \in \mathbb{N}$, $\frac{(n-k)!}{n!} V_n(\mathcal{G}) \leq \sum_{i=0}^{k} V_{n+i}(\mathcal{A})$, then taking $\limsup_{n\to\infty} \frac{1}{n} \log_2(\cdot)$ in both sides of the inequality gives the expected result. For every path π', there exist two paths π and π'' such that π starts in the initial state $(q_0, \mathbf{0})$ and $\pi\pi'\pi''$ leads to a final region. By Lemma 2 we have $\frac{(n-k)!}{n!} V_{\pi'} \leq V_{\pi\pi'\pi''}$. Summing over all the paths π' of length n we obtain the wanted inequality $\frac{(n-k)!}{n!} V_n(\mathcal{G}) \leq \sum_{\pi' \in \Delta^n} V_{\pi\pi'\pi''} \leq \sum_{i=0}^{k} V_{n+i}$ (indeed all the paths of the sum are distinct and contribute to one V_{n+i} for $i \in \{0, \dots, k\}$). □

2.3.3 Proof of Proposition 5

Existence of the limit in $[-\infty, \log_2(M\Delta)]$

The following lemma gives useful properties of V_π:

Lemma 3.
- *Sub-multiplicativity: for all $\pi, \pi' \in \Delta^*$: $V_{\pi\pi'} \leq V_\pi V'_\pi$;*
- *Upper bound: for all $\pi \in \Delta^*$: $V_\pi \leq M^{|\pi|}$;*

which translate directly on volumes $V_n(\mathcal{G})$:

Lemma 4.
- *Sub-multiplicativity: $V_{n+m}(\mathcal{G}) \leq V_n(\mathcal{G}) V_m(\mathcal{G})$.*
- *Upper bound: $V_n(\mathcal{G}) \leq M^n |\Delta|^n$.*

As a consequence the sequence $\log_2(V_n(\mathcal{G}))$ is sub-additive $\log_2(V_{n+m}(\mathcal{G})) \leq \log_2(V_n(\mathcal{G})) + \log_2(V_m(\mathcal{G}))$. By a classical lemma on sub-additive sequences [Fek23], we deduce that $\frac{1}{n} \log_2(V_n(\mathcal{G}))$ admits a limit in $[-\infty, \log_2(M\Delta)]$.

2.3.4 Recurrent equations on volume functions [AD09a]

Here we expose the beautiful equations on languages and volumes described by Asarin and Degorre in [AD09a]. We slightly generalize this work by allowing some locations to be non final and we give a more concise notation based on the integrations over timed transition defined in section 2.2.4.

Given a BDTA $\mathcal{A}$, for every state s, let $\mathbb{L}(s)$ be the set of all the timed words corresponding to the runs of the automaton starting at this state, let $\mathbb{L}_n(s)$ be its sublanguage consisting of its words of length n, and $v_n(s)$ the volume of this sublanguage.

By definition of runs of a timed automaton, we obtain the following language equations:

$$\mathbb{L}_0(s) = \varepsilon \text{ if } s \text{ is final}; \quad \mathbb{L}_0(s) = \emptyset \text{ otherwise};$$
$$\mathbb{L}_{k+1}(s) = \bigcup_{\alpha : s \triangleright \alpha \neq \perp} \mathtt{Lab}(\alpha)\mathbb{L}_k(s \triangleright \alpha)$$

Passing to volume we get:

$$v_0(s) = 1_F(s) \quad \text{where } 1_F \text{ is the indicator function of the final states } F; \tag{2.3}$$
$$v_{k+1}(s) = \int_{\mathbb{A}} v_k(s \triangleright \alpha)\, d\alpha \quad \text{with } v_k(\perp) = 0, \tag{2.4}$$

The volume functions v_n satisfy the following nice property:

Lemma 5 ([AD09a])**.** *The function $v_n(q, \vec{x})$ restricted to a location q and a clock region can be expressed by a polynomial of degree n with rational coefficients in variables $\vec{x}$.*

Thus in order to compute the volume $V_n(\mathcal{A})$ one should find by symbolic integration polynomial functions $v_k(s)$ for $k = 0..n$, and finally compute $v_n(q_0, 0)$.

Theorem 1 ([AD09a])**.** *For a BDTA $\mathcal{A}$, the volume $V_n(\mathcal{A})$ is a rational number, computable from $\mathcal{A}$ and n using the procedure described above.*

A recursive characterization of volume functions (given by (2.3) and (2.4)) holds also for TRGs (one must take $F = \mathbb{S}$ in (2.3)).

We can define another volume sequence by $V_n^{\sup}(\mathcal{G}) =_{\text{def}} \sup_{s \in \mathbb{S}} \mathtt{Vol}(\mathbb{L}_n(s))$ which defines the same entropy as the volume sequence of runs and timed words (see (2.6) below).

Coincidence between the different entropies

We first compare the different volumes associated to a path.

Lemma 6. *If π is a path of length n starting in a d-dimensional region $\mathbf{r}_p$ then*

$$\frac{(n-d)!}{n!} V_\pi \leq \int_{\Gamma_p} \mathtt{Vol}[P_\pi(\vec{\gamma})] d\vec{\gamma} \leq \frac{1}{d!} \sup_{\vec{x} \in \mathbf{r}_p} \mathtt{Vol}[P_\pi(\vec{x})] \leq \frac{1}{d!} V_\pi.$$

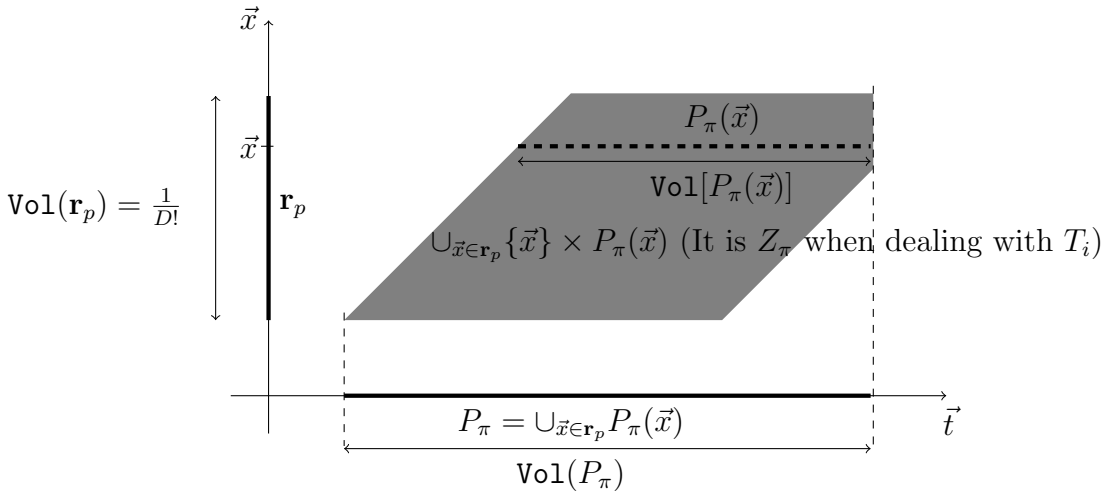

Figure 2.4: The three polytopes associated to a path.

Proof. We advise the reader to look at Figure 2.4 for the geometric intuition of the proof. The third inequality comes from language inclusion: for all $\vec{x} \in \mathbf{r}_p$, $P_\pi(\vec{x}) \subseteq P_\pi$ and thus $\mathtt{Vol}(P_\pi(\vec{x})) \leq \mathtt{Vol}(P_\pi)$. The second inequality holds since $\frac{1}{d!} = \int_{\Gamma_p} 1 d\vec{\gamma}$ is the volume of Γ_p. The first inequality is the difficult one. Mutatis mutandis, it is proved in the same way as Lemma 2 using a zone Z_π (defined later) whose volume is $\int_{\Gamma_p} \mathtt{Vol}[P_\pi(\vec{\gamma})] d\vec{\gamma}$ and its projection P_π^{date} of volume $\mathtt{Vol}(P_\pi) = V_\pi$. The zone Z_π involves variables $T_{-d}, \ldots, T_{-1}, T_1, \ldots, T_n$ links to π as follows. The T_i for positive i are the dates of the zone P_π^{date}. We denote by $x_1 > x_2 \ldots > x_d$ the d affinely independent clock values in $\mathbf{r}_p$ (each one corresponds to a γ_i). For $i \in \{1, \ldots, d\}$ we define $T_{-i} = -x_i < 0$ with the intuition that T_i records the last date in the past when the clock x_i was reset. The relation between all the T_i defines a zone Z_π. Its volume is $\int_{Z_\pi} 1 dT_{-D} \ldots dT_n$ which, after the change of coordinates $x_i \leftarrow T_{-i}$ for $i \in \{1, \ldots, d\}$, gives the expected value $\int_{\mathbf{r}} \mathtt{Vol}[P_\pi(\vec{\gamma})] d\vec{\gamma}$.

An example of a zone Z_π is the zone $Z_{\delta^2\delta^3}$ associated to the cycle $\delta^2\delta^3$ of Example 2.2. In that case $T_{-1} = -x = -\gamma$ and $(T_{-1}, T_1, T_2) \in Z_{\delta^2\delta^3}$ iff $-1 < T_{-1} < 0$, $0 < T_1$, $0 < T_1 - T_{-1} < 1$ and $0 < T_2 - T_1 < 1$. □

By summing over all paths π of length n of a TRG we can relate the volumes of its timed language with the volumes of its runs and with the suprema of its volume functions $V_n^{\sup}(\mathcal{G}) = \sup_{s\in\mathbb{S}} \mathtt{Vol}(\mathbb{L}_n(s))$.

$$\frac{(n-d)!}{n!} V_n(\mathcal{G}) \leq \mathtt{Vol}(\mathcal{R}_n(\mathcal{G})) \leq |Q| V_n^{\sup}(\mathcal{G}) \leq |Q| V_n(\mathcal{G}) \tag{2.5}$$

where $d = |C|$ is the number of clocks. And by taking $\lim_{n\to\infty} \log_2(\cdot)/n$ of each term, we conclude the proof of Proposition 5:

$$\mathcal{H}(\mathcal{G}) = \lim_{n\to\infty} \log_2 \mathtt{Vol}(\mathcal{R}_n(\mathcal{G}))/n = \lim_{n\to\infty} (\log V_n^{\sup}(\mathcal{G}))/n = \lim_{n\to\infty} (\log_2 V_n(\mathcal{G}))/n. \tag{2.6}$$

Chapter 3

The thin-thick alternative and its consequences

Abstract of the chapter

In previous literature on timed automata, it was noticed that they are in several aspects too precise, which leads sometimes to strange artifacts, mathematical pathologies or unrealistic models. In particular, some timed automata are non-implementable, non-robust, behave badly under discretization, have many Zeno runs etc. In this chapter, we propose a unifying approach to most of these issues for timed region graph (or equivalently bounded deterministic timed automata). We classify these TRGs either as **thin** or as **thick**. In thin TRGs, all the infinite trajectories are, in some weak sense, Zeno; the discretization of long trajectories is difficult, since it requires very small discretization step. In thick TRGs, most of trajectories are non-Zeno and behave well under discretization; such TRGs satisfy a sort of pumping lemma. Formally, the thin-thick alternative is based on the notion of entropy of timed regular languages introduced by E. Asarin and A. Degorre in [AD09a, AD09b] and recalled in the preliminaries (Chapter 2). Thin languages have the entropy $= -\infty$ while thick have a larger one.

Chapter structure

This chapter goes as follows: first we make simple observations on pathological behaviours and relate their volumes to those of simplices. Then, inspired by Puri, we characterize reachability in algebraic terms, introducing the *monoid of orbit graphs*. This eventually leads us to the definition of a forgetful path. After this, we exhibit a Lyapunov function, which decreases along all pathological runs. Next comes our pumping lemma (Theorem 3), where it is shown that sufficiently long paths containing a ball of radius η necessarily contain a forgetful cycle. This result borrows a useful theorem from Simon (Theorem 4), about factorization forests. Finally we conclude by stating the equivalences of Theorem 5, which justify that the rough dichotomy *thin* vs. *thick* is in fact a precise way to distinguish between "ill behaving" and "well behaving" TRG (and thus timed automata); and by showing, in Theorem 6, that this can be decided by algebraic methods (PSPACE algorithm).

Technical assumptions

The theory of this chapter applied to every language recognized by bounded deterministic timed automata. We consider w.l.o.g. for the entropy fleshy timed region graphs and their languages (see Chapter 2). In particular every path considered are fleshy. Several results holds also for non fleshy paths e.g. the first equivalence of proposition 8 and the definition of forgetful paths can include the non fleshy paths. We refer the reader to [Sta12, SBMR13] for use of forgetfulness when non-fleshiness is allowed.

3.1 Preliminaries

3.1.1 Thinness, simplices and examples

Our analysis of thin languages will start with a simple observation that the volume of k-dimensional simplices tends to 0 faster than any exponential:

Proposition-definition 2. *We call simplices of "type 1" and of "type 2", the sets of points $\vec{t} \in \mathbb{R}^k$ respectively satisfying the sets of inequalities $0 \leq t_1 + \cdots + t_k \leq 1$, $t_i \geq 0$ and $0 \leq t_1 \leq \cdots \leq t_k \leq 1$. Simplices of both types have volume $\frac{1}{k!}$.*

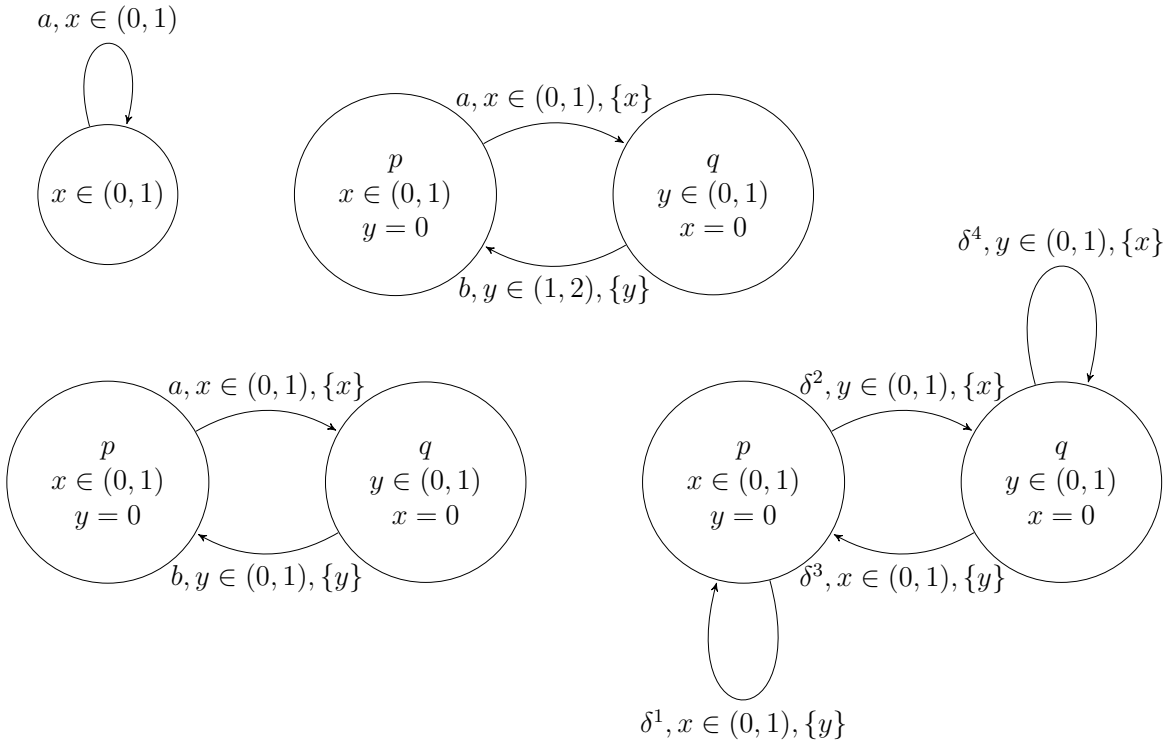

Figure 3.1: First row: thin TRGs $\mathcal{G}_1$, $\mathcal{G}_2$. Second row: thick ones $\mathcal{G}_3$, $\mathcal{G}_4$. Entry regions are given by conditions in nodes.

The TRGs depicted in Figure 3.1 illustrate the concepts of thinness and thickness. On one hand, $\mathcal{G}_1$ and $\mathcal{G}_2$ are examples of TRGs having thin languages. The case of $\mathcal{G}_1$ is straightforward: $\mathbb{L}_n(\mathcal{G}_1) = \{t_1, \ldots, t_n \mid \sum_{i \leq n} t_i < 1\}$ is a simplex of type 1, thus $\mathbb{L}(\mathcal{G}_1)$ is thin. That of $\mathcal{G}_2$ is slightly more involved, as $\mathbb{L}_n(\mathcal{G}_2) = \{t_1, \ldots, t_n \mid \forall i, t_{2i} + t_{2i+1} < 1 \wedge t_{2i+1} + t_{2i+2} < 1\}$. But we can make the following change of variables: $u_{2i+1} = 1 - t_{2i+1}$ and $u_{2i} = t_{2i}$, mapping the language polytope into the simplex $0 < u_1 < \ldots u_n < 1$. This transformation preserves volumes, thus $V_n(\mathcal{G}_2) = \frac{1}{n!}$. This is an example of TRG that is thin although it satisfies the *progress cycle condition* (i.e. resetting all clocks along each cycle).

On the other hand, examples[1] $\mathcal{G}_3$ and $\mathcal{G}_4$ are thick. Indeed their entropies can be computed symbolically using techniques of [AD09a], which give us respectively $\log_2 \frac{2}{\pi}$ and $\log_2 \log_2(e)$. Note that $\mathcal{G}_4$ does not satisfy the progress cycle condition.

3.1.2 Point to point reachability: algebraic characterization

In this section, we characterize the reachability relation of a BDTA in terms of an algebraic structure: the monoid of orbit graphs. Our analysis is less detailed than those in [CJ99, Dim02, Krc09] and follows the lines of [Pur00].

Monoid of orbit graphs

For a location $q \in Q$, we denote by $V(q) = \{S_1, \ldots, S_p\}$ the vertices of the closed region $\overline{\mathbf{r}_q}$. Any point $\vec{x}$ in the region is uniquely described by its barycentric coordinates $\lambda_1, \ldots, \lambda_p$, i.e. nonnegative numbers such that $\sum_{i=1}^p \lambda_i = 1$ and $\vec{x} = \sum_{i=1}^p \lambda_i S_i$.

Given $q, q' \in Q$, we call *orbit graph* a tuple (G, q, q') with G a graph with vertices $V(q) \uplus V(q')$ if q and q' are different and $V(q)$ otherwise, and with edges going from $V(q)$ to $V(q')$. Informally, an edge from S to S' means that the clock vector at the vertex S can reach the clock vector at S' along some transition or path.

Orbit graphs compose in the natural way: given (G_1, q_1, q_1'), and (G_2, q_2, q_2') their product $(G, q_1, q_2') = (G_1, q_1, q_1') \cdot (G_2, q_2, q_2')$ is defined if $q_1' = q_2$. There is an edge from S to S'' in G if and only if there exists S' such that (S, S') and (S', S'') are edges of G_1 and G_2. Whenever $q_1' \neq q_2$, we put $(G_1, q_1, q_1') \cdot (G_2, q_2, q_2')$ equal to some special (absorbing) element $\mathbf{0}$. The set $\mathfrak{G}$ of orbit graphs, augmented with $\mathbf{0}$ and a neutral element $\mathbf{1}$ has a structure of finite monoid.

An orbit graph (G, q, q') can be represented by its adjacency matrix M of size $|V(q)| \times |V(q')|$. Products in the monoid of orbit graphs are easy to compute using matrices: $M(G_1 G_2) = M(G_1) \otimes M(G_2)$ where the "product" $\otimes$ is defined by

$$(A \otimes B)_{ij} = \max_k \min(A_{ik}, B_{kj}).$$

There exists a natural morphism $\gamma : \Delta^* \to \mathfrak{G}$ from paths to orbit graphs defined as follows. For a transition δ between q and q', we define the orbit graph $\gamma(\delta) = (G, q, q')$

[1] $\mathcal{G}_3$ is the SCC of maximum entropy of the running example of the thesis already seen in Figure 1.1, 2.1 and 2.3. $\mathcal{G}_4$ is the TRG $\mathcal{G}^{ex1}$ of the previous chapter.

whose graph G has edges $\{(S, S') \in V(q) \times V(q') \mid \exists t, S \xrightarrow{(\delta,t)} S'\}$. For a path $\pi = \delta_1 \dots \delta_n$, we define $\gamma(\pi) = \gamma(\delta_1) \dots \gamma(\delta_n)$ (it will be called the orbit graph of the path π). For the empty path we have $\gamma(\varepsilon) = \mathbf{1}$, and for any non-consecutive path $\gamma(\pi) = \mathbf{0}$.

For example, the orbit graphs of cycles ab and ba of $\mathcal{G}_3$ and $\mathcal{G}_4$ are complete, the orbit graphs of the other examples of the chapter are given in Figure 3.2.

Adding clock resets

For future use, we must enrich the monoid of orbit graphs by adding information on clock resets. Elements of the monoid $\mathcal{M}$ are couples (orbit graph, subset of clocks) (and also, as before, two special elements $\mathbf{0}, \mathbf{1}$), the product rule is:

$$(O_1, X) \cdot (O_2, Y) = \begin{cases} (O_1 \cdot O_2, X \cap Y), & \text{if } O_1 \cdot O_2 \neq \mathbf{0} \\ \mathbf{0}, & \text{otherwise.} \end{cases}$$

For each $\pi \in \Delta^*$ we denote by $\nu(\pi)$ the set of clocks not reset along the path π. We define a morphism $\mu : \Delta^* \to \mathcal{M}$ as follows: $\mu(\pi) = (\gamma(\pi), \nu(\pi))$.

Orbit graphs and reachability

The orbit graph of a path $\gamma(\pi)$ remarkably determines its reachability relation.

Lemma 7 (Puri [Pur00])**.** [2] *Let $\vec{x}$ and $\vec{x}'$ be two clock vectors with barycentric coordinates $\vec{\lambda}$ and $\vec{\lambda}'$. Then $(\vec{x}, \vec{x}') \in \mathtt{Reach}(\overline{\pi})$ iff there exists a stochastic matrix $P \preceq M(\gamma(\pi))$, such that $\vec{\lambda}P = \vec{\lambda}'$.*

Here matrix "inequality" $A \preceq B$ means that $B_{ij} = 0 \Rightarrow A_{ij} = 0$ for all i, j.

The following particular case is of interest to us:

Proposition 8. *$\gamma(\pi)$ is complete iff $\mathtt{Reach}(\overline{\pi}) = \overline{\mathbf{r}_q} \times \overline{\mathbf{r}_{q'}}$, or equivalently iff $\mathtt{Reach}(\pi) = \mathbf{r}_q \times \mathbf{r}_{q'}$.*

In this case, we say that π is *forgetful*. The intuition is that clock values reached after reading π are independent of clock values before reading it. Remark that the orbit graph of a forgetful cycle always is an idempotent of the monoid of orbit graphs. Such elements of this monoid (and corresponding elements in $\mathcal{M}$) will be referred to as *forgetful idempotents*.

Proof of proposition 8. We use Lemma 7 to show that $\gamma(\pi)$ is complete iff $\mathtt{Reach}(\overline{\pi}) = \overline{\mathbf{r}_q} \times \overline{\mathbf{r}_{q'}}$. Suppose that $\gamma(\pi)$ is complete. For all $\vec{x}, \vec{x}'$, we denote by $\vec{\lambda}, \vec{\lambda}'$ the vectors with corresponding barycentric coordinates. We define P as the matrix with rows equal to $\vec{\lambda}'$, we have $\vec{\lambda}P = \vec{\lambda}'$ and then $(\vec{x}, \vec{x}') \in \mathtt{Reach}(\overline{\pi})$. We conclude that $\gamma(\pi)$ being complete implies that $\mathtt{Reach}(\overline{\pi}) = \overline{\mathbf{r}_q} \times \overline{\mathbf{r}_{q'}}$. The converse is trivial.

The second equivalent characterization is a consequence of Proposition 3. □

[2] An intuition behind this lemma could be as follows. A clock vector with barycentric coordinates $\vec{\lambda}$ in a region can be seen as a probabilistic distribution over vertices of this region. The lemma says that this distribution, evolves as in some Markov chain.

Other particular cycles

Two other kinds of cycles are often considered in the literature: in a *progress cycle* [Pur00] (already mentioned above), every clock is reset at some edge; in a *regenerating cycle* [SV07], at least one edge resets all the clocks.

The condition of progress cycle can be seen as a weaker kind of forgetting: the state after such a cycle is exactly determined by the delays of the cycle (see Lemma 9 below). Nevertheless the orbit graph of a progress cycle is not always strongly connected (e.g. cycle ab of $\mathcal{G}_2$ depicted in Figure 3.2); in that case, clock values in starting states and ending states are still dependent.

More precisely, we have the following strict inclusions:

Proposition 9. *progress cycles $\supsetneq$ forgetful cycles $\supsetneq$ regenerating cycles.*

Proof. First inclusion: if a cycle π is not progressing, then there is one clock x, which is not reset along that cycle. The value of x cannot decrease along the transitions of this cycle. Moreover, necessarily this clock is one of the non-zero clocks of the entry region $\mathbf{r}$ of the starting location of π. Runs realizing π must start with clock $x = x_0 > \lfloor x_0 \rfloor$ and must end with $x = x_1 \geq x_0$. Thus $\texttt{Reach}(\overline{\pi}) \neq \overline{\mathbf{r}} \times \overline{\mathbf{r}}$.

Second inclusion: a regenerating cycle π necessarily traverses, after the full reset, a location q having singleton $\{0\}$ as its entry region. Let us call p the location where π starts. We define π_1 and π_2 such that $\pi = \pi_1 \cdot \pi_2$, where π_1 goes from p to q and π_2 goes from q to p. Necessarily $\texttt{Reach}(\pi_1) = \mathbf{r}_p \times \{0\}$ and $\texttt{Reach}(\pi_2) = \{0\} \times \mathbf{r}_p$. The reachabilty relation of π is the composition of that of π_1 and π_2, i.e. $\mathbf{r}_p \times \mathbf{r}_q$, thus π is forgetful.

In order to prove that the inclusions are strict, consider in Figure 3.1, the forgetful cycle labeled ab in $\mathcal{G}_3$, which is not regenerating, and the progress cycle labeled ab in $\mathcal{G}_6$, which is not forgetful. □

A remark is in order: in most works using progress or regenerating cycles, **all** the cycles are required to satisfy the considered property. In our work, **existence** of one forgetful cycle is necessary and sufficient to characterize "non-degenerate" (i.e. thick) automata.

3.1.3 Linear Lyapunov functions and sub-exponential volume.

The aim of this section is to prove Lemma 11. It informally states that the iteration of a non-forgetful cycle yields a fast decreasing volume (and thus an entropy equal to $-\infty$). The proof of this lemma involves Lyapunov functions and affine expansive functions defined as follows.

Given a cycle π we say that $f(\vec{x}) \geq 0$ is a *Lyapunov function* for this cycle if for any $(\vec{x}, \vec{x}') \in \texttt{Reach}(\overline{\pi})$ it holds that $f(\vec{x}') \leq f(\vec{x})$. An affine function $g : \mathbb{R}^n \mapsto \mathbb{R}$ is *expansive* if it is of the form $(t_1, \ldots, t_m) \mapsto C + \sum_{j=1}^{m} \alpha_j t_j$ with $|\alpha_j| \geq 1$ for some $j \leq m$.

We need three lemmas to prove Lemma 11. Indeed the core of its proof will be in three steps:

step 1 Lyapunov functions exist for non-forgetful cycles (Lemma 8);

step 2 such a Lyapunov function can be expressed at each cycle as an expansive affine function of the delays read along the cycle (Lemma 9);

step 3 inequalities involving expansive affine functions yields a fast decreasing volume (Lemma 10).

If a cycle is non-forgetful, and moreover its orbit graph is not strongly connected, then it is possible to find a linear Lyapunov function:[3]

Lemma 8. *For a cycle π, if $\gamma(\pi)$ is not strongly connected then there exists a non-empty $I \subsetneq \{1, \dots, p\}$ such that $f_I(\vec{x}) =_{def} \sum_{i \in I} \lambda_i$ is a Lyapunov function for π, where $\vec{\lambda}$ stands for barycentric coordinates of $\vec{x}$.*

Proof. Let $(\vec{x}, \vec{x}') \in \texttt{Reach}(\overline{\pi})$ and $\vec{\lambda}$, $\vec{\lambda}'$ the corresponding barycentric coordinates. We must show that $\sum_{j \in I} \lambda'_j \leq \sum_{i \in I} \lambda_i$. Let P be a matrix such that $\vec{\lambda} P = \vec{\lambda}'$ (it exists by virtue of Lemma 7). There exists an SCC I of $\gamma(\pi)$ without incoming edges from other SCCs. After a change of indices putting those of I before those of its complement $\bar{I}$, the matrix P takes the following form:

$$P = \begin{pmatrix} P_{I \to I} & P_{I \to \bar{I}} \\ 0 & P_{\bar{I} \to \bar{I}} \end{pmatrix}.$$

If we decompose $\vec{\lambda}$ in $(\vec{\lambda}_I, \vec{\lambda}_{\bar{I}})$ and $\vec{\lambda}'$ in $(\vec{\lambda}'_I, \vec{\lambda}'_{\bar{I}})$ we have $\vec{\lambda}_I P_{I \to I} = \vec{\lambda}'_I$ and we are done since: $\sum_{j \in I} \lambda'_j = \sum_{j \in I} \sum_{i \in I} \lambda_i P_{ij} = \sum_{i \in I} \lambda_i \sum_{j \in I} P_{ij} \leq \sum_{i \in I} \lambda_i$. □

In this lemma, as before, $\{1, \dots, p\}$ are indices of the vertices of the region where π starts (and ends). In fact I corresponds to an initial strongly connected component (SCC) of the orbit graph, i.e. an SCC without incoming edges from other SCCs. According to the lemma, the state moves from the facet spanned by I towards other vertices of the region and cannot come back (see Figure 3.2).

The next lemma describes the same Lyapunov function $f_I(\vec{x})$ in terms of the timed word read along a progress cycle π.

Lemma 9. *If π is a progress path terminating in some p-dimensional region $\mathbf{r}$, then the clock vector $\vec{x}$ obtained after reading a timed word $\vec{t} \times \pi$ (from any initial clock vector) is a function of $\vec{t}$. Moreover, for all non-empty $I \subsetneq \{1, \dots, p\}$, there exists an expansive affine function g such that $f_I(\vec{x}) = g(\vec{t})$.*

Proof. Let $m = |\pi|$. We show that there exist coefficients $\alpha_1, \dots, \alpha_m \in \{-p, \dots, p\}$ not all null and an integer constant c such that $\sum_{i \in I} \lambda_i = c + \sum_{j=1}^{m} \alpha_j t_j$.

Up to a reordering of the clocks and a fusion of equal clocks we can suppose that the region $\bar{\mathbf{r}}$ is

$$\lfloor \vec{x} \rfloor + \{(\{x_1\}, \dots, \{x_p\}) \mid 0 \leq \{x_1\} \leq \cdots \{x_p\} \leq 1\},$$

[3]The reader acquainted with Lyapunov functions will remark that, in contradiction to the custom, our Lyapunov functions are linear. Nevertheless, they still serve to characterize a stability property: the tendency of a cyclic path to bring the clock vector closer to some facet of a region. In our case, we consider it a "bad" property.

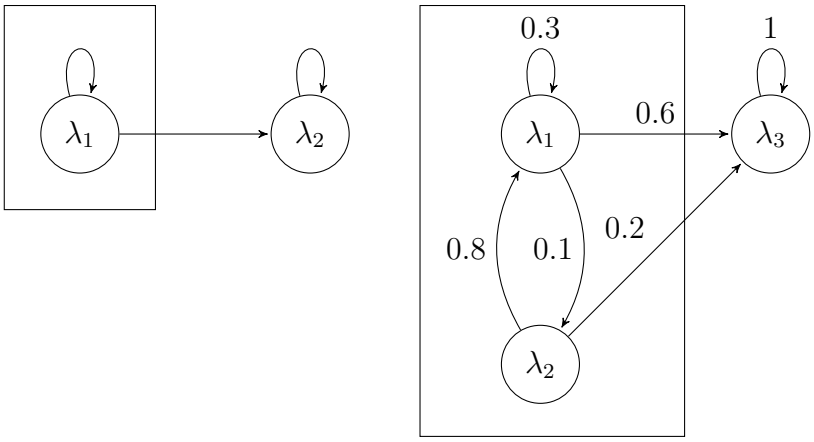

Figure 3.2: Two non strongly connected orbit graphs, the first one is the orbit graph of the cycle of $\mathcal{G}_1$, of the cycle ab of $\mathcal{G}_2$ and of the cycles δ^1 and δ^4 of $\mathcal{G}_4$. States move from the initial SCC (in the box) to the final one. By choosing the convex combination of paths given by the Markov chain on the second orbit graph we pass from state $(\lambda_1 = 0.2, \lambda_2 = 0.5, \lambda_3 = 0.3)$ to state $(\lambda'_1 = 0.46, \lambda'_2 = 0.02, \lambda'_3 = 0.52)$. The sum $\lambda_1 + \lambda_2$ can only decrease.

where $\lfloor \vec{x} \rfloor = (\lfloor x_1 \rfloor, \ldots, \lfloor x_p \rfloor)$.

Vertices of the region are $s_1 = \lfloor \vec{x} \rfloor + (0, \ldots, 0)$, $s_2 = \lfloor \vec{x} \rfloor + (0, \ldots, 1)$, $s_{p+1} = \lfloor \vec{x} \rfloor + (1, \ldots, 1)$. Therefore $\vec{x} = \sum_{i \in I} \lambda_i s_i = \lfloor \vec{x} \rfloor + (\lambda_{p+1}, \lambda_{p+1} + \lambda_p, \ldots, \lambda_{p+1} + \lambda_p + .. + \lambda_2)$ and then for $i \geq 2$ we have $\lambda_i + (\lfloor x_{p+2-i} \rfloor - \lfloor x_{p+1-i} \rfloor) = x_{p+2-i} - x_{p+1-i}$. This last quantity is, in absolute value, the sum of all delays between resets of clocks x_{p+2-i} and x_{p+1-i}. Therefore every λ_i $(i \in \{2, \ldots, p\})$ is of the form $C_i \pm \sum t_j$ with $C_i \in \mathbb{Z}$. If $1 \notin I$ then $\sum_{i \in I} \lambda_i$ is of the expected form. Otherwise, as $\lambda_1 = 1 - \sum_{i \geq 2} \lambda_i$, there exists $J \subset \{2, \ldots, p\}$ such that $\sum_{i \in I} \lambda_i = 1 + \sum_{i \in J} \pm \lambda_i$; the sum is also of the expected form. Moreover, there is one non-zero coefficient because $\sum_{i \in I} \lambda_i$ is not constant (otherwise dimension of the region would be less than p). □

Lemma 10. *Let $P \subseteq [0, M]^n$. If there exists k indices $0 = i_1 < \ldots < i_k \leq n$ and k expansive affine functions $g_1, \ldots, g_k$ such that for all $(t_1, \ldots, t_n) \in P$:*

$$1 \geq g_1(t_1, \ldots, t_{i_1}) \geq g_2(t_{i_1+1}, \ldots, t_{i_2}) \geq \ldots \geq g_k(t_{i_{k-1}+1}, \ldots, t_{i_k}) \geq 0$$

then $\texttt{Vol}(P) \leq \frac{M^{n-k}}{k!}$.

Proof. We describe an affine change of coordinate $\phi : (u_1, \ldots, u_n) \ldots (t_1, \ldots, t_n)$ with Jacobian determinant modulus $|J(\phi)| \geq 1$ and such that $\texttt{Vol}(\phi(P)) \leq \frac{M^{n-k}}{k!}$. Then the conclusion follows immediately $\texttt{Vol}(P) = |J(\phi)|^{-1} \texttt{Vol}(\phi(P)) \leq \texttt{Vol}(\phi(P)) \leq \frac{M^{n-k}}{k!}$.

For $l \in \{1, \ldots, k\}$, the function g_l has the following form

$$g_l(t_{i_{l-1}+1}, \ldots, t_{i_l}) = c_l + \sum_{j=1}^{i_l - i_{l-1}} \alpha_{j,l} t_{i_{l-1}+j} \text{ with } |\alpha_{j_l,l}| \geq 1 \text{ for some } j_l.$$

We can assume up to a permutation of coordinates (it does not change the volume) that $j_l = i_l$. The change of coordinates ϕ is defined as follows $u_{i_l} \leftarrow g_l(t_{i_{l-1}+1}, \ldots, t_{i_l})$ for $l \in \{1, \ldots, k\}$ and the other coordinates remain unchanged: $u_i \leftarrow t_i$.

Every vector $(u_1, \ldots, u_n) \in \phi(P)$ satisfies $0 \leq u_{i_1} \leq \ldots \leq u_{i_k} \leq 1$ and $u_i \in [0, M]$ for the other coordinates. Therefore $\texttt{Vol}(\phi(P)) \leq \frac{M^{n-k}}{k!}$. It remains to prove that $|J(\phi)| \geq 1$..

The Jacobian matrix is lower triangular, thus the Jacobian determinant is the product of the entries in the diagonal. These entries are 1 ($n-k$ times) and the $\alpha_{i_l,l}$ for $l \in \{1, \ldots, k\}$. We are done: $|J(\phi)| = \prod_{l=1}^{k} |\alpha_{i_l,l}| \geq 1$. □

Now we can state the key technical lemma of this section.

Lemma 11. *Let $\pi_1, \ldots, \pi_k$ be k cycles of Δ^* such that $\mu(\pi_1), \ldots, \mu(\pi_k)$ are all equal to a same non-forgetful idempotent of $\mathcal{M}$, then $V_{\pi_1 \cdots \pi_k} \leq \frac{M^{n-k}}{k!}$ where $n = |\pi_1| + \cdots + |\pi_k|$.*

Proof. If G is an idempotent orbit graph (thus equal to its transitive closure), G is complete if and only if G is strongly connected. We will distinguish two disjoint kinds of non-forgetful idempotents, those associated to non-progress cycles and those associated to progress cycles with non strongly connected orbit graphs. In the former case a clock is not reset all along the path $\pi_1 \cdots \pi_k$, thus $P_{\pi_1 \ldots \pi_k}$ is in a simplex of type 1 and the volume satisfies the inequality to prove. In the latter case, $\pi_1, \ldots, \pi_k$ are progress cycles with $\gamma(\pi_1) = \ldots = \gamma(\pi_k)$ a non strongly connected orbit graph. For $l \in \{1, \ldots, k\}$ we denote by i_l the index of the last transition of the l^{th} cycle. By virtue of Lemma 8 there exists I such that

$$1 \geq f_I(\vec{x}_0) \geq f_I(\vec{x}_{i_1}) \geq \ldots \geq f_I(\vec{x}_{i_k}) \geq 0.$$

Moreover, by Lemma 9 there exists expansive affine functions $g_1, \ldots, g_k$ (each one corresponding to a cycle) such that $f_I(\vec{x}_{i_l}) = g_1(t_{i_l+1}, \ldots, t_{i_l})$ for every $l \in \{1, \ldots, k\}$. Hence

$$1 \geq g_1(t_1, \ldots, t_{i_1}) \geq g_2(t_{i_1+1}, \ldots, t_{i_2}) \geq \ldots \geq g_k(t_{i_{k-1}+1}, \ldots, t_{i_k}) \geq 0.$$

Hypotheses of Lemma 10 are satisfied, the conclusion follows. □

3.2 Main section

3.2.1 Pumping lemma for long thick paths

For a given real $\eta > 0$, we say that a path π is *η-thick* if $V_\pi \geq \eta^{|\pi|}$. The following "pumping lemma" will play the key role in characterization of thick languages below and can be interesting by itself.

Theorem 3 (pumping lemma)**.** *For every TRG $\mathcal{G}$ and every $\eta > 0$, there exists N_η such that any η-thick path longer than N_η contains a forgetful cycle.*

The rest of this section is devoted to the proof of this result. We use Simon's theorem on factorization forests to factorize paths and find some repeated idempotent. Then, absence of forgetful-cycles yields repetition of non-forgetful idempotent along every path which by Lemma 11 imply thinness.

A *factorization forest* of a word π is an unranked labeled tree with leaves labeled by the letters of π, with root labeled by π and with two types of internal nodes:

- binary node labeled by a word $\pi_1 \cdot \pi_2$ with two children labeled by the words π_1 and π_2;
- idempotent node labeled by a word $\pi_1 \dots \pi_k$ with all $\mu(\pi_i)$ equal to a same idempotent and with children labeled by the words $\pi_1, \dots, \pi_k$.

Theorem 4 (Simon [Sim90])**.** *If μ is a morphism from Δ^* to a finite monoid $\mathcal{M}$, then every word admits a factorization forest of height at most $h(\mathcal{M}) = 9|\mathcal{M}|$.*

We suppose that there are no forgetful cycles on a long path π and consider its factorization forest of height at most $h(\mathcal{M})$. When its length n grows up, the number of leaves also grows and since the height is bounded, branching of nodes must get larger and larger. These hugely branched nodes are idempotent and satisfy hypotheses of Lemma 11, thus their volume is very small, which implies that V_π is also small. Lemma 12 below quantifies this "smallness" of V_π as function of the length of π and height of its factorization forest, and Theorem 3 follows immediately from this proposition.

Let $\texttt{LVol}$ be the function defined on paths by $\texttt{LVol}(\pi) = \log_2 V_\pi - |\pi| \log_2 M$. This function is subadditive non-positive, i.e. $\texttt{LVol}(\pi_1 \cdot \pi_2) \leq \texttt{LVol}(\pi_1) + \texttt{LVol}(\pi_2) \leq 0$. Let $L(n, h)$ be the maximum of $\texttt{LVol}(\pi)$ over paths π of length n that do not contain forgetful idempotents and admit a factorization forest of height at most h.

Lemma 12. *For any height h, for any $B > 0$, there exists $N_{h,B} \in \mathbb{N}$ such that for all $n \geq N_{h,B}$ the inequality $L(n, h) \leq -nB$ holds.*

Proof. We will define $N_{h,B}$ by induction on the height h. Let a be a factorization forest of height h with n leaves and $\pi_1, \dots, \pi_k$ be the children of the root. We distinguish two disjoint cases:

1. There are more than $m = \frac{n}{2N_{h-1,2B}}$ subtrees having less than $N_{h-1,2B}$ leaves.
2. There are less than $m = \frac{n}{2N_{h-1,2B}}$ subtrees with less than $N_{h-1,2B}$ leaves. Here the juicy part (sons with enough leaves to satisfy induction hypothesis) has more than $\frac{n}{2}$ leaves.

- In the first case: root is an idempotent node and we can apply Lem. 11:

$$\texttt{LVol}(\pi) \leq \log \frac{M^{n-k}}{k!} - n \log M = -k \log M - \log k! \leq -m \log M - \log m!,$$

which is upper bounded by $-nB$ for n large enough.

- In the second case, for $i \leq k$, we denote by n_i the length of the path π_i and by $h_i \leq h-1$ the height of its corresponding subtree. We can conclude using properties of `LVol` and the inductive hypothesis:

$$\begin{aligned}\texttt{LVol}(\pi) &\leq \sum_{i=1}^{k} \texttt{LVol}(\pi_i) \leq \sum_{i=1}^{k} L(n_i, h_i - 1) \\ &\leq \sum_{n_i \geq N_{h-1,2B}} L(n_i, h_i - 1) \leq -2B \sum_{n_i \geq N_{h-1,2b}} n_i \leq -2B\frac{n}{2} = -nB.\end{aligned}$$

□

To conclude the proof of Theorem 3, given $\eta > 0$, let $C = \log_2(\eta/M)$ and $h = h(\mathcal{M})$ the bound on height of factorization forest. Using Lemma 12, we obtain that a path longer than $N_{h,C}$ without forgetful idempotents cannot be η-thick. □

3.2.2 Characterizing thick languages

In the theorem below we characterize thick languages with forgetfulness and give two other equivalent characterizations of thickness. We say that there is a *limit cycle* along π if there exists a clock vector $\vec{x}$ and a time sequence $\vec{t}$ such that $\vec{x} \xrightarrow{\vec{t},\pi} \vec{x}$. Given $\varepsilon > 0$, in *ε-discrete limit cycles* all the components of $\vec{x}$ and $\vec{t}$ should be multiple of ε.

Theorem 5 (characterizations of thickness)**.** *For a BDTA in region split form the following conditions are equivalent and define thick languages:*

1. $\mathcal{H} > -\infty$*;*
2. *there exists a forgetful cycle;*
3. *there exists a limit cycle;*
4. *there exists an ε-discrete limit cycle with $\varepsilon > 0$.*

Equivalence between 3 and 4 can be found in [Krc09]. $2 \Rightarrow 3$ is straightforward.

Proof that $4 \Rightarrow 1$. By definition of an ε-discrete limit cycle, there exist a path π (let m be its length), $q_0, \ldots, q_{m-1}$,$\vec{x}_0, \ldots, \vec{x}_{m-1}$ and $u_1, \ldots, u_m \in \{\varepsilon, 2\varepsilon, \ldots, M - \varepsilon\}$ such that $(q_0, \vec{x}_0) \xrightarrow{(u_1,\pi_1)} (q_1, \vec{x}_1) \ \ldots \xrightarrow{(u_m,\pi_m)} (q_0, \vec{x}_0)$ and all the $\vec{x}_i$ are not on the frontier of regions and have discrete coordinates. Extending u periodically permits to have a point in P_{π^n} such that $\sum_{i=j}^{k} u_i \in [A + \varepsilon, B - \varepsilon]$ for each inequality $\sum_{i=j}^{k} t_i \in (A, B)$ defining the contiguous polytope P_{π^n}. Taking $t_i \in (u_i - \frac{\varepsilon}{m}, u_i + \frac{\varepsilon}{m})$ defines a hypercube of side $\frac{2\varepsilon}{m}$ included in P_{π^n} whose volume is therefore greater than $(\frac{2\varepsilon}{m})^{nm}$. Then $\mathcal{H}(\mathcal{G}) \geq \log_2 \frac{2\varepsilon}{m} > -\infty$. □

Proof that $1 \Rightarrow 2$. We notice first that a thick language contains long thick paths.

Lemma 13. *If $\mathcal{H} > -\infty$, there exists $\eta > 0$ such that for all n big enough, there exists an η-thick path of length n.*

Proof. We use the characterization of the entropy in terms of $\hat{V}_n = \sum_{\pi \in \Delta^n} V_\pi$ given in Proposition 7. Let $\beta = 2^{\mathcal{H}-1}$. For n large enough $\hat{V}_n \geq \beta^n$. Let π be one of the paths of Δ^n of maximal volume, then $\hat{V}_n \leq V_\pi |\Delta|^n$ and so if we pose $\eta = \frac{\beta}{|\Delta|}$ we have $V_\pi \geq \eta^n$. □

Combining Lemma 13 with Theorem 3 we find a required forgetful cycle. □

Theorem 6 (complexity of deciding thickness)**.** *The thickness property can be decided in PSPACE.*

Proof. We denote by $Q_{\mathbf{Reg}} = Q \times \mathbf{Reg}$ the set of locations of the region graph and by $\Delta_{\mathbf{Reg}}$ the set of fleshy transition of the region graph. For a location $q \in Q_{\mathbf{Reg}}$ we denote by $K_{V(q)}$ the complete graph on the vertices of $\overline{\mathbf{r}_q}$. By Theorem 5 a TA is thick if and only if there exists an orbit graph $(K_{V(q)}, q, q) \in \gamma(\Delta^*)$ such that q is reachable from $(q_0, \mathbf{0})$ by fleshy transitions and a final location q' is reachable from q by fleshy transitions. These two latter reachability questions can be solved in PSPACE using Savitch's theorem. The membership question $(K_{V(q)}, q, q) \in \gamma(\Delta^*)$ is equivalent to reachability of this element from $\gamma(\epsilon)$ in the graph of the monoid $\mathfrak{G}$ i.e the graph with vertex set $\mathfrak{G}$ and edges of the form $(O, O \cdot \gamma(\delta_{\mathbf{r},\mathbf{r}'}))$, $O \in \mathfrak{G}$, $\delta_{\mathbf{r},\mathbf{r}'} \in \Delta_{\mathbf{Reg}}$. There is at most $2^{(|C|+1)^2}$ orbit graphs for each couple of locations $(q, q') \in Q_{\mathbf{Reg}}$. The monoid $\mathfrak{G}$ has thus at most $Q_{\mathbf{Reg}} \times Q_{\mathbf{Reg}} \times 2^{(|C|+1)^2}$ elements and the reachability problem in its graph is also in PSPACE. □

In fact this problem is PSPACE complete as a consequence of [SBMR13].

3.2.3 Thin and thick SCC

The theory developed above can be refined using a decomposition of $\mathcal{G}$ into strongly connected components (SCC) $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k$. It is immediate from Proposition 4 that $\mathcal{G}$ is thin iff so are all the subautomata $\mathcal{G}_i$.

It is easy to see that long and thick paths spend most of the time in thick SCCs.

Theorem 7. *For every TRG $\mathcal{G}$ and every $\eta, \alpha > 0$, there exists $N_{\eta,\alpha}$ such that for any η-thick path of length $n > N_{\eta,\alpha}$ at most $n\alpha$ states belong to thin SCCs.*

Proof. Let $\lambda, \alpha > 0$ and π a λ-thick path of length n. Let r be the number of SCC in the TRG. We suppose by contradiction that π wanders more than αn transitions on thin SCCs and thus has a factor π' of length greater than $\frac{\alpha n}{r}$ in one thin SCC. In thin SCC, for every γ, for m large enough i.e. greater than a constant N_γ, every path π'' of length m is γ-thin: $V''_\pi < \gamma^m$. If $\frac{\alpha n}{r}$ is greater than N_γ then $V_\pi \leq V'_\pi < \gamma^{\frac{\alpha n}{r}}$, which contradicts the λ-thickness when choosing γ such that $\gamma^{\frac{\alpha}{r}} = \lambda$. □

The following technical lemma will be useful for the proof of the main theorem of Chapter 5 (Theorem 25).

Lemma 14. *In a thick SCC $\mathcal{G}_i$, there exists a constant c such that there is a forgetful cycle of length c on each location of $\mathcal{G}_i$.*

Proof. Let π be some forgetful cycle and $\mathbf{r}$ be the region where π starts and ends. Let $\mathbf{r'}$ be a region in the same SCC as $\mathbf{r}$. There is a path $\pi_{(1)}$ from $\mathbf{r}$ to $\mathbf{r'}$ and a path $\pi_{(2)}$ from $\mathbf{r'}$ to $\mathbf{r}$. We will show that $\pi_{(1)}\pi\pi_{(2)}$ is a forgetful cycle on $\mathbf{r'}$. As Puri stated in [Pur00], in an orbit graph, there is an outgoing edge from each vertex of the starting region and an incoming edge to each vertex of the ending region. Let S, S' be two vertices of $\mathbf{r'}$, let S_1, S_2 be two vertices of $\mathbf{r}$ such that (S, S_1) and (S_2, S') are respectively edges of $\gamma(\pi_{(1)})$ and $\gamma(\pi_{(2)})$. As $\gamma(\pi)$ is complete there is an edge between S_1 and S_2 and so there is an edge between S and S' in $\gamma(\pi_{(1)}\pi\pi_{(2)})$. To sum up there is a forgetful cycle $\pi_{\mathbf{r'}}$ on each region $\mathbf{r'}$ of the strongly connected sub-graph, the least common multiple of all length of $\pi_{\mathbf{r'}}$ gives an appropriate constant c. □

3.3 Conclusion and perspectives

We have identified the class of thick timed automata (those with non-vanishing language volume). Most runs in such automata are thick and exhibit a nice behaviour: they spend most of the time in thick strongly connected components (Theorem 7) and visit from time to time forgetful cycles (Theorem 3).

The key concept of forgetfulness presented here (and introduced for the first time in our paper [BA11]) is used in several chapters of this thesis (Chapter 4 and 5) as well as in our article [ABD13] and in other works such as [Sta12, SBMR13].

A direction of future work is to extend the thin-thick dichotomy to the case of non-fleshy paths and to determine when the two size measures of [AD10] are defined (see also the conclusion of Chapter 5).

In the context of verification, we believe that when analyzing a thick timed automaton, it suffices to check that the thick paths satisfy the specification, while thin ones can violate it.

Chapter 4

A maximal entropy stochastic process for a timed region graph

Abstract of the chapter

Several ways of assigning probabilities to runs of timed automata (TA) have been proposed recently. When only the TA is given, a relevant question is to design a probability distribution which represents in the best possible way the runs of the TA. We give an answer to it using a maximal entropy approach. We introduce our variant of stochastic model, the stochastic process over runs which permits to simulate random runs of any given length with a linear number of atomic operations. We adapt the notion of Shannon (continuous) entropy to such processes. Our main contribution is an explicit formula defining a process Y^* which maximizes the entropy. This formula is an adaptation of the so-called Shannon-Parry measure to the timed automata setting. The process Y^* has the nice property to be ergodic. As a consequence it has the asymptotic equipartition property and thus the random sampling wrt. Y^* is quasi uniform.

Chapter structure

In section 4.1 we recall the theory of maximal entropy Markov chain on finite graph. In the rest of the chapter we lift results of this section to the timed setting. In section 4.2 we introduce stochastic processes over runs (SPOR) of a timed region graph (defined in Chapter 2), the timed analogues of Markov chains on finite graphs. We also give definition of entropies of these continuous objects inspired by [Sha48] for the processes and by [AD09a] for the timed region graph. In the main section (section 4.3), after giving the technical assumptions, we state and prove the two main theorems: the existence of the maximal entropy SPOR which is ergodic and the asymptotic equipartition property for this process.

4.1 Maximal entropy Markov chain on a graph

In this section we recall classical results about the Markov chain of maximal entropy for a finite graph. The notations and definitions used are inspired by the books [LM95] and [Lot05].

4.1.1 Markov chain on a graph

A graph is defined by a finite set of states Q and a set of transitions Δ. Any transition $\delta \in \Delta$ has a starting state $\delta^- \in Q$ and an ending state $\delta^+ \in Q$ (there can be several transitions between the same two states).

A *path* $\delta_0 \cdots \delta_{n-1} \in \Delta^*$ ($n \geq 1$) is a word of consecutive transitions (${\delta_{i+1}}^- = {\delta_i}^+$ for $i \in \{0, \dots, n-2\}$). We denote by $\texttt{PATH}_n(G)$ the set of paths of length n.

A *Markov chain* on a graph G is given by

- initial state probabilities $p_0(q)$ for $q \in Q$ i.e. such that $\sum_{q \in Q} p_0(q) = 1$;
- conditional probabilities on transitions $p(\delta|\delta^-)$ i.e. such that for all $q \in Q$, $\sum_{\delta|\delta^- = q} p(\delta|q) = 1$ (and such that $p(\delta|q) = 0$ if $q \neq \delta^-$).

The following chain rule defines a probability distribution p_n on $\texttt{PATH}_n(G)$:

$$p_n(\delta_0 \cdots \delta_{n-1}) = p_0({\delta_0}^-) p(\delta_0|{\delta_0}^-) \dots p(\delta_{n-1}|{\delta_{n-1}}^-). \tag{4.1}$$

We also denote by p_n the induced probability measure on $\texttt{PATH}_n(G)$ i.e. for $A \subseteq \texttt{PATH}_n(G)$, $p_n(A) = \sum_{\pi \in A} p_n(\pi)$.

The initial probabilities and the conditional probabilities are respectively represented by a row vector $\vec{p_0}$ and a $Q \times Q$ stochastic matrix P such that:

$$P_{ij} = \sum_{\delta|\delta^- = i, \delta^+ = j} p(\delta|i).$$

With this notation P^k_{ij} is the probability that j is reached from i in k steps. A Markov chain is called *irreducible* if so is its transition matrix P i.e. for all $i, j \in Q$, there exists $k \in \mathbb{N}$ such that $P^k_{i,j} > 0$.

4.1.2 Ergodic stochastic processes

Stochastic processes It is convenient to use the vocabulary of stochastic processes to deal with Markov chains. We will use this vocabulary in the timed case and thus the definitions given here will be useful all along the chapter.

A stochastic process is a sequence of random variables $Y = Y_0, \dots, Y_n, \dots$ with values in a common measurable[1] set $\mathbb{D}$ (e.g. $\mathbb{D} = \Delta$).

[1] We refer the reader to [Bil12] for an introduction to measure and probability theory.

The stochastic process associated to a Markov chain on a graph is described by its joint law for each n:

$$P(Y_0 = \delta_0, \ldots, Y_{n-1} = \delta_{n-1}) = p_n(\delta_0, \ldots, \delta_{n-1}).$$

We will sometimes abuse the notation and denote a Markov chain by its corresponding stochastic process Y.

A stochastic process is said *stationary* whenever for each $n \in \mathbb{N}$ the joint law of[2] $Y_i \cdots Y_{n+i}$ does not depends on $i \in \mathbb{N}$, i.e. for all measurable set $\mathcal{D} \subseteq \mathbb{D}^{n+1}$, $P(Y_i \cdots Y_{n+i} \in \mathcal{D}) = P(Y_0 \cdots Y_n \in \mathcal{D})$ for $n \in \mathbb{N}$.

Stationarity is easy to describe in the discrete case: the stochastic process associated to a Markov chain on a graph is *stationary* if and only if

$$\vec{p}_0 P = \vec{p}_0. \tag{4.2}$$

Probability measure on bi-infinite words and ergodicity Given a measurable set $R \subseteq \mathbb{D}^{n+1}$ with $n \geq 0$, one can extend it into a set of bi-infinite sequences $R_\infty \subseteq \mathbb{D}^{\mathbb{Z}}$ as follows: $R_\infty = \{(y_i)_{i\in\mathbb{Z}} \in \mathbb{D}^{\mathbb{Z}} | y_0 \cdots y_n \in R\}$.

Let σ be the shift map on $\mathbb{D}^{\mathbb{Z}}$ i.e. $\sigma((y_i)_{i\in\mathbb{Z}}) = (y'_i)_{i\in\mathbb{Z}}$ with $y'_i = y_{i-1}$.

A probability measure μ on $\mathbb{D}^{\mathbb{Z}}$ is called shift invariant if $\mu(\sigma(A)) = \mu(A)$ for every μ-measurable set $A \subseteq \mathbb{D}^{\mathbb{Z}}$.

Let Y be a *stationary* stochastic process then by a classical extension theorem due to Kolmogorov one can define a shift invariant probability measure P_Y on $\mathbb{D}^{\mathbb{Z}}$ such that $P_Y(R_\infty) = P(Y_0 \cdots Y_{n-1} \in R)$ for every measurable $R \subseteq \mathbb{D}^n$ with $n \geq 1$.

The probability $P_Y(A)$ of a shift invariant set $A \subseteq \mathbb{D}^Z$ (i.e. $\sigma(A) = A$) can be characterized as follows

$$P_Y(A) = \lim_{n\to+\infty} P(Y_0 \cdots Y_{n-1} \in A_n) \text{ where } A_n = \{r \mid \exists y \in A, y_0 \cdots y_{n-1} = r\}. \tag{4.3}$$

A stochastic process Y is *ergodic* whenever it is stationary and every shift-invariant measurable set A has probability $P_Y(A)$ equal to 0 or 1. The following proposition gives sufficient conditions for ergodicity of the stochastic process associated to a Markov chain.

Proposition 10. *If a Markov chain is stationary and irreducible then it defines an ergodic stochastic process.*

4.1.3 Entropies

There are different notions of entropies. Their mutual connection and their meanings are discussed in the rest of this section. Here we only summarize the definitions and propositions we lift to the timed setting. We refer the reader to [Lot05, LM95, CT06] for more explanations about notions of Markov chain, entropies, almost equipartition properties...

[2]To simplify the notation, we use words instead of tuples, e.g. $Y_i \cdots Y_{n+i}$ instead of $(Y_i, \cdots, Y_{n+i})$.

Proposition-definition 8 (Entropy of a graph)**.** *Given a finite graph G, the following limit exists and is called the entropy of G:*

$$h(G) = \lim_{n\to\infty} \frac{1}{n} \log_2(|\mathtt{PATH}_n(G)|).$$

Proposition-definition 9 (Entropy of a stationary Markov chain)**.** *Let Y be a stationary Markov chain on a finite graph G then*

$$-\frac{1}{n} \sum_{\pi\in\mathtt{PATH}_n(G)} p_n(\pi) \log_2 p_n(\pi) \to_{n\to\infty} -\sum_{q\in Q} p_0(q) \sum_{\delta\in\Delta} p(\delta|q) \log_2 p(\delta|q).$$

This limit is called the entropy[3] *of the Markov chain, denoted by $h(Y)$.*

4.1.4 The asymptotic equipartition property for Markov-chain

The asymptotic equipartition property (AEP) also known as the Shannon-McMillan-Breiman theorem roughly states that almost every path of a length n generated according to an ergodic process has approximately the same probability to be chosen: $2^{-nh(Y)}$ (with $h(Y)$ the entropy of the process considered).

To state the theorem, we must recall first the notion of almost sureness. A property is said to hold almost surely (abbreviated by a.s.) when the set where it is false has probability 0. For instance, in the following theorem, (4.4) means that $P_Y(\{(y_i)_{i\in\mathbb{Z}}| - (1/n)\log_2 p_n(y_0\cdots y_{n-1}) \to_{n\to+\infty} h(Y)\}) = 1$.

Theorem 10 (AEP for Markov chain)**.** *Let Y be the ergodic stochastic process associated to an irreducible stationary Markov chain. It holds that*

$$-(1/n)\log_2 p_n(Y_0\cdots Y_{n-1}) \to_{n\to+\infty} h(Y) \quad a.s. \tag{4.4}$$

This theorem applied to a stochastic process Y^* such that $h(Y^*) = h(G)$ means that long paths have a high probability to have a quasi uniform probability:

$$p^*_n(Y^*_0\cdots Y^*_{n-1}) \approx 2^{-nh(Y^*)} = 2^{-nh(G)} \approx 1/|\mathtt{PATH}_n(G)|.$$

4.1.5 The Shannon-Parry Markov chain

In fact there exists a unique Markov chain such that $h(Y^*) = h(G)$, the Shannon-Parry Markov chain [Sha48, Par64]. Its construction is based on the classical Perron-Frobenius theorem recalled just below (see also [Lot05]).

The *spectral radius* of a matrix is the maximal modulus of its eigenvalues.

Theorem 11 (Perron-Frobenius)**.** *If M is the adjacency matrix of a strongly connected graph G (i.e. M has non-negative entries and is irreducible) then*

[3] In information theory the term entropy rate is sometimes preferred see e.g. [CT06]

- *the spectral radius ρ of M is a simple*[4] *eigenvalue of M and of its transposed matrix M^T with corresponding eigenvectors (defined up to a scalar constant) v and w which are positive;*

- *any non-negative eigenvector of M (resp. M^T) is collinear to v (resp. w).*

The following proposition links the entropy of a graph with the spectral radius of its adjacency matrix.

Proposition 11. *The entropy of G and the spectral radius ρ of its adjacency matrix are linked by the following equality $h(G) = \log_2(\rho)$.*

Theorem 12 (Shannon-Parry)**.** *If G is strongly connected then*

- *every stationary Markov chain on G satisfies $h(Y) \leq h(G)$;*

- *there exists a unique stationary Markov chain Y^* such that $h(Y^*) = h(G)$;*

- *Y^* is ergodic.*

Given a strongly connected graph G, let ρ, v, w be given by the Perron-Frobenius theorem above. Eigenvectors v and w are chosen such that $\langle v, w\rangle = \sum_{q\in Q} v_q w_q = 1$ (eigenvectors are defined up to a scalar constant). The Shannon-Parry Markov chain Y^* on G is given by: for every $q \in Q$, $\delta \in \Delta$,

$$p_0^*(q) = v_q w_q; \quad p^*(\delta|\delta^-) = \frac{v_{\delta^+}}{\rho v_{\delta^-}}. \tag{4.5}$$

The transition probability matrix of Y^* is defined by: for every $i, j \in Q$,

$$P_{ij} = \frac{M_{ij} v_j}{\rho v_i}. \tag{4.6}$$

4.2 Stochastic processes on timed region graphs

The definition of timed region graphs (TRG) and their properties are given in the preliminary chapter of the thesis (Chapter 2). We use as a running example the TRG $\mathcal{G}^{\texttt{ex1}}$ from this preliminary chapter depicted in Figure 2.2 (also called $\mathcal{G}^4$ in the previous chapter and depicted in Figure 3.1)

[4]The generalized eigenspace of an eigenvalue λ of a matrix A is the set of f such that $(A - \lambda Id)^k f = 0$ for some k. When it has dimension 1 then λ is called *simple*. This definition holds also when A is a more general positive operator as used in the following (section 4.3).

4.2.1 SPOR of a timed region graph

A *stochastic process over runs* (SPOR) of a timed region graph $\mathcal{G}$ is a stochastic process $(Y_n)_{n\in\mathbb{N}}$ such that

C.1) each Y_n takes its values in $\mathbb{D} =_{\text{def}} \mathbb{S} \times \mathbb{A}$, it is of the form $Y_n = (S_n, A_n)$;

C.2) The initial state S_0 has a probability density function (PDF) $p_0 : \mathbb{S} \to \mathbb{R}^+$ i.e. for every $\mathcal{S} \in \mathfrak{B}(\mathbb{S})$, $P(S_0 \in \mathcal{S}) = \int_{s\in\mathcal{S}} p_0(s)ds$ (in particular $P(S_0 \in \mathbb{S}) = \int_{s\in\mathbb{S}} p_0(s)ds = 1$).

C.3) Probability on every timed transition only depends on the current state: for every $n \in \mathbb{N}$, $\mathcal{A} \in \mathfrak{B}(\mathbb{A})$, for almost every[5] $s \in \mathbb{S}$, $y_0 \cdots y_n \in (\mathbb{S} \times \mathbb{A})^n$,

$$P(A_n \in \mathcal{A}|S_n = s, Y_n = y_n, \ldots, Y_0 = y_0) = P(A_n \in \mathcal{A}|S_n = s),$$

moreover this probability is given by a conditional PDF $p(.|s) : \mathbb{A} \to \mathbb{R}^+$ such that $P(A_n \in \mathcal{A}|S_n = s) = \int_{\alpha\in\mathcal{A}} p(\alpha|s)d\alpha$ and $p(\alpha|s) = 0$ if $s \triangleright \alpha = \perp$ (in particular $P(A_n \in \mathbb{A}|S_n = s) = \int_{\alpha\in\mathbb{A}} p(\alpha|s)d\alpha = 1$).

C.4) States are updated deterministically knowing the previous state and transition: $S_{n+1} = S_n \triangleright A_n$.

Given a timed region graph a SPOR of it is uniquely and entirely described by the initial and transitional PDFs $p_0(s)$ and $p(\alpha|s)$.

The Markovian properties C.3) and C.4) permit to define probability density functions for portion of runs $Y_i \cdots Y_{i+n-1}$ knowing the value of S_i (see (4.8) below) : for $\vec{\alpha} \in \mathbb{A}^n$ and $s_0 \in \mathbb{S}$ we define $p_n(\vec{\alpha}|s_0)$ by the following chain rule

$$p_n(\vec{\alpha}|s_0) = p(\alpha_0|s_0)p(\alpha_1|s_1)\ldots p(\alpha_{n-1}|s_{n-1}). \tag{4.7}$$

where for each $j = 1..n-1$ the state updates are defined by $s_j = s_{j-1} \triangleright \alpha_{j-1}$. Then $p_n(.|s)$ satisfies

$$P((S_i, A_i)\cdots(S_{i+n-1}, A_{i+n-1}) \in R|S_i = s) = \int_{\mathbb{A}^n} p_n(\vec{\alpha}|s)1_{[s,\vec{\alpha}]\in R}d\vec{\alpha}. \tag{4.8}$$

The PDF for $Y_0 \cdots Y_{n-1}$ is $p_n[s, \vec{\alpha}] =_{\text{def}} p_0(s)p_n(\vec{\alpha}|s)$ i.e.

$$P(Y_0 \cdots Y_{n-1} \in R) = \int_{\mathcal{R}_n} p_n[s, \vec{\alpha}]1_{[s,\vec{\alpha}]\in R}d[s, \vec{\alpha}].$$

The following proposition permits to characterize stationarity of a SPOR (defined in section 4.1.2) in terms of initial and conditional PDFs as in the discrete case (4.2):

[5] A property *prop* (like "f is positive", "well defined"...) on a set B holds *almost everywhere* when the set where it is false has measure (volume) 0: $\int_B \mathbf{1}_{b\not\models prop}db = 0$.

Proposition 12 (Characterization of stationarity)**.** *A SPOR is stationary if and only if for all measurable set $\mathcal{S} \in \mathfrak{B}(\mathbb{S})$ the following holds:*

$$\int_{\mathbb{S}} \int_{\mathbb{A}} p_0(s)p(\alpha|s)1_{s \triangleright \alpha \in \mathcal{S}} d\alpha ds = \int_{\mathbb{S}} p_0(s')1_{s' \in \mathcal{S}} ds'$$

Proof. The left-hand side of the equality is $P(S_0 \triangleright A_0 \in \mathcal{S}) = P(S_1 \in \mathcal{S})$ while the right hand-side is $P(S_0 \in \mathcal{S})$. Thus we must prove that a SPOR is stationary if and only if S_1 has the PDF p_0 (and thus the same law as S_0).

The "only if" part is straightforward. For the other part let Y be a SPOR such that S_1 has the PDF p_0. We first show by recurrence that for all $i \geq 0$, S_i has the PDF p_0. For this, we assume that S_n has the PDF p_0 for some $n \geq 1$ and prove that S_{n+1} has the same law as S_1 and thus has the PDF p_0. For every measurable set of states $\mathcal{S} \in \mathfrak{B}(\mathbb{S})$,

$$\begin{aligned} P(S_{n+1} \in \mathcal{S}) &= \int_{\mathbb{S}} \int_{\mathbb{A}} p_0(s)p(\alpha|s)P(S_n \triangleright A_n \in \mathcal{S}|S_n = s, A_n = \alpha)d\alpha ds \\ &= \int_{\mathbb{S}} \int_{\mathbb{A}} p_0(s)p(\alpha|s)1_{s \triangleright \alpha \in \mathcal{S}} d\alpha ds \\ &= P(S_1 \in \mathcal{S}) \end{aligned}$$

Thus for all $i \geq 0$, S_i has the PDF p_0. Now we remind from (4.8) that the PDF of $Y_i \cdots Y_{i+n-1}$ knowing that $S_i = s$ is $p_n(\vec{\alpha}|s)$. We conclude that $Y_i \cdots Y_{i+n-1}$ has the PDF $p_n(s, \vec{\alpha}) = p_0(s)p_n(\vec{\alpha}|s)$ and thus the same law as $Y_0 \cdots Y_{n-1}$.

□

Given a measurable function $f : \mathcal{R}_n \to \mathbb{R}$, we denote by $E_{P_Y}(f)$ its expectation wrt. P_Y: $E_{P_Y}(f) =_{\text{def}} \int_{\mathcal{R}_n} f[s, \vec{\alpha}]p_n[s, \vec{\alpha}]d[s, \vec{\alpha}]$.

Simulation according to a SPOR Given a SPOR Y, a run $(s_0, \vec{\alpha}) \in \mathcal{R}_n$ can be generated randomly wrt. Y with a linear number of the following operations: random pick according to p_0 or $p(.|s)$ and computing a successor. Indeed it suffices to pick s_0 according to p_0 and for $i = 0..n-1$ to pick α_i according to $p(.|s_i)$ and to make the update $s_{i+1} = s_i \triangleright \alpha_i$.

4.2.2 Entropy

In this sub-section, we define entropy for timed region graphs and SPORs. The former adapted from [AD09a] is the timed analogue of entropy of a graph (Proposition-definition 8) while the latter adapted from the Shannon's continuous entropy [Sha48] is the timed analogue of entropy of a finite state Markov chain (Proposition-definition 9).

Entropy of a timed region graph

Recall from Proposition 5 that the entropy of a timed region graph $\mathcal{G}$ is characterized by

$$\mathcal{H}(\mathcal{G}) = \lim_{n \to \infty} \frac{1}{n} \log_2(\texttt{Vol}(\mathcal{R}_n)).$$

- When $\mathcal{H}(\mathcal{G}) > -\infty$, the timed region graph is *thick*, the volume $\texttt{Vol}(\mathcal{R}_n)$ behaves wrt. n like an exponent: $2^{n\mathcal{H}(\mathcal{G})}$ (multiplied by a sub-exponential term i.e. $\lambda^{-n} << 2^{n\mathcal{H}(\mathcal{G})}/\texttt{Vol}(\mathcal{R}_n) << \lambda^n$ for every $\lambda > 1$).
- When $\mathcal{H}(\mathcal{G}) = -\infty$, the timed region graph is *thin*, the volume decays faster than any exponent: $\forall \rho > 0,\ \texttt{Vol}(\mathcal{R}_n) << \rho^n$.

Entropy of a SPOR

Proposition-definition 13. *If Y is a stationary SPOR, then*

$$E_{P_Y}(-\log p_n[s_0, \alpha_0 \cdots \alpha_n])/n \to_{n\to\infty} E_{P_Y}(-\log p(\alpha_0|s_0))$$

which can be re-written as

$$-\frac{1}{n}\int_{\mathcal{R}_n} p_n[s,\vec{\alpha}]\log_2 p_n[s,\vec{\alpha}]d[s,\vec{\alpha}] \to_{n\to\infty} -\int_{\mathbb{S}} p_0(s)\int_{\mathbb{A}} p(\alpha|s)\log_2 p(\alpha|s)d\alpha ds.$$

This limit is called the entropy of Y, denoted by $H(Y)$.

Proof.

$$\begin{aligned} E_{P_Y}(-\log p_n[s_0, \alpha_0 \cdots \alpha_n])/n &= E_{P_Y}(-\log p_0(s_0)\prod_{i=0}^{n} p(\alpha_i|s_i))/n \\ &= E_{P_Y}(-\log p_0(s_0))/n - \sum_{i=0}^{n} E_{P_Y}(\log p(\alpha_i|s_i))/n \\ &= E_{P_Y}(-\log p_0(s_0))/n - E_{P_Y}(\log p(\alpha_0|s_0)) \end{aligned}$$

This quantity tends to $E_{P_Y}(-\log p(\alpha_0|s_0))$ when $n \to +\infty$.

□

Proposition 13. *Let $\mathcal{G}$ be a timed region graph and Y be a stationary SPOR on $\mathcal{G}$. Then the entropy of Y is upper bounded by that of $\mathcal{G}$: $H(Y) \leq \mathcal{H}(\mathcal{G})$.*

Proof. The proof follows from the following fact: for all $n \in \mathbb{N}$, $h(p_n) \leq \log_2(\texttt{Vol}(\mathcal{R}_n))$ where $h(p_n) =_{\text{def}} -\int_{\mathcal{R}_n} p_n[s,\vec{\alpha}]\log_2 p_n[s,\vec{\alpha}]d[s,\vec{\alpha}]$ is the Shannon's continuous entropy of the PDF p_n. We need some definitions and properties concerning Kullback-Leibler divergence before proving this fact.

The Kullback-Leibler divergence[6] (KL-divergence) from a PDF p_n to another p'_n is

$$\int_{\mathcal{R}_n} p_n[s,\vec{\alpha}]\log_2 \frac{p_n[s,\vec{\alpha}]}{p'_n[s,\vec{\alpha}]}d[s,\vec{\alpha}].$$

[6]this notion has several other names such as relative entropy, Kullback-Leibler distance, KLIC, ... Its general definition (including the present setting) is ensured by the GYP-Theorem (see e.g. Theorem 2.4.2 of [Pin64]).

The KL-divergence is always non-negative with equality to 0 if and only if p_n and p'_n are equal almost everywhere (see e.g. [CT06] chapter 8). It permits to measure how far a probability distribution is from another one.

Now we can prove that $h(p_n) \leq \log_2(\texttt{Vol}(\mathcal{R}_n))$. The KL-divergence from an arbitrary distribution p_n to the uniform distribution $[s, \vec{\alpha}] \mapsto 1/\texttt{Vol}(\mathcal{R}_n)$ is $\log_2(\texttt{Vol}(\mathcal{R}_n)) - h(p_n) \geq 0$ with equality if and only if p_n is uniform almost everywhere. □

The main contribution of this chapter is a construction of an ergodic SPOR Y^* for which the equality $H(Y^*) = \mathcal{H}(\mathcal{G})$ holds i.e. a timed analogue of the Shannon-Parry Markov chain recalled in section 4.1.

4.3 The maximal entropy SPOR

In this main section, $\mathcal{G}$ is a timed region graph satisfying the technical condition below (section 4.3.1). We present an ergodic SPOR Y^* for which the upper bound on entropy is reached $H(Y^*) = \mathcal{H}(\mathcal{G})$ (Theorem 14). We prove also an asymptotic equipartition property for ergodic SPOR (Theorem 15) whose main corollary is that runs r generated according to Y^* has a high probability to have a quasi-uniform density of probability $p^*_n(r) \approx 1/\texttt{Vol}(\mathcal{R}_n)$.

4.3.1 Technical assumptions

In this section we explain and justify several technical assumptions on the timed region graph $\mathcal{G}$ we make in the following. Some of them where already assumed in other chapters and are just recalled, there is a novel one: the weak progress cycle condition.

Bounded delays. If the delays were not bounded the sets of runs $\mathcal{R}_n$ (for $n \geq 1$) would have infinite volumes and thus a quasi uniform random generation cannot be achieved.

Fleshy transitions. We consider timed region graphs whose transitions are *fleshy* [AD09a]: there is no constraints of the form $x = c$ in their guards. Non fleshy transitions yield a null volume and are thus useless. Deleting them reduces the size of the timed region graph considered and ensures that every path has a positive volume (see Chapter 3 and [AD09a] for more justifications and details).

Strong connectivity of the set of locations. We will consider only timed region graph which are strongly connected i.e. locations are pairwise reachable. This condition (usual in the discrete case we generalize) is not restrictive since the set of locations can be decomposed in strongly connected components and then a maximal entropy SPOR can be designed for each component. Moreover the entropy of a TRG is equal to the maximal entropy of its SCCs (Proposition 4).

Thickness. In the maximal entropy approach we adopt, we need that the entropy is finite $\mathcal{H}(\mathcal{G}) > -\infty$. This is why we restrict our attention to *thick* timed region graphs. The dichotomy between thin and thick timed region graphs was characterized precisely in Chapter 3 where it turns out that thin timed region graphs are in a sense degenerate. The key characterization of thickness is the existence of a forgetful cycle. When the locations are

strongly connected, existence of such a forgetful cycle ensures that the state space $\mathbb{S}$ is strongly connected i.e. for all $s, s' \in \mathbb{S}$ there exists $\vec{\alpha} \in \mathbb{A}^*$ such that $s \triangleright \vec{\alpha} = s'$.

Weak progress cycle condition. In [AD09a] the following assumption was made: for some positive integer constant D, on each path of D consecutive transitions, all the clocks are reset at least once. An equivalent condition (already mentioned in Chapter 3) known as the *progress cycle condition* was assumed (and justified) by Puri [Pur00] and followers in many works on robustness of timed languages.

Here we use a weaker condition: for a positive integer constant D, a timed region graph satisfies the *D weak progress condition* (D-WPC) if on each path of D consecutive transitions at most one clock is not reset during the entire path.

The timed region graph on Figure 2.2 does not satisfy the progress cycle condition (e.g. x is not reset along δ^1) but satisfies the 1-WPC.

4.3.2 Main theorems

Here we give the two main theorems of the chapter. The proof of Theorem 14 is given in section 4.3.5 as it requires some material exposed in section 4.3.3.

Theorem 14 (maximal entropy)**.** *There exists a positive real ρ and two functions $v, w : \mathbb{S} \mapsto \mathbb{R}$ positive almost everywhere such that the following equations define the PDF of an ergodic SPOR Y^* with maximal entropy ($H(Y^*) = \mathcal{H}(\mathcal{G})$):*

$$p_0^*(s) = w(s)v(s); \quad p^*(\alpha|s) = \frac{v(s \triangleright \alpha)}{\rho v(s)}. \tag{4.9}$$

Objects ρ, v, w will be defined in the next section (section 4.3.3).

The maximal entropy SPOR Y^* has also as nice features simple PDFs for n-length runs (obtained by plugging (4.9) into the chain rule (4.7)):

$$p_n^*(\vec{\alpha}|s) = \frac{v(s \triangleright \vec{\alpha})}{\rho^n v(s)}; \quad p_n^*(s, \vec{\alpha}) = \frac{w(s)v(s \triangleright \vec{\alpha})}{\rho^n}. \tag{4.10}$$

An ergodic SPOR satisfies an asymptotic equipartition property (AEP) (see [CT06] for classical AEP and [AC88] which deals with the case of non necessarily Markovian stochastic processes with density). Here we give our own AEP. It strongly relies on the pointwise ergodic theorem (see [Bil12]) and on the Markovian property satisfied by every SPOR (conditions C.3 and C.4).

Theorem 15 (AEP for SPOR)**.** *If $Y = (S_i, A_i)_{i \in \mathbb{N}}$ is an ergodic SPOR then*

$$-(1/n) \log_2 p_n[S_0, A_0 \cdots A_n] \to_{n \to +\infty} H(Y) \quad a.s.$$

To prove the theorem we use as a lemma (a weak version of) the pointwise ergodic theorem applied to the shift invariant probability measure P_Y. We refer the reader to [Bil12] for a general version of this theorem.

Lemma 15 (Pointwise ergodic theorem for P_Y)**.** *If Y is an ergodic SPOR, for every measurable function $f : \mathbb{D} \to \mathbb{R}$ such that $E_{P_Y}(|f|) < +\infty$, almost surely a bi-infinite run $y \in \mathbb{D}^{\mathbb{Z}}$ satisfies*

$$\frac{1}{n}\sum_{k=0}^{n-1} f(y_k) \to_{n\to+\infty} E_{P_Y}(f). \tag{4.11}$$

Proof of Theorem 15. We use Lemma 15 with the function $f : \mathbb{D} \to \mathbb{R}$ defined by $f : (s, \alpha) \mapsto -\log_2 p(\alpha|s)$. The left-hand side of (4.11) is equal to

$$-\frac{1}{n}\sum_{k=0}^{n-1} \log_2 p(\alpha_k|s_k) = -\frac{1}{n}\log_2 p_n[s_0, \alpha_0 \cdots \alpha_{n-1}] - \frac{1}{n}\log_2 p_0(s_0).$$

The right-hand side of (4.11) is

$$E_{P_Y}(f) = -\int_{\mathbb{S}} p_0(s) \int_{\mathbb{A}} p(\alpha|s) \log p(\alpha|s) d\alpha ds = H(Y).$$

It remains to show that $E_{P_Y}(|f|) < +\infty$. For this, we write $|f|$ as $|f| = f + 2f^-$ where $f^- = (|f| - f)/2$ is the negative part of f. By linearity of the expectation $E_{P_Y}(|f|) = E_{P_Y}(f) + 2E_{P_Y}(f^-)$ is finite since $E_{P_Y}(f) = H(Y) < +\infty$ and

$$E_{P_Y}(f^-) = \int_{\mathbb{S}} p_0(s) \int_{\alpha\in\mathbb{A}|-\log p(\alpha|s)>0} -p(\alpha|s) \log p(\alpha|s) d\alpha ds < +\infty$$

(since the function $x \mapsto -x \log_2 x$ is upper-bounded). □

Theorem 15 applied to the maximal entropy SPOR Y^* means that long runs have a high probability to have a quasi uniform density:

$$p^*_n[S^*_0, A^*_0 \cdots A^*_n] \approx 2^{-nH(Y^*)} = 2^{-n\mathcal{H}(\mathcal{G})} \approx 1/\mathtt{Vol}(\mathcal{R}_n).$$

4.3.3 Definition and properties of ρ, v and w

The maximal entropy SPOR is a lifting to the timed setting of the Shannon-Parry Markov chain of a finite strongly connected graph recalled in section 4.1. The definition of this chain is based on the Perron-Frobenius theory applied to the adjacency matrix M of the graph. The timed analogue of M is the operator Ψ introduced in [AD09a]. The objects ρ,v and w used in the definition of the maximal entropy SPOR (4.9) are spectral attributes of Ψ. To define ρ,v and w, we will use the theory of positive linear operators (see e.g. [KLS89]) instead of the Perron-Frobenius theory used in the discrete case.

The operator Ψ

Before defining the operator Ψ we describe the functional space where it is act. The set $L_2(\mathbb{S})$ is the Hilbert space of square integrable functions[7] from $\mathbb{S}$ to $\mathbb{R}$ with the scalar product

[7]Strictly speaking, elements of $L_2(\mathbb{S})$ (and $L_1(\mathbb{S})$) are classes of equivalent functions pairwise equal almost everywhere.

$\langle f, g\rangle = \int_{\mathbb{S}} f(s)g(s)ds$ and associated norm $||f||_2 = \sqrt{\langle f, f\rangle}$. We will also use the functional space $L_1(\mathbb{S})$ of integrable function with norm $||f||_1 = \int_{\mathbb{S}} |f(s)|ds$. As the state space $\mathbb{S}$ has a finite measure, $L_2(\mathbb{S})$ is included in the functional space $L_1(\mathbb{S})$.

For a function $f : \mathbb{S} \to \mathbb{R}$ we denote by $f > 0$ (resp. $f \geq 0$) the fact that $f(s) > 0$ (resp. $f(s) \geq 0$) for almost every $s \in \mathbb{S}$.

Proposition-definition 16. *The operator Ψ of a timed region graph defined by*

$$\forall f \in L_2(\mathbb{S}),\ \forall s \in \mathbb{S},\ \Psi(f)(s) = \int_{\mathbb{A}} f(s \triangleright \alpha)d\alpha \text{ (with } f(\perp) = 0\text{)}, \tag{4.12}$$

is a positive continuous linear operator on $L_2(\mathbb{S})$.

Proof. It is clear from the definition of Ψ that the operator is positive i.e. if $f \geq 0$ then so is $\Psi(f)$.

To show that Ψ is a continuous operator on $L_2(\mathbb{S})$ it suffices to prove that for all $f \in L_2(\mathbb{S})$ the operator norm $||\Psi(f)||_2 = (\int_{\mathbb{S}} \Psi(f)(s)^2 ds)^{\frac{1}{2}}$ is upper bounded by $[|\Delta|\mathtt{Vol}(\mathbb{A})]^{\frac{1}{2}}||f||_2$. In other words we will prove that

$$\int_{\mathbb{S}} \Big(\int_{\mathbb{A}} f(s \triangleright \alpha)d\alpha\Big)^2 ds \leq |\Delta|\mathtt{Vol}(\mathbb{A}) \int_{s'} f(s')^2 ds'. \tag{4.13}$$

We first prove the following inequality for every $f \in L_2(\mathbb{S})$:

$$\int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds \leq |\Delta| \int_{s'} f(s')^2 ds' = |\Delta|.||f||_2^2. \tag{4.14}$$

For this purpose we decompose the left-hand side into a sum over $\delta \in \Delta$:

$$\int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds = \sum_{\delta \in \Delta} \int_{(\vec{\gamma},t) \in G(\delta)} f((\delta^-, \vec{\gamma}) \triangleright (t, \delta))^2 d\vec{\gamma} dt \tag{4.15}$$

where $G(\delta) =_{\text{def}} \{(\vec{\gamma}, t) \in \Gamma_{\delta^-} \times [0, M] \mid (\delta^-, \vec{\gamma}) \triangleright (t, \delta) \neq \perp\}$. Now, for every $\delta \in \Delta$, we will do a change of coordinate. Let d and d' be the dimension of $\mathbf{r}_{\delta^-}$ and $\mathbf{r}_{\delta^+}$ respectively. For a real y we denote by $\{y\}$ its fractional part. Let $t, \vec{\gamma}, \vec{\gamma}'$ such that $(\delta^-, \vec{\gamma}) \triangleright (t, \delta) = (\delta^+, \vec{\gamma}')$. Modulo a permutation of coordinates that only depends on δ we have $(\{\gamma_1 + t\}, \ldots \{\gamma_d + t\}, \{t\}) = (\vec{\gamma}', \vec{\sigma})$ for some $\vec{\sigma} \in [0, 1]^{d+1-d'}$. Indeed there are two possible cases for the coordinate $\{t\}$:

- either there exist clocks null in δ^- and not null in δ^+ and thus $\{t\}$ corresponds to these clocks and is thus a coordinate of $\vec{\gamma}'$,
- either $\{t\}$ is a coordinate of $\vec{\sigma}$;

and for the other coordinates:

- a coordinate γ_i of $\vec{\gamma}$ corresponding to a non resetting clock yields a new coordinate $\{\gamma_i + t\}$ of $\vec{\gamma}'$;

- a coordinate γ_i of $\vec{\gamma}$ corresponding to a resetting clock yields a coordinate $\{\gamma_i + t\}$ of $\vec{\sigma}$.

The change of coordinates $(\vec{\gamma}, t) \mapsto (\vec{\gamma}', \vec{\sigma})$ from the set $G(\delta)$ to its image denoted by $G'(\delta)$ is linear with a Jacobian equal to 1. Making this change of coordinates in (4.15) yields:

$$\int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds = \sum_{\delta \in \Delta} \int_{(\vec{\gamma}', \vec{\sigma}) \in G'(\delta)} f(\delta^+, \vec{\gamma}')^2 d\vec{\gamma}' d\vec{\sigma}$$

If we denote by $g_\delta(\vec{\gamma}') = \texttt{Vol}(\{\vec{\sigma} \mid (\vec{\gamma}', \vec{\sigma}) \in G'(\delta)\})$ we can simplify the last integral:

$$\int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds = \sum_{\delta \in \Delta} \int_{\vec{\gamma}' \in \Gamma_{\delta^+}} f(\delta^+, \vec{\gamma}')^2 g_\delta(\vec{\gamma}') d\vec{\gamma}'$$

The coordinates of $\vec{\sigma}$ belong to $[0, 1]$ and thus for every $\vec{\gamma}' \in \Gamma_{\delta^+}$, the set $\{\vec{\sigma} \mid (\vec{\gamma}', \vec{\sigma}) \in G'(\delta)\}$ is included in a hypercube of side 1. We deduce that $g_\delta(\vec{\gamma}') \leq 1$ for every $\vec{\gamma}' \in \Gamma_{\delta^+}$ and obtain the expected inequality (4.14):

$$\int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds \leq |\Delta|. \sum_{q' \in Q} \int_{\vec{\gamma}' \in \Gamma_{q'}} f(q', \vec{\gamma}')^2 d\vec{\gamma}' = |\Delta|. ||f||_2^2$$

Now we can prove (4.13). Fubini's theorem applied to (4.14) ensures that $\alpha \mapsto f(s \triangleright \alpha)^2$ is defined and integrable for almost every s. Thus we can apply Cauchy–Schwartz inequality (in $L_2(\mathbb{A})$) to the constant function 1 and the function $\alpha \mapsto f(s \triangleright \alpha)$:

$$[\Psi(f)(s)]^2 = \Big(\int_{\mathbb{A}} f(s \triangleright \alpha) d\alpha \Big)^2 \leq \texttt{Vol}(\mathbb{A}) \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha. \tag{4.16}$$

Combining (4.14) and (4.16) we get (4.13) and conclude the proof:

$$\begin{aligned} ||\Psi(f)||_2^2 &= \int_{\mathbb{S}} \Psi(f)(s)^2 ds = \int_{\mathbb{S}} \Big(\int_{\mathbb{A}} f(s \triangleright \alpha) d\alpha \Big)^2 ds \\ &\leq \texttt{Vol}(\mathbb{A}) \int_{\mathbb{S}} \int_{\mathbb{A}} f(s \triangleright \alpha)^2 d\alpha ds \quad \text{(by (4.16))} \\ &\leq |\Delta| \texttt{Vol}(\mathbb{A}) ||f||_2^2 \quad \text{(by (4.14))}. \end{aligned}$$

□

Intuitively $\Psi(f)(s)$ is the integral of f over all the one-step successor $s \triangleright \alpha$ of s. In the same way, for a positive integer k, $\Psi^k(f)(s)$ is the integral of f over all the k-step successor $s \triangleright \vec{\alpha}$ of s:

$$\forall f \in L_2(\mathbb{S}),\ \forall s \in \mathbb{S},\ \Psi^k(f)(s) = \int_{\mathbb{A}^k} f(s \triangleright \vec{\alpha}) d\vec{\alpha} \text{ (with } f(\perp) = 0) \tag{4.17}$$

The adjoint operator Ψ^* (acting also on $L_2(\mathbb{S})$) is the analogue of $M^\top$. It is formally defined by the equation:

$$\forall f, g \in L_2(\mathbb{S}),\ \langle \Psi(f), g \rangle = \langle f, \Psi^*(g) \rangle . \tag{4.18}$$

Characterizing the adjoint of an operator is easier when it is a so-called Hilbert-Schmidt integral operator as we will describe now.

Kernels and matrix notation

An operator Ψ is said to be an *Hilbert-Schmidt integral operator* (HSIO) if there exists a function $k \in L_2(\mathbb{S} \times \mathbb{S})$ (called the *kernel*) such that

$$\forall f \in L_2(\mathbb{S}),\ \forall s \in \mathbb{S},\ \Psi(f)(s) = \int_{s' \in \mathbb{S}} k(s,s')f(s')ds'.$$

With HSIOs, the analogy with matrices is strengthened and easier to use; e.g. when Ψ has a kernel k then Ψ^* has the kernel: $k^*(s,s') = k(s',s)$ (it is a direct analogue of matrix transposition). Moreover HSIOs have the good property to be *compact*. The compactness of Ψ^k for some $k \geq 0$ was the key technical point used in [AD09a] to prove a theorem similar to our Theorem 17 below. Here the following proposition implies that Ψ^D and $(\Psi^*)^D$ are Hilbert-Schmidt integral operator (with D the constant occurring in the weak progress condition).

Proposition 14. *(Ψ^n and Ψ^{*n} are HSIOs) For every $n \geq D$ there exists a function $k_n \in L_2(\mathbb{S} \times \mathbb{S})$ such that: $\Psi^n(f)(s) = \int_{\mathbb{S}} k_n(s,s')f(s')ds'$ and $\Psi^{*n}(f)(s) = \int_{\mathbb{S}} k_n(s',s)f(s')ds'$.*

This proposition is a straightforward corollary of the following precise lemma (Lemma 16 below) used also in the proof of irreducibility of Ψ and Ψ^* (Proposition 15). To state this lemma we recall from Chapter 2 the definition of the reachability relation and adopt a matrix notation. For $q, q' \in Q$, we denote by $\texttt{Reach}(n,q,q')$ the set of couple $(\vec{\gamma},\vec{\gamma}')$ such that $(q',\vec{\gamma}')$ is reachable in n steps from $(q,\vec{\gamma})$; formally:

$$\texttt{Reach}(n,q,q') =_{\text{def}} \{(\vec{\gamma},\vec{\gamma}') \in \Gamma_q \times \Gamma_{q'} \mid \exists \vec{\alpha} \in \mathbb{A}^n,\ (q,\vec{\gamma}) \triangleright \vec{\alpha} = (q',\vec{\gamma}')\}.$$

It is convenient to adopt the following matrix notation: each function f of $L_2(\mathbb{S})$ is represented by a row vector (also written f) of functions $f_q \in L_2(\Gamma_q)$. The operator Ψ is represented as a $Q \times Q$ matrix $[\Psi]$ for which each entry $[\Psi]_{q,q'}$ is an operator from $L_2(\Gamma_{q'})$ to $L_2(\Gamma_q)$. Action of $[\Psi]$ on f is given by the following formula:

$$\forall i \in Q,\ ([\Psi]f)_i = \sum_{j \in Q} [\Psi]_{ij} f_j.$$

With this matrix notation the matrix for Ψ^* is simply defined by: for all $i,j \in Q$, $[\Psi^*]_{ij} = ([\Psi]_{ji})^*$.

Now we can state the technical lemma describing the kernels of the operators $[\Psi^n]_{ij}$ for $n \geq D$.

Lemma 16. *For every $i,j \in Q$ and $n \geq D$, the operator $[\Psi^n]_{ij} : L_2(\Gamma_i) \to L_2(\Gamma_j)$ has a kernel $k_{n,i,j} \in L_2(\Gamma_{\mathbf{r}} \times \Gamma_{\mathbf{r}'})$ positive almost everywhere in $\texttt{Reach}(n,i,j)$, continuous and piecewise polynomial.*

Proof. We first introduce a notation for the successor of a vector $\vec{\gamma}$ by a delay vector $\vec{t} \in [0,M]^n$. Let $\pi = \delta_1 \cdots \delta_n$ be a path from a location q to a location q', $\vec{\gamma} \in \Gamma_q$, $\vec{t} \in [0,M]^n$

and $\vec{\alpha} = (t_1, \delta_1) \dots (t_n, \delta_n)$. If there exists $\vec{\gamma}'$ such that $(q, \vec{\gamma}) \triangleright \vec{\alpha} = (q', \vec{\gamma}')$ then we define $\vec{\gamma} \triangleright_\pi \vec{t} = \vec{\gamma}'$ else we define $\vec{\gamma} \triangleright_\pi \vec{t} = \bot$.

We denote by $P_\pi(\vec{\gamma})$ the polytope of delay vector $\vec{t}$ that can be read from $\vec{\gamma}$ along π i.e. $P_\pi(\vec{\gamma}) = \{\vec{t} \mid \vec{\gamma} \triangleright_\pi \vec{t} \neq \bot\}$. We denote by $\texttt{Reach}(\pi) = \{(\vec{\gamma}, \vec{\gamma} \triangleright_\pi \vec{t}) \mid \vec{\gamma} \triangleright_\pi \vec{t} \neq \bot\}$. We also define an operator Ψ_π as follows.

$$\Psi_\pi(f)(\vec{\gamma}) = \int_{P_\pi(\vec{\gamma})} f(\vec{\gamma} \triangleright_\pi \vec{t}) d\vec{t}. \tag{4.19}$$

Then Ψ^n can be decomposed into a sum of operators Ψ_π as follows:

$$[\Psi^n]_{qq'} f_{q'}(\vec{\gamma}) = \sum_{\pi | \pi \text{ goes from } q \text{ to } q' \text{ and } |\pi| = n} \Psi_\pi(f_{q'})(\vec{\gamma}).$$

Now it suffices to prove that if π is a path leading from q to q' and such that $|\pi| = n \geq D$ then Ψ_π has a kernel k_π which is piecewise polynomial and non-zero in $\texttt{Reach}(\pi)$.

The idea of the proof is to operate a change of coordinates which transforms several time delays of $\vec{t}$ into the vector $\vec{\gamma}'$. Let d' be the dimension of the ending region $\mathbf{r}_{q'}$. In $\mathbf{r}_{q'}$, there are d' non zero clocks with pairwise distinct fractional parts which correspond to coordinates of $\vec{\gamma}'$. We sort them as follows $y^1 < \dots < y^{d'}$. By the D weak progress condition, at most one clock is not reset during π, this must be the greatest, i.e. $y^{d'}$. If $y^{d'}$ was not reset along π its value is of the form $y^{d'} = x + \sum_{i=i_{d'}}^n t_i$ where $i_{d'} = 1$ and x is a clock (possibly null) of the starting region $\mathbf{r}_p$, otherwise it is of the form $y^{d'} = \sum_{i=i_{d'}}^n t_i$ where $i_{d'} - 1 \in \{1, \dots n-1\}$ is the index of the transition where $y^{d'}$ was reset for the last time. Similarly for the other clocks we define $i_1 > i_2 > \dots > i_{d'}$ where for each $l \in \{1, \dots, d'-1\}$, $i_l - 1$ is the index of the transition where y^l was reset for the last time. We have thus $y^l = \sum_{i=i_l}^n t_i$.

The change of coordinates consists in replacing coordinates indexed by $I =_{\text{def}} \{i_1, \dots, i_{d'}\}$ by $\vec{\gamma}' = \vec{\gamma} \triangleright_\pi \vec{t}$ and by staying unchanged coordinates in $\bar{I} =_{\text{def}} \{1, \dots, n\} \setminus I$. With few symbols: $\vec{t} = (\vec{t}_{\bar{I}}, \vec{t}_I)_I \mapsto (\vec{t}_{\bar{I}}, \vec{\gamma}')_I$ where $\vec{c} = (\vec{a}, \vec{b})_I$ means that $\vec{b}$ are coordinates of $\vec{c}$ indexed by I while $\vec{a}$ are the others. This change of coordinates preserves the volumes as shown in the following paragraph.

Firstly, it is easy to see that the function which maps $\vec{t}_I = (t_{i_1}, \dots, t_{i_{d'}})$ to $(y^1, \dots, y^{d'})$ is a volume preserving transformation. Indeed, it holds that $(y^1, \dots, y^{d'})^\top = M(t_{i_1}, \dots, t_{i_{d'}})^\top + \vec{b}$ where M is an upper triangular matrix with only 1 on the diagonal and where $\vec{b}$ is a row vector. Secondly, reordering sorted clocks $y^1 < \dots < y^{d'}$ into sorted fractional part $\gamma'_1 < \dots < \gamma'_{d'}$ can be achieved by a translation and a permutation of coordinates.

Now, let us consider the domains of integration before and after the change of coordinates. The old domain of integration is $P_\pi(\vec{\gamma}) = \{\vec{t} \mid \vec{\gamma} \triangleright_\pi \vec{t} \neq \bot\}$, this domain is a polytope. We denote by $P'_\pi(\vec{\gamma})$ the new domain of integration i.e. $(\vec{t}_{\bar{I}}, \vec{\gamma}')_I \in P'_\pi(\vec{\gamma})$ iff $(\vec{t}_{\bar{I}}, \vec{t}_I)_I \in P_\pi(\vec{\gamma})$.

When we fix $(\vec{\gamma}, \vec{\gamma}') \in \texttt{Reach}(\pi)$ we denote by $P_\pi(\vec{\gamma}, \vec{\gamma}')$ the set of vectors $\vec{t}_{\bar{I}}$ such that $(\vec{t}_{\bar{I}}, \vec{\gamma}')_I \in P'_\pi(\vec{\gamma})$. This corresponds intuitively to the set of timed vectors which lead from $\vec{\gamma}$ to $\vec{\gamma}'$. Applying the change of coordinates in (4.19) yields

$$\Psi_\pi(f)(\vec{\gamma}) = \int_{\Gamma_j} \left(\int_{P_\pi(\vec{\gamma}, \vec{\gamma}')} \mathbf{1}_{\vec{t}_{\bar{I}} \in P_\pi(\vec{\gamma}, \vec{\gamma}')} d\vec{t}_{\bar{I}} \right) f(\vec{\gamma}') d\vec{\gamma}'$$

The expected form of Ψ_π is obtained by defining the kernel as

$$k_\pi(\vec{\gamma},\vec{\gamma}') = \texttt{Vol}[P_\pi(\vec{\gamma},\vec{\gamma}')] = \int_{P_\pi(\vec{\gamma},\vec{\gamma}')} \mathbf{1}_{\vec{t}_{\bar{I}} \in P_\pi(\vec{\gamma},\vec{\gamma}')} d\vec{t}_{\bar{I}}.$$

It remains to prove that this kernel is piecewise polynomial and non null when $(\vec{\gamma},\vec{\gamma}') \in \texttt{Reach}(\pi)$. It holds that $(\vec{\gamma},\vec{\gamma}') \in \texttt{Reach}(\pi)$ if and only if the set $P_\pi(\vec{\gamma},\vec{\gamma}')$ is non empty. In this case $P_\pi(\vec{\gamma},\vec{\gamma}')$ is moreover an open polytope (a polytope involving strict inequalities) as a section of the open polytope $P'_\pi(\vec{\gamma})$. Its volume is thus positive and so is $k_\pi(\vec{\gamma},\vec{\gamma}')$.

The polytope $P_\pi(\vec{\gamma},\vec{\gamma}')$ can be defined by a conjunction of inequalities of the following form: $\sum_{i\in\bar{I}} a_i t_i + \sum_{i=1}^{d} b_i\gamma_i + \sum_{i=1}^{d'} c_i\gamma'_i > e$ with $a_i,\ b_i,\ c_i,\ e \in \mathbb{N}$. The volume of such a polytope (when integrating the t_i) can be shown to be piecewise polynomial and continuous in $\vec{\gamma}$ and $\vec{\gamma}'$. We conclude that k_π is piecewise polynomial, continuous and non null on $\texttt{Reach}(\pi)$.

□

When $\texttt{Reach}(n,q,q') = \Gamma_q \times \Gamma_{q'}$, this lemma ensures that the kernel $k_{n,i,j}$ is positive almost everywhere in the whole set $\Gamma_q \times \Gamma_{q'}$. This case, useful in the following, occurs for some $n \geq D$ as stated by the two lemmas just below.

Lemma 17. *For every $q,q' \in Q$, there exists $n \geq D$ such that $\texttt{Reach}(n,q,q') = \Gamma_q \times \Gamma_{q'}$.*

Proof. This lemma is a direct consequence of results of Chapter 3. A path π from q to q' is called forgetful if $\texttt{Reach}(\pi) = \Gamma_{\mathbf{r}_q} \times \Gamma_{\mathbf{r}_{q'}}$ where $\texttt{Reach}(\pi)$ is the reachability relation restrained to π defined in the proof of lemma 16. Every path which contains a forgetful cycle is forgetful. If $\mathcal{G}$ is thick it contains a forgetful cycle f (with $|f| > 0$). Let $l \in Q$ such that f leads from l to l and $\pi,\pi' \in \Delta^*$ such that π leads from q to l and π' leads from l to q'. Such paths exist by strong connectivity of the set of locations. Let $m \geq D$, the path $\pi f^D \pi'$ is forgetful and leads from q to q' and thus $\texttt{Reach}(n,q,q') = \Gamma_q \times \Gamma_{q'}$ with $n = D|f| + |\pi| + |\pi'| \geq D$.

□

Lemma 16 and 17 imply the following one.

Lemma 18. *For every $q,q' \in Q$, there exists $n \geq D$ such that $[\Psi^n]_{qq'}$ has a kernel $k_{n,q,q'}$ positive almost everywhere on $\Gamma_q \times \Gamma_{q'}$.*

The spectral radius ρ and the eigenfunctions v and w.

Before describing ρ, v and w, we recall several definitions from spectral theory. The spectrum of an operator A acting on a functional space $\mathcal{F}$ is the set of scalars $\lambda \in \mathbb{C}$ such that $A - \lambda Id$ is not invertible (where Id is the identity of $\mathcal{F}$). The *spectral radius* of an operator is the radius of the smallest disc centred in the origin and containing all its spectrum. Last but not least, if for some $f \in \mathcal{F} \setminus \{0\}$ and $\lambda \in \mathbb{C}$ it holds that $A(f) - \lambda f = 0$ then λ is called an *eigenvalue* and f is called an *eigenfunction* of A for λ.

As in the discrete case (Proposition 11), the entropy is equal to the logarithm of the spectral radius (Theorem 17 below). This was the main theorem of [AD09a]. We must prove this theorem in our setting since the functional space of [AD09a] was different from ours and assumptions on the model were somewhat more restrictive. We need two lemmas, the first one links the entropy with the norm of the operator (in $L_1(\mathbb{S})$), the second one ensures some regularity of the eigenfunctions of Ψ.

The original intuition of [AD09a] was that the iterates of Ψ on the constant function 1 permit to compute volumes (see also Equations (2.3) and (2.4) in the Chapter 4). More precisely $(\Psi^n 1)(s)$ is equal to the volume of n-length timed words $\vec{\alpha}$ that can be read from s (i.e. $s \triangleright \vec{\alpha} \neq \bot$). Formally $(\Psi^n 1)(s) = \int_{\mathbb{A}^n} \mathbf{1}_{s \triangleright \vec{\alpha} \neq \bot} d\vec{\alpha}$. To get the volume of runs, it suffices to integrate over the state space $\mathbb{S}$ in this equation:

$$||\Psi^n 1||_1 = \int_{\mathbb{S}} (\Psi^n 1)(s) ds = \texttt{Vol}(\mathcal{R}_n). \tag{4.20}$$

As a consequence the entropy of the timed region graph can be defined using the operator Ψ as follows:

Lemma 19. $\mathcal{H}(\mathcal{G}) = \limsup_{n \to +\infty} \frac{1}{n} \log_2(||\Psi^n 1||_1)$.

Lemma 20. *For each eigenvalue $\lambda \neq 0$, each solution f of the eigenfunction equation $\Psi(f) = \lambda f$ (resp. $\Psi^*(f) = \lambda f$) is continuous and bounded*[8].

Proof. Let f be a solution of the eigenfunction equation $\Psi(f) = \lambda f$. Lemma 16 implies that Ψ^D is a kernel operator with a kernel k_D piecewise polynomial (and thus bounded on $\mathbb{S}^2$). The function f satisfies: for almost every s,

$$\Psi^D(f)(s) = \lambda^D f(s) = \int_{\mathbb{S}} k_D(s, s') f(s') ds'.$$

By Lemma 16 $k_{D,i,j}$ is piecewise polynomial continuous and non null on $\texttt{Reach}(D, i, j)$ for every $i, j \in Q$. The function $\vec{\gamma} \mapsto \int_{\Gamma_j} k_{D,i,j}(\vec{\gamma}, \vec{\gamma}') f_j(\vec{\gamma}') d\vec{\gamma}'$ is continuous since the function $\vec{\gamma} \mapsto k_{D,i,j}(\vec{\gamma}, \vec{\gamma}') f_j(\vec{\gamma}')$ is defined and continuous for almost every $\vec{\gamma}' \in \Gamma_j$ and bounded by $\sup(k_{D,i,j}) f_j(\vec{\gamma}')$ for every $\vec{\gamma}' \in \Gamma_j$. Moreover, for every $\vec{\gamma} \in \Gamma_i$, the function $\vec{\gamma} \mapsto \int_{\Gamma_j} k_{D,i,j}(\vec{\gamma}, \vec{\gamma}') f_j(\vec{\gamma}') d\vec{\gamma}'$ is bounded by $\sup(k_{D,i,j})||f||_1$. When summing over $i, j \in Q$ we obtain that $f : s \mapsto \lambda^{-D} \int k_D(s, s') f(s') ds'$ is continuous and bounded (as a finite sum of continuous and bounded functions).

A similar proof can be written for Ψ^* since it has the kernel $k_D^*(s', s) = k_D(s, s')$.

□

Now, we can state the theorem which gives the definition and the first properties of ρ used to defined the maximal entropy SPOR (4.9). The objects v and w are also introduced here, yet their uniqueness (up to a scalar constant) is postponed to the next theorem (Theorem 18).

[8] To be more formal, f as an element of $L_2(\mathbb{S})$ is a class of functions pairwise equal almost everywhere, it admits a unique representative that is continuous and bounded.

Theorem 17 (adapted from [AD09a] to $L_2(\mathbb{S})$)**.** *The spectral radius ρ is a positive eigenvalue for Ψ (resp. Ψ^*) with an eigenfunction $v \geq 0$ (resp. $w \geq 0$). Moreover it holds that $\mathcal{H}(\mathcal{G}) = \log_2(\rho)$.*

Proof. We adapt to the functional space $L_2(\mathbb{S})$ the proof of the main theorem of [AD09a].

Proof that $\mathcal{H}(\mathcal{G}) \leq \log_2 \rho$ The so-called Gelfand formula gives

$$\rho = \lim_{n\to\infty} ||\Psi^n||_2^{\frac{1}{n}}$$

where we recall that $||\Psi^n||_2 = \sup_{f\in L_2(\mathbb{S}),||f||_2>0} ||\Psi^n f||_2/||f||_2$. In particular we have

$$||\Psi^n 1||_2 \leq ||\Psi^n||_2 ||1||_2$$

and thus

$$\limsup_{n\to\infty} \frac{\log(||\Psi^n 1||_2)}{n} \leq \log_2 \rho.$$

We conclude that

$$\mathcal{H}(\mathcal{G}) = \limsup_{n\to\infty} \frac{\log(||\Psi^n 1||_1)}{n} \leq \limsup_{n\to\infty} \frac{\log(||\Psi^n 1||_2)}{n} \leq \log_2 \rho.$$

where the first equality is Lemma 19 and the first inequality comes from the Cauchy-Schwartz inequality:

$$||\Psi^n 1||_1 \leq ||\Psi^n 1||_2 ||1||_2 = ||\Psi^n 1||_2 \sqrt{\texttt{Vol}(\mathbb{S})}.$$

Proof that ρ is a positive eigenvalue for Ψ and Ψ^* By the preceding part of the proof and using the hypothesis $\mathcal{H}(\mathcal{G}) > -\infty$ we have $\rho \geq 2^{\mathcal{H}}(\mathcal{G}) > 0$. According to Theorem 9.3 of [KLS89], a necessary condition for the spectral radius (when it is positive) to be an eigenvalue of an operator A with an eigenfunction $f \geq 0$ is the compactness of some power A^n of A. This is ensured by proposition 14 as HSIOs are compact operators. Thus there exists $v \geq 0$ such that $\Psi(v) = \rho v$ and $w \geq 0$ such that $\Psi^*(w) = \rho w$.

Proof that $\log_2 \rho = \mathcal{H}(\mathcal{G})$ Lemma 20 ensures that the eigenfunction v defined above is continuous and bounded (everywhere). Let C be an upper bound for v i.e. a positive constant such that $\forall s \in \mathbb{S},\ 0 \leq v(s) < C$. Therefore:

$$\forall s \in \mathbb{S},\ n \in \mathbb{N},\ \rho^n v(s) = \Psi^n(v)(s) \leq C\Psi^n(1)(s). \tag{4.21}$$

Integrating wrt. s we get:

$$0 < \rho^n ||v||_1 \leq C||\Psi^n(1)||_1 = C\texttt{Vol}(\mathcal{R}_n).$$

Taking $\liminf_{n\to\infty} \frac{1}{n}\log(.)$ we obtain:

$$\log_2 \rho \leq \liminf_{n\to\infty} \frac{1}{n}\log(\texttt{Vol}(\mathcal{R}_n)) \leq \limsup_{n\to\infty} \frac{1}{n}\log(\texttt{Vol}(\mathcal{R}_n)) = \mathcal{H}(\mathcal{G}) \leq \log_2 \rho \tag{4.22}$$

where the last inequality comes from the first part of the proof. Thus all inequalities of (4.22) are equalities and we conclude that $\log_2 \rho = \mathcal{H}(\mathcal{G})$.

□

Uniqueness of v and w

Theorem 18 (Perron-Frobenius like theorem for Ψ)**.** *The spectral radius ρ is a simple eigenvalue of Ψ and Ψ^* with corresponding eigenfunction $v \geq 0$ and $w \geq 0$. Any non-negative eigenfunction of Ψ (resp. Ψ^*) is proportional to v (resp. w).*

Thus eigenfunctions v and w introduced in Theorem 17 are unique up to a scalar constant. The constants are chosen such that $\langle w, v\rangle = 1$. This makes the functions of (4.9) well defined provided that v is positive. Positivity of v (and w) is the purpose of the next section (section 4.3.3).

Expressing ρ, v, w as solutions of integral equations with kernel It is worth mentioning that for any $n \geq D$, the objects ρ, v and w are solutions of the eigenvalue problems $\int_{\mathbb{S}} k_n(s,s')v(s')ds' = \rho^n v(s)$ and $\int_{\mathbb{S}} k_n(s',s)w(s')ds' = \rho^n w(s)$ with $v \geq 0$ and $w \geq 0$; uniqueness of v and w (up to a scalar constant) is ensured by Theorem 18. The matrix notation, where we denote as in Lemma 16 by $k_{n,q,q'}$ the kernel of $[\Psi^n]_{qq'}$, gives a system of integral equations for v and ρ:

$$\sum_{q' \in Q} \int_{\Gamma_{q'}} k_{n,q,q'}(\vec{\gamma}, \vec{\gamma}')v_{q'}(\vec{\gamma}')d\vec{\gamma}' = \rho^n v_q(\vec{\gamma}), \quad \text{for } q \in Q,\ \vec{\gamma} \in \Gamma_q \tag{4.23}$$

and another system for w and ρ:

$$\sum_{q \in Q} \int_{\Gamma_q} k_{n,q,q'}(\vec{\gamma}, \vec{\gamma}')w_q(\vec{\gamma})d\vec{\gamma} = \rho^n w_{q'}(\vec{\gamma}'), \quad \text{for } q' \in Q,\ \vec{\gamma}' \in \Gamma_{q'}. \tag{4.24}$$

Further computability issues for $\rho,\ v$ and w are discussed in the conclusion.

Proof of theorem 18 The proof of Theorem 18 is based on theorem 11.1 condition e) of [KLS89] (recalled in Theorem 19 below) which is a generalization of the Perron-Frobenius theorem to positive linear operators. The main hypothesis to prove is the irreducibility of Ψ whose analogue in the discrete case is the irreducibility of the adjacency matrix M. Recall from section 4.1 that M is irreducible if for all states i, j there exists $n \geq 1$ such that $M^n_{ij} > 0$ (this is equivalent to the strong connectivity of the graph).

In the following we denote by $L_2^+(\mathbb{S})$ the subset of $L_2(\mathbb{S})$ of functions $f \geq 0$. The operator Ψ is said to be *irreducible* if the following condition holds: if $\Psi(f) \leq af$ for some $a > 0$ and $f \in L_2^+(\mathbb{S}) \setminus \{0\}$, then f is *quasi-interior* (which means that $\langle f, g\rangle > 0$ for every $g \in L_2^+(\mathbb{S}) \setminus \{0\}$).

The irreducibility of Ψ and Ψ^* (Proposition 15 below) is essentially due to the strong connectivity of the state space $\mathbb{S}$ which also mean the positivity of kernels between every two locations q, q' (Lemma 18 above).

Proposition 15. *Ψ and Ψ^* are irreducible.*

Proof. Let $f \in L_2^+(\mathbb{S}) \setminus \{0\}$ and $a > 0$ such that $\Psi(f) \leq af$. Let $g \in L_2^+(\mathbb{S}) \setminus \{0\}$; we show that $\langle f, g \rangle > 0$. There are $i, j \in Q$ such that $g_i \in L_2^+(\Gamma_i) \setminus \{0\}$, $f_j L_2^+(\Gamma_j) \setminus \{0\}$. By Lemma 18 there exists an n such that $[\Psi^n]_{ij}$ has a kernel $k_{n,i,j}$ positive almost everywhere and thus $[\Psi^n]_{ij}(f_j)(s) = \int_{\Gamma_j} k_{n,i,j}(s, s')f(s')ds' > 0$ for almost every s. Therefore $\langle [\Psi^n]_{ij} f_j, g_i \rangle > 0$. We are done:

$$a^n \langle f, g \rangle \geq \langle \Psi^n f, g \rangle \geq \langle [\Psi^n]_{ij} f_j, g_i \rangle > 0.$$

This also prove the irreducibility of Ψ^* since $k^*_{n,i,j}(s, s') = k_{n,j,i}(s', s)$. □

The conclusions of Theorem 17 give the hypotheses of theorem 11.1 condition e) of [KLS89] (Theorem 19 below). We define the cone $K = L_2^+(\mathbb{S})$. It satisfies $\Psi(K) \subseteq K$, it is minihedral ([KLS89]6.1 example d)) and is reproducing i.e. all functions $f \in L_2(\mathbb{S})$ can be written as $f = f^+ - f^-$ with $f^-, f^+ \in K$. The conclusions of this last theorem complete the proof of our theorem.

Theorem 19 ([KLS89], theorem 11.1 condition e))**.** *Suppose that $\Psi K \subseteq K$, Ψ has a normalized eigenfunction $v \in K$ with corresponding eigenvalue ρ (where ρ is the spectral radius of Ψ), K is reproducing and minihedral, the operator Ψ is irreducible and the operator Ψ^* has an eigenfunction w in K^* for the eigenvalue ρ. Then the eigenvalue is simple and there is no other normalized eigenfunction different from v in K.*

Positivity of v and w

Proposition 16. *The eigenfunction v of Ψ (resp. w of Ψ^*) for ρ is positive almost everywhere.*

Proof. As $v \in L_2^+(\mathbb{S}) \setminus \{0\}$, there exists q' such that $v_{q'}$ is non-null on $\Gamma_{q'}$. Let $q \in Q$. We show that v_q is positive almost everywhere. By Lemma 18, there exists $n \geq D$ such that $k_{n,q,q'}(\vec{\gamma}, \vec{\gamma}')$ is positive almost everywhere and thus $k_{n,q,q'}(\vec{\gamma}, \vec{\gamma}')v_{q'}(\vec{\gamma}')$ is non-negative and non-null almost everywhere. We deduce using (4.23) that $v_q(\vec{\gamma})$ is positive for almost every $\vec{\gamma} \in \Gamma_q$. The proof can be adapted for Ψ^* and w using (4.24) instead of (4.23). □

4.3.4 Examples

Running example completed

We consider again the timed region graph depicted in Figure 2.2. The matrix notation of (4.12) is:

$$[\Psi] \begin{pmatrix} f_{\mathbf{r}_p} \\ f_{\mathbf{r}_q} \end{pmatrix} = \begin{pmatrix} \gamma \mapsto \int_\gamma^1 f_{\mathbf{r}_p}(\gamma')d\gamma' + \int_0^1 f_{\mathbf{r}_q}(\gamma')d\gamma' \\ \gamma \mapsto \int_0^1 f_{\mathbf{r}_p}(\gamma')d\gamma' + \int_\gamma^1 f_{\mathbf{r}_q}(\gamma')d\gamma' \end{pmatrix}$$

We can deduce that operators Ψ and Ψ^* are HSIO with matrices of kernels:

$$\begin{pmatrix} \mathbf{1}_{0<\gamma\leq\gamma'<1} & \mathbf{1}_{0<\gamma'<1} \\ \mathbf{1}_{0<\gamma'<1} & \mathbf{1}_{0<\gamma\leq\gamma'<1} \end{pmatrix}; \begin{pmatrix} \mathbf{1}_{0<\gamma'\leq\gamma<1} & \mathbf{1}_{0<\gamma'<1} \\ \mathbf{1}_{0<\gamma'<1} & \mathbf{1}_{0<\gamma'\leq\gamma<1} \end{pmatrix}.$$

The eigenvalue equations $[\Psi]v = \rho v$ and $[\Psi^*]w = \rho w$ written in the form of (4.23) and (4.24) (for $n = 1$) yield

$$\begin{aligned}
\rho v_{\mathbf{r}_p}(\gamma) &= \int_\gamma^1 v_{\mathbf{r}_p}(\gamma')d\gamma' + \int_0^1 v_{\mathbf{r}_q}(\gamma')d\gamma';\\
\rho v_{\mathbf{r}_q}(\gamma) &= \int_0^1 v_{\mathbf{r}_p}(\gamma')d\gamma' + \int_\gamma^1 v_{\mathbf{r}_q}(\gamma')d\gamma';\\
\rho w_{\mathbf{r}_p}(\gamma) &= \int_0^\gamma w_{\mathbf{r}_p}(\gamma')d\gamma' + \int_0^1 w_{\mathbf{r}_q}(\gamma')d\gamma';\\
\rho w_{\mathbf{r}_q}(\gamma) &= \int_0^1 w_{\mathbf{r}_p}(\gamma')d\gamma' + \int_0^\gamma w_{\mathbf{r}_q}(\gamma')d\gamma'.
\end{aligned}$$

We differentiate once the equations and obtain:

$$\rho v'_{\mathbf{r}_i}(\gamma) = -v_{\mathbf{r}_i}(\gamma); \rho w'_{\mathbf{r}_i}(\gamma) = w_{\mathbf{r}_i}(\gamma) \ (i \in \{p, q\}).$$

Thus the functions are of the form $v_{\mathbf{r}_i}(\gamma) = v_{\mathbf{r}_i}(0)e^{-\gamma/\rho}$, $w_{\mathbf{r}_i}(\gamma) = w_{\mathbf{r}_i}(0)e^{\gamma/\rho}$. Remark that $\rho v_{\mathbf{r}_p}(0) = \int_0^1 v_{\mathbf{r}_p}(\gamma')d\gamma' + \int_0^1 v_{\mathbf{r}_q}(\gamma')d\gamma' = \rho v_{\mathbf{r}_q}(0)$ and thus $v_{\mathbf{r}_p} = v_{\mathbf{r}_q}$ (we can divide by ρ which is positive since $\rho = 2^{\mathcal{H}(\mathcal{G})}$ and the timed region graph is thick i.e. $\mathcal{H}(\mathcal{G}) > -\infty$).

Similarly $w_{\mathbf{r}_p}(1) = w_{\mathbf{r}_q}(1)$ yields $w_{\mathbf{r}_p} = w_{\mathbf{r}_q}$.

The constant ρ satisfies the condition

$$v_{\mathbf{r}_p}(0) = 2\int_0^1 v_{\mathbf{r}_p}(\gamma')d\gamma'/\rho = 2v_{\mathbf{r}_p}(1) = 2v_{\mathbf{r}_p}(0)e^{-1/\rho}.$$

Therefore $e^{1/\rho} = 2$ and thus $\rho \in \{1/(\ln(2) + i2k\pi) \mid k \in \mathbb{Z}\}$. The spectral radius is the eigenvalue of maximal modulus corresponding to $k = 0$, $\rho = 1/\ln(2)$. Then the eigenfunctions are $v = \begin{pmatrix} v_{\mathbf{r}_p}(\gamma) \\ v_{\mathbf{r}_q}(\gamma) \end{pmatrix} = C \begin{pmatrix} 2^{-\gamma} \\ 2^{-\gamma} \end{pmatrix}$ with $C > 0$ and $w = \begin{pmatrix} w_{\mathbf{r}_p}(\gamma) \\ w_{\mathbf{r}_q}(\gamma) \end{pmatrix} = C' \begin{pmatrix} 2^{\gamma} \\ 2^{\gamma} \end{pmatrix}$ with $C' > 0$.

Finally the maximal entropy SPOR for $\mathcal{G}^{\texttt{ex1}}$ is given by:

$$\begin{aligned}
&p_0^*(p, (\gamma, 0)) = p_0^*(q, (0, \gamma)) = \frac{1}{2} \text{ for } \gamma \in (0, 1);\\
&p^*(t, \delta^1 | p, (\gamma, 0)) = p^*(t, \delta^4 | q, (0, \gamma)) = \frac{2^{-t}}{\rho} \text{ for } \gamma \in (0, 1), t \in [0, 1 - \gamma);\\
&p^*(t, \delta^2 | p, (\gamma, 0)) = p^*(t, \delta^3 | q, (0, \gamma)) = \frac{2^{\gamma - t}}{\rho} \text{ for } \gamma \in (0, 1), t \in (0, 1).
\end{aligned}$$

Running example of the thesis

Consider the TRG $\mathcal{G}_3$ depicted in Figure 3.1 from Chapter 3 also related to timed automata depicted in Figure 1.1, 2.1 and 2.3. The entry regions of this TRG are $\mathbf{r}_p = \{(x, y) \mid 0 = y < x < 1\}$ and $\mathbf{r}_q = \{(x, y) \mid 0 = x < y < 1\}$.

The operators Ψ and Ψ^* are equal[9]. Indeed they are HSIOs with the same matrices of kernels:

$$\begin{pmatrix} 0 & \mathbf{1}_{0<\gamma'<1-\gamma<1} \\ \mathbf{1}_{0<\gamma'<1-\gamma<1} & 0 \end{pmatrix}$$

The maximal entropy SPOR of $\mathcal{G}_3$ is given by the following PDFs:

$$p_0^*(p,(\gamma,0)) = p_0^*(q,(0,\gamma)) = \cos^2\left(\frac{\pi}{2}\gamma\right) \text{ for } \gamma \in (0,1);$$

$$p^*(t,a|p,(\gamma,0)) = p^*(t,b|q,(0,\gamma)) = \frac{\pi}{2}\frac{\cos(\frac{\pi}{2}t)}{\cos(\frac{\pi}{2}\gamma)}\mathbf{1}_{t<1-\gamma} \text{ for } \gamma \in (0,1), t \in [0,1-\gamma);$$

4.3.5 Proof of the maximal entropy theorem (Theorem 14)

We give the proof of Theorem 14 in several steps

Proof that Y^* is a SPOR

The eigenfunctions v and w are positive almost everywhere and are chosen such that $\int_{\mathbb{S}} p_0^*(s) = \langle v, w\rangle = 1$. Moreover $v(s \triangleright \alpha) = 0$ when $s \triangleright \alpha = \perp$ and thus $p(\alpha|s)$ is defined for almost every $s \in \mathbb{S}$, $\alpha \in \mathbb{A}$ and equals 0 when $s \triangleright \alpha = \perp$. Finally for almost every $s \in \mathbb{S}$: $\int_{\mathbb{A}} p^*(\alpha|s)d\alpha = \int_{\mathbb{A}} \frac{v(s\triangleright\alpha)}{\rho v(s)}d\alpha = \frac{\Psi(v)(s)}{\rho v(s)} = 1$ since v is an eigenfunction for ρ.

Proof that Y^* is stationary

Proof. We use the characterization of stationarity given in Proposition 12. For every measurable set of states $\mathcal{S} \in \mathfrak{B}(\mathbb{S})$,

$$\begin{aligned}
\int_{\mathbb{S}}\int_{\mathbb{A}} p_0(s)p(\alpha|s)1_{s\triangleright\alpha\in\mathcal{S}}d\alpha ds &= \int_{\mathbb{S}}\int_{\mathbb{A}} v(s)w(s)\frac{v(s\triangleright\alpha)}{\rho v(s)}1_{s\triangleright\alpha\in\mathcal{S}}d\alpha ds \\
&= \int_{\mathbb{S}} w(s)\int_{\mathbb{A}} v(s\triangleright\alpha)1_{s\triangleright\alpha\in\mathcal{S}}d\alpha ds/\rho \\
&= \langle w, \Psi(v1_{\mathcal{S}})\rangle/\rho \\
&= \langle \Psi^*(w), v1_{\mathcal{S}}\rangle/\rho \quad \text{by definition of } \Psi^* \text{ see (4.18)} \\
&= \langle w, v1_{\mathcal{S}}\rangle \ (w \text{ is an eigenfunction of } \Psi^* \text{ for } \rho) \\
&= \int_{\mathbb{S}} p_0(s)1_{s\in\mathcal{S}}ds.
\end{aligned}$$

□

[9]Such a self adjoint operator (i.e. $\Psi = \Psi^*$) in a Hilbert space has nice properties.

Proof that $H(Y^*) = \mathcal{H}(\mathcal{G})$

$$\begin{aligned}
H(Y^*) &= -\int_{\mathbb{S}} p_0(s) \int_{\mathbb{A}} p(\alpha|s) \log_2 p(\alpha|s) d\alpha ds \\
&= -\int_{\mathbb{S}} v(s)w(s) \int_{\mathbb{A}} \frac{v(s \triangleright \alpha)}{\rho v(s)} \log_2 \frac{v(s \triangleright \alpha)}{\rho v(s)} d\alpha ds \\
&= -\frac{1}{\rho} \int_{\mathbb{S}} w(s) \int_{\mathbb{A}} v(s \triangleright \alpha)[\log_2 v(s \triangleright \alpha) - \log_2(\rho v(s))] d\alpha ds \\
&= -\frac{1}{\rho} \langle w, \Psi(v \log_2 v) \rangle + \frac{1}{\rho} \langle w \log_2 v, \Psi(v) \rangle + \frac{\log_2 \rho}{\rho} \langle w, \Psi(v) \rangle \\
&= -\frac{1}{\rho} \langle \Psi^*(w), v \log_2 v \rangle + \langle w \log_2 v, v \rangle + \log_2(\rho) \langle w, v \rangle \ \text{ (since } \Psi(v) = \rho v) \\
&= -\langle w, v \log_2 v \rangle + \langle w \log_2 v, v \rangle + \log_2(\rho) \ \ (\langle w, v \rangle = 1 \text{ and } \Psi^*(w) = \rho w) \\
&= \log_2(\rho) = \mathcal{H}(\mathcal{G}).
\end{aligned}$$

Ergodicity of Y^*

We first introduce a "stochastic" operator φ which is the continuous analogue of a stochastic matrix. Then we relate this operator with Y^* (Equation (4.25) and Proposition 17) and prove an ergodic property on φ (Proposition 19). This property permits to prove the ergodicity of Y^*.

The operator φ defined below acts on the functional space $L_2(v^2 ds)$ of function f such that $fv \in L_2(\mathbb{S})$. The dual space of $L_2(v^2 ds)$ is the set of functions g such that $g/v \in L_2(\mathbb{S})$. The norm on $L_2(v^2 ds)$ is $||f||_{L_2(v^2 ds)} = ||fv||_2$.

Let $\varphi : L_2(v^2 ds) \to L_2(v^2 ds)$ be the linear operator defined by $\varphi(f) = \frac{\Psi(vf)}{\rho v}$. One can see using the equality $\langle \varphi(f), g \rangle = \langle f, \varphi^*(g) \rangle$ that $\varphi^*(g) = v\Psi^*\left(\frac{g}{\rho v}\right)$. Indeed,

$$\left\langle \frac{\Psi(vf)}{\rho v}, g \right\rangle = \left\langle \Psi(vf), \frac{g}{\rho v} \right\rangle = \left\langle vf, \Psi^*\left(\frac{g}{\rho v}\right) \right\rangle = \left\langle f, v\Psi^*\left(\frac{g}{\rho v}\right) \right\rangle.$$

We have constructed the operator φ by analogy with the transition probability matrix of the Shannon-Parry Markov chain recalled in (4.6).

The operators φ^i with $i \geq 0$ are associated with the conditional PDFs $p_i^*(\vec{\alpha}|s) = p_i^*(\vec{\alpha})/p_0^*(s)$ (defined in (4.7) and characterized in (4.10)) as follows:

Lemma 21. *For every $f \in L_2(v^2 ds)$, $s \in \mathbb{S}$, the following equality holds:*

$$\varphi^i(f)(s) = \int_{\vec{\alpha} \in \mathbb{A}^i} p_i^*(\vec{\alpha}|s) f(s \triangleright \vec{\alpha}) d\vec{\alpha}. \tag{4.25}$$

Proof. By a straightforward induction we have that $\varphi^i(f) = \frac{\Psi^i(vf)}{\rho^i v}$. Then $\varphi^i(f)(s) = \int_{\mathbb{A}^i} \frac{v(s \triangleright \vec{\alpha})}{\rho^i v(s)} f(s \triangleright \vec{\alpha}) ds$ which is equal to the expected result since by virtue of (4.10), $p_i^*(\vec{\alpha}|s) = \frac{v(s \triangleright \vec{\alpha})}{\rho^i v(s)}$. □

It is also worth mentioning that when Ψ^i has a kernel $k_i(s, s_i)$ then φ has the kernel $p_i^*(s, s_i) =_{\text{def}} \frac{v(s_i)}{\rho^i v(s)} k_i(s, s_i)$. With this notation,

$$\varphi^i(f)(s) = \int_{\mathbb{S}} p_i^*(s, s_i) f(s_i) ds_i. \tag{4.26}$$

Thus $p_i^*(s, s_i)$ is the (density of) probability that $S_i^* = s_i$ knowing that $S_0^* = s$ and $\varphi^i(f)(s)$ can be interpreted as the expectation of the random variable $f(S_i^*)$ knowing that $S_0^* = s$. The ergodic property for φ (Proposition 19 below), states that this value converges (in $L_2(v^2 ds)$ and for Cesàro means) towards the constant $\langle f, p_0^* \rangle = \int_{\mathbb{S}} f(s) p_0^*(s) ds$. This constant is the expectation of $f(S_i^*)$ for each $i \in \mathbb{N}$. Thus the initial state of a run generated according to Y^* is "forgotten". Intuitively, this will be the key sufficient condition for the ergodicity of Y^*.

To prove Proposition 19 we need to study the spectral properties of φ. They are analogous to those of the matrix of an irreducible stationary Markov chain on a graph.

Proposition 17. *The spectral radius of φ is 1. It is a simple eigenvalue of φ for which the constant function 1 is an eigenfunction ($\varphi(1) = 1$). Every positive eigenfunction of φ is constant. p_0^* is an eigenfunction of φ^* for 1 ($\varphi^*(p_0^*) = p_0^*$). Every positive eigenfunction of φ^* is proportional to p_0^*.*

Proof. One can see that λ belongs to the spectrum of φ iff λ/ρ belongs to the spectrum of Ψ and thus 1 is the spectral radius of φ.

The functions 1 and p_0^* are eigenfunctions of φ and φ^* for the spectral radius: $\varphi(1) = \frac{\Psi(v)}{\rho v} = 1$ and $\varphi^*(p_0^*) = v\Psi^*\left(\frac{vw}{\rho v}\right) = v\Psi^*(w)/\rho = vw = p_0^*$.

The other properties are ensured by Theorem 19 (already used to prove Theorem 18). □

We need also another property to ensure the convergence of the iterates of φ on a function f.

Proposition 18 (Spectral gap)**.** *Some power φ^p ($p \in \mathbb{N}$) has a spectral gap, i.e. the spectral radius of φ^p is a simple eigenvalue and the rest of its spectrum belongs to the disc $C_\lambda = \{z \| |z| \leq \lambda\}$ for some $\lambda < \rho$.*

Proof. First of all, Proposition 17 just above guarantees that the spectral radius is a simple eigenvalue of φ and thus of φ^p. It remains to prove that the rest of the spectrum of φ^p lies in a disc C_λ with $\lambda < 1$.

We can apply the theorem at the beginning of section 3.4 of the appendix of [SW99]. This theorem states that there exists $p \in \mathbb{N}$ such that every eigenvalue ω of modulus 1 satisfies $\omega^p = 1$ and thus φ^p has only one eigenvalue of modulus 1. The other eigenvalues ω^p of φ^p are such that $\omega^p < \beta$ for some $\beta < 1$ since there is no accumulation point other than 0 (the spectrum of φ^p has the same shape as the spectrum of Ψ^p which has no accumulation point other than 0 since it is compact). Therefore φ^p has a spectral gap β. □

With such a spectral gap, as stated in Lemma 22 just below, iterates of φ^p on a function $f \in L_2^+(v^2 ds)$ converges towards the constant function $\langle f, p_0^* \rangle$.

Lemma 22. *For every $f \in L_2^+(v^2 ds)$ the following holds*

$$\varphi^{pk}(f) \to_{k\to+\infty} \langle f, p_0^* \rangle \ \text{ in } L_2(v^2 ds).$$

Proof. This is ensured by Theorem 15.4 of [KLS89] whose hypothesis is the existence of a gap for φ^p (Proposition 18). □

Proposition 19 (ergodic property for φ)**.** *For every $f \in L_2^+(v^2 ds)$, the following holds*[10]

$$\frac{1}{n}\sum_{i=1}^{n} \varphi^i(f)(s) \to_{n\to+\infty} \langle f, p_0^* \rangle \ \text{ in } L_2(v^2 ds)$$

Proof. We let $g_n(s) = \frac{1}{n}\sum_{i=1}^n \varphi^i(f)(s) - \langle f, p_0^* \rangle$ and show that $||g_n||_{L_2(v^2 ds)}$ converges to 0 as $n \to +\infty$.

It holds that

$$||g_n||_{L_2(v^2 ds)} \leq \sum_{j=1}^{p} \frac{1}{n} \sum_{i=0}^{n-1} ||\varphi^{pi+j}(f) - \langle f, p_0^* \rangle ||_{L_2(v^2 ds)}.$$

Now it suffices to remark that for every $j \in \{1, \ldots, p\}$ the sequence $||\varphi^{pi+j}(f) - \langle f, p_0^* \rangle ||_{L_2(v^2 ds)}$ converges to 0 as $i \to +\infty$ and thus so does its Cesàro mean. This convergence follows from Lemma 22 applied to $\varphi^j(f)$ since $\varphi^{pi+j}(f) = \varphi^{pi}(\varphi^j f)$.

□

As we have already discussed, in some sense Y^* forgets its past. This intuition is made more clear with the following lemma: for Cesàro average and asymptotically, coordinates $Y^*_{m+i}, \ldots, Y^*_{2m+i-1}$ and coordinates $Y^*_0, \ldots, Y^*_{m-1}$ are distributed as if they were independent from each others.

Lemma 23. *Let R, R' be two measurable subsets of $\mathbb{D}^m$ ($m \in \mathbb{N}$) then*

$$\frac{1}{n}\sum_{i=1}^{n} P_{Y^*}(R_\infty \cap \sigma^{m+i}(R'_\infty)) \to_{n\to\infty} P_{Y^*}(R_\infty) P_{Y^*}(R'_\infty).$$

Proof. Let f be the function defined by

$$f(s) = P(Y_0^* \cdots Y_{m-1}^* \in R' | S_0^* = s) = \int_{\mathbb{A}^m} p_m(\vec{\alpha}'|s) 1_{[s,\vec{\alpha}'] \in R'} d\vec{\alpha}'$$

We first prove the two following equations:

$$\frac{1}{n}\sum_{i=1}^{n} P_{Y^*}(R_\infty \cap \sigma^{m+i}(R'_\infty)) = \int_R p_m[s, \vec{\alpha}] \left(\frac{1}{n}\sum_{i=1}^{n} \varphi^i(f)(s \triangleright \alpha) \right) d[s, \vec{\alpha}] \tag{4.27}$$

and

$$P_{Y^*}(R_\infty) P_{Y^*}(R'_\infty) = \int_R p_m[s, \vec{\alpha}] \langle f, p_0^* \rangle \, d[s, \vec{\alpha}]. \tag{4.28}$$

[10] This proposition is akin to von Neumann's mean ergodic theorem (see e.g. Theorem 4.5.2 of [BS02]) whose conclusion is similar to ours (yet the hypotheses differ).

Proof of (4.27) By definition of R_∞ and R'_∞:

$$P_{Y^*}(R_\infty \cap \sigma^{m+i}(R'_\infty)) = P(Y_0^* \cdots Y_{m-1}^* \in R \text{ and } Y_{m+i}^* \cdots Y_{2m+i-1}^* \in R')$$

which is equal to

$$\int_R p_m[s,\vec{\alpha}] P(Y_{m+i}^* \cdots Y_{2m+i-1}^* \in R' | S_m^* = s \triangleright \vec{\alpha}) d[s,\vec{\alpha}].$$

Now, it suffices to prove that for every $s \in \mathbb{S}$ the following equality holds

$$P(Y_{m+i}^* \cdots Y_{2m+i-1}^* \in R' | S_m^* = s) = \varphi^i(f)(s). \tag{4.29}$$

Using characterization (4.25) of $\varphi^i(f)(s)$ we obtain that

$$\varphi^i(f)(s) = \int_{\mathbb{A}^i} p_i(\vec{\alpha}|s) \int_{\mathbb{A}^m} p_m(\vec{\alpha}'|s \triangleright \vec{\alpha}) 1_{[s \triangleright \vec{\alpha}, \vec{\alpha}'] \in R'} d\vec{\alpha}' d\vec{\alpha}$$

which can be rewritten as

$$\varphi^i(f)(s) = \int_{\mathbb{A}^{m+i}} p_{m+i}(\vec{\alpha}|s) 1_{y_i \cdots y_{m+i-1} \in R'} d\vec{\alpha}$$

where $y_0 \cdots y_{m+i-1} \in \mathbb{D}^{m+i}$ denotes the extended version of the run $[s,\vec{\alpha}]$. Thus we obtain the expected equality (4.29) using stationarity of Y^*:

$$P(Y_{m+i}^* \cdots Y_{2m+i-1}^* \in R' | S_m^* = s) = P(Y_i^* \cdots Y_{i+m-1}^* \in R' | S_0^* = s) = \varphi^i(f)(s)$$

Proof of (4.28) By definition of f:

$$\langle f, p_0^* \rangle = \int_{\mathbb{S}} f(s) p_0^*(s) ds = \int_{\mathbb{S}} \int_{\mathbb{A}^m} p_0^*(s) p_m(\vec{\alpha}'|s) 1_{[s,\vec{\alpha}'] \in R'} d\vec{\alpha}' ds = P_{Y^*}(R'_\infty).$$

Thus

$$P_{Y^*}(R_\infty) P_{Y^*}(R'_\infty) = \int_R p_m[s,\vec{\alpha}] P_{Y^*}(R'_\infty) d[s,\vec{\alpha}] = \int_R p_m[s,\vec{\alpha}] \, \langle f, p_0^* \rangle \, d[s,\vec{\alpha}].$$

End of the proof We can complete the proof with the following sequences of inequalities and equalities:

$$\begin{aligned}
&\left|\frac{1}{n}\sum_{i=1}^{n} P_{Y^*}(R_\infty \cap \sigma^{m+i}(R'_\infty)) - P_{Y^*}(R_\infty)P_{Y^*}(R'_\infty)\right| \\
&= \left|\int_R p_m[s,\vec{\alpha}]\left(\frac{1}{n}\sum_{i=1}^{n}\varphi^i(f)(s\triangleright\alpha) - \langle f, p_0^*\rangle\right) d[s,\vec{\alpha}]\right| \quad \text{(by (4.27) and (4.28))} \\
&= \left|\int_R p_m[s,\vec{\alpha}] g_n(s\triangleright\alpha) d[s,\vec{\alpha}]\right| \text{ with } g_n : s' \to \sum_{i=1}^{n}\varphi^i(f)(s') - \langle f, p_0^*\rangle \\
&\leq \int_R p_m[s,\vec{\alpha}]\,|g_n(s\triangleright\alpha)|\,d[s,\vec{\alpha}] \quad \text{(by triangular inequality)} \\
&\leq \int_{\mathbb{S}}\int_{\mathbb{A}^m} p_m[s,\vec{\alpha}]|g_n(s\triangleright\vec{\alpha})|d\vec{\alpha}ds \\
&= \int_{\mathbb{S}} \varphi^m(|g_n|)(s)p(s)ds = \int_{\mathbb{S}} \varphi^m(|g_n|)(s)v(s)w(s)ds \\
&\leq ||w||_\infty \int_{\mathbb{S}} \varphi^m(|g_n|)(s)v(s)ds \text{ (since } w \text{ is bounded by Lemma 20)} \\
&\leq ||w||_\infty ||\varphi^m(|g_n|)v||_2 \sqrt{\texttt{Vol}(\mathbb{S})} \text{ (by Cauchy-Schwartz inequality)} \\
&= ||w||_\infty ||\varphi^m(|g_n|)||_{L_2(v^2ds)} \sqrt{\texttt{Vol}(\mathbb{S})} \\
&\leq ||w||_\infty ||\varphi^m||_{L_2(v^2ds)} ||g_n||_{L_2(v^2ds)} \sqrt{\texttt{Vol}(\mathbb{S})} \to_{n\to+\infty} 0 \text{ (by Proposition 19).}
\end{aligned}$$

□

Now we can achieve the proof that Y^* is ergodic.

Consider a shift invariant set A. We will show that $P_{Y^*}(A) \in \{0,1\}$. We assume that $P_{Y^*}(A) < 1$ and show that $P_{Y^*}(A) \leq P_{Y^*}(A)^2$. These inequalities imply that $P_{Y^*}(A) = 0$.

Using (4.3), for every ϵ, there exists an $m \in \mathbb{N}$ such that

$$P(Y_0^* \cdots Y_{m-1}^* \in A_m) = P_{Y^*}(A_{m,\infty}) \in [P_{Y^*}(A), P_{Y^*}(A) + \epsilon].$$

By set inclusion we have:

$$P_{Y^*}(A) \leq P_{Y^*}(A_{m,\infty} \cap \sigma^{m+i}(A_{m,\infty})).$$

Taking the Cesàro average, we get:

$$P_{Y^*}(A) \leq \frac{1}{n}\sum_{i=1}^{n} P_{Y^*}(A_{m,\infty} \cap \sigma^{m+i}(A_{m,\infty})).$$

Taking the limit and using Lemma 23 we obtain:

$$P_{Y^*}(A) \leq P_{Y^*}(A_{m,\infty})^2 \leq (P_{Y^*}(A) + \epsilon)^2.$$

When ϵ tends to 0, we obtain the expected inequality. This last paragraph completed the proof of Theorem 14.

4.4 Conclusion and perspectives

In this chapter, we proved the existence of an ergodic stochastic process over runs of a timed region graph $\mathcal{G}$ with maximal entropy, provided $\mathcal{G}$ has finite entropy ($\mathcal{H}(\mathcal{G}) > -\infty$) and satisfies the D weak progress condition.

In the introduction, we already motivated the theory of this chapter by potential applications in verification and in coding theory.

4.4.1 Technical challenges

Getting rid of the D-WPC In our recent work [ABD13], we manage to prove the existence of a spectral gap for Ψ without assuming the D-weak progress condition. We think that such a spectral gap suffices to have existence and uniqueness of a maximal entropy SPOR. Nevertheless the functional space of continuous function used in [ABD13] has a dual space which is less intuitive to use (at least for the author) than $L_2(\mathbb{S})$, e.g. the meaning of w in this functional space is still to be understood.

Computing ρ, v, w The next question is to know how simulation can be realized in practice. Symbolic computations of ρ and v have been proposed in [AD09a] for subclasses of deterministic TA, the algorithm can be adapted to compute w. In the same article, an iterative procedure is also given to estimate the entropy $\mathcal{H} = \log_2(\rho)$. We prove in [ABD13] that this procedure converges exponentially fast due to the presence of the spectral gap mentioned above. We think that approximations of ρ, v and w using an iterative procedure on Ψ and Ψ^* would give a SPOR with entropy as close to the maximum as we want. However, several challenges remain to be solved. As described above, we must clarify the link between the present work and [ABD13] and understand for instance what would be the iterates of Ψ^*. Another technical hypothesis we want to get rid of is the decomposition of the state space in regions. This decomposition can lead to an exponential blow-up of the size of the model. In works on timed automata, regions are often replaced by zones which are in practice far less numerous. It is a challenging task for us to define Ψ and then the maximal entropy stochastic process Y^* on a state space decomposed in zones.

Discretizing Y^* Let $\mathcal{G}$ be a timed region graph, if we consider only states with clocks multiples of a discretization step ε we obtain a finite graph $\mathcal{G}_\epsilon$ whose paths represent runs of $\mathcal{G}$ with clocks and delays multiple of ε. This finite graph has a maximal entropy Markov chain $p^*_{\mathcal{G}_\varepsilon}$. It would be interesting to show that when ε tends to 0 the Markov chain $p^*_{\mathcal{G}_\varepsilon}$ gets closer and closer (in a sense we must define) to the maximal entropy SPOR of the timed region graph. This would permit to compute the maximal entropy SPOR of a timed region graph with any required precision.

Chapter 5

Timed symbolic dynamics

Abstract of the chapter

In this chapter we develop a theory of timed symbolic dynamics. This theory explains the recent results of volumetry of timed languages within a symbolic dynamics framework. On the positive side, several results are leveraged from the classical symbolic dynamics such as the definition of timed shift space. On the negative side, several results have no natural extension to the timed case. For instance, we show that the topological entropy is infinite for timed shift spaces and also that there is no correspondence between sliding block codes and morphisms of shift space, contradicting (in the timed case) the so-called Curtis-Hedlund-Lyndon Theorem. We give remedy to each of these negative results. In particular we show how the classical entropy is naturally replaced by the volumetric entropy $\mathcal{H}$ of Asarin and Degorre or by the (parametric) ε-entropy h_ε. The two entropies are related in Theorem 25 by the formula

$$h_\varepsilon = \log_2(1/\varepsilon) + \mathcal{H} + o(1).$$

This solves the problem left open in [AD09b, AD09a] of computability of the entropy. Some timed automata yield languages of n-length timed words of degenerate dimension and thus of null volume. We adapt the metric mean dimension of Lindenstrauss, Weiss and Gromov to the timed setting to measure these non full dimensional language. This measure is more precise than the dichotomy between thin and thick timed languages described in Chapter 3.

Chapter structure

After several preliminaries (section 5.1) we briefly recall in section 5.2 the classical symbolic dynamics. We introduce in section 5.3 compact alphabet shift spaces a general framework containing the timed shift space defined later in section 5.4 devoted to timed sofic shift. In this latter section we discretize the shift space and the entropy (section 5.4.3) and explore the notion of metric mean dimension (section 5.4.4). Section 5.5 is devoted to morphisms of compact alphabet shift spaces. We conclude in section 5.6.

5.1 Preliminaries

5.1.1 Words and factors

In this section, A denotes a finite or infinite set called alphabet. We denote by $A^{\mathbb{Z}}$ the set of bi-infinite words over A i.e. words of the form $x = (x_i)_{i\in\mathbb{Z}}$ with $x_i \in A$. Given a bi-infinite word $x \in A^{\mathbb{Z}}$ and two indices $i, j \in \mathbb{Z}$ with $i \leq j$, the finite word $x_i x_{i+1} \cdots x_j$ is called a factor of x and is denoted by $x_{[i..j]}$. We denote by $\sigma_A : A^{\mathbb{Z}} \to A^{\mathbb{Z}}$ the (left) *shift* map which operates on bi-infinite words x by shifting coordinates to the left i.e. if $y = \sigma_A(x)$ then for all $i \in \mathbb{Z},\ y_i = x_{i+1}$. When A is clear from the context, we will only write σ instead of σ_A.

5.1.2 Topology

We refer the reader to [BS02] and [LM95] for more details.

Metric spaces

A metric space (A, δ) is a set endowed with a *distance* i.e. a mapping from A^2 to $\mathbb{R}^+$ such that $\forall x, y, z \in A,\ \delta(x, y) = \delta(y, x);\ \delta(x, y) \geq 0;\ \delta(x, y) = 0$ iff $x = y;\ \delta(x, z) \leq \delta(x, y) + \delta(y, z)$. The following sets are metric spaces used in this chapter:

- An interval $[0, M]$ where $M \in \mathbb{N} \setminus \{0\}$ and a distance δ_1 such that $\delta_1(x, y) = |x - y|$.
- A finite alphabet Σ and a distance δ_Σ such that $\delta_\Sigma(l, l') = \mathbf{1}_{l,l'} = \begin{cases} 1 & \text{if } l \neq l' \\ 0 & \text{otherwise} \end{cases}$.
- A *timed alphabet* $\mathbb{A} = [0, M] \times \Sigma$ and a distance $\delta_3 = \delta_1 + \delta_\Sigma$ i.e $\delta_3((a, t), (a', t')) = |t - t'| + \mathbf{1}_{a,a'}$

in the last example Σ denotes a finite alphabet whose elements are called *events* and M is an upper bound on delays between pairwise consecutive events. A "letter" (t, a) informally means that the event a is produced after waiting a time t.

The product of a finite set Σ with a metric space A should be understood as several disjoint copies of A. The closed (resp. open) sets of $A \times \Sigma$ are the finite unions of $\{a\} \times F$ where $a \in \Sigma$ and F is a closed (resp. open) subset of A.

If (A, δ) then so is (A^n, δ^n) with δ^n defined by

$$\delta^n[(x_1, \ldots, x_n), (y_1, \ldots, y_n)] = \sup_{i\in\{1,\ldots,n\}} \delta(x_i, y_i).$$

For instance, for a finite alphabet Σ and two words $w, w' \in \Sigma^n$, $\delta^n_\Sigma(w, w') = 1$ iff $w \neq w'$, 0 otherwise.

In the same way $A^{\mathbb{Z}}$ has a distance $\bar{\delta}$ defined by :

$$\bar{\delta}(x, x') = \sup_{i\in\mathbb{Z}} \frac{\delta(x_i, x'_i)}{2^{|i|}}.$$

Remark 3. *A sequence of bi-infinite words converges if and only if all its coordinates converge i.e.* $x^n \to_{n\to\infty} x$ *iff* $\forall i \in \mathbb{Z}, x_i^n \to_{n\to\infty} x_i$.

Compacity, dynamical systems

Let (X, δ) be a metric space. A subset $Y \subseteq X$ is an *ε-net* of X if every element of X is at most ε far apart from an element of Y: $\forall x \in X,\ \exists y \in A$ such that $\delta(x, y) \leq \varepsilon$. X is said to be *pre-compact* if, for all $\varepsilon > 0$, there exists a finite ε-net of it. We denote by $\mathcal{N}_\varepsilon(X, \delta)$ the minimal cardinality of ε-net of X.A set $Y \subseteq X$ is said to be ε-separated if all two different elements of Y are at least ε far apart from each other. $\forall x, y \in Y,\ x \neq y \Rightarrow \delta(x, y) > \varepsilon$. In a pre-compact set, all ε-separated set is of finite cardinality. We denote by $\mathcal{S}_\varepsilon(X, \delta)$ the maximal cardinality of ε-separated[1] sets of X.

Lemma 24 ([KT59], see also [BS02])**.** *Given a pre-compact metric space X the followings inequalities hold* $\mathcal{S}_{2\varepsilon}(X, \delta) \leq \mathcal{N}_\varepsilon(X, \delta) \leq \mathcal{S}_\varepsilon(X, \delta)$

A metric space is said compact if it is pre-compact and complete. The properties used in this paper are:

- The compact subsets of $\mathbb{R}^n$ are the closed and bounded subset of it.
- Finite or infinite product of compacts (i.e $(A^{\mathbb{Z}}, \bar{\delta})$) are compacts.
- Closed subset of compacts are compacts.
- Compacts of a metric space are closed and of bounded diameter (The distance between every two elements of a compact A is upper bounded by a constant $\texttt{diam}(A)$).
- The image of a compact by a continuous mapping is a compact.
- Metric spaces given in examples are compact.

A *dynamical system* is a couple (X, f) where X is a metric space and f is a homeomorphism of X i.e. a continuous bijection from X to X whose reciprocal f^{-1} is continuous.

Informally, we can see X as the state space of the system. The function f is the evolution law of the system, it gives the dynamics: given a starting state x_0, the states $f(x_0), f^2(x_0), ...$ are the successors of x, $f^n(x_0)$ is the state at the "moment" n. The function f^{-1} permits to go back in the past.

5.1.3 Shift spaces

Proposition-definition 20. *If $(\mathcal{C}, \delta)$ is a metric compact space then $((\mathcal{C}^{\mathbb{Z}}, \bar{\delta}), \sigma)$ is a dynamical system called full shift on $\mathcal{C}$.*

[1] The values $\log_2(\mathcal{N}_\varepsilon(X, \delta))$ and $\log_2(\mathcal{S}_\varepsilon(X, \delta))$ are known as ε-entropy and ε-capacity of X [KT59].

Proof. The reciprocal of σ is given by the right shift : $\sigma^{-1}(y) = x$ where $\forall i \in \mathbb{Z}, x_i = y_{i-1}$. Both shifts are continuous because they map converging sequences to converging sequences: $x^n \to_{n\to\infty} x$ iff $\forall i \in \mathbb{Z}, x_i^n \to_{n\to\infty} x_i$ iff $\sigma(x^n) \to_{n\to\infty} \sigma(x)$. □

A subspace X of $\mathcal{C}^{\mathbb{Z}}$ is called a sub-shift of $\mathcal{C}^{\mathbb{Z}}$ whenever it is closed and shift invariant: $\sigma(X) = X$. Clearly sub-shifts are also dynamical systems.

Given a shift space X, the set of factors of length n of bi-infinite words of X is denoted by $X_n = \{x_{[i+1..i+n]} \mid x \in X,\ i \in \mathbb{Z}\}$.

In the broad field of research of symbolic dynamics [BS02, LM95], the shift spaces considered are on finite alphabet ($|\mathcal{C}|$ is finite). Shift spaces on more general compact spaces have also been studied in [LW00].

5.1.4 ε-entropies and topological entropy

The topological entropy permits to measure the complexity of a system. Intuitively a system is complex when it is sensitive to initial conditions. There are several equivalent ways to define topological entropy, here we give a definition due to Bowen [Bow71].

Let $((X, d), f)$ be a dynamical system. For all $n \in \mathbb{N}^*$ we define the distance between the n first iterations of f on $x, y \in X$ by:

$$d_n(x, y) = \max_{0\leq k\leq n-1}(d(f^k(x), f^k(y))).$$

The idea is that two points x and y are ε far apart for d_n if when iterating f at most n times, we can distinguish them with a precision ε. An ε-net for d_n is thus an approximation of the system during n iteration and with precision ε. The N-ε entropy $h_\varepsilon^N(X)$ measures the growth rate of these sets w.r.t n:

$$h_\varepsilon^N(X) = \limsup_{n\to\infty} \frac{1}{n} \log_2(\mathcal{N}_\varepsilon(X, d_n)).$$

Similarly the S-ε-entropy is :

$$h_\varepsilon^S(X) = \limsup_{n\to\infty} \frac{1}{n} \log_2(\mathcal{S}_\varepsilon(X, d_n)).$$

The topological entropy $h_{\text{top}}(X)$ is defined by :

$$h_{\text{top}}(X) = \lim_{\varepsilon\to 0} h_\varepsilon^N(X)$$

which is also, by virtue of Lemma 24, equal to $\lim_{\varepsilon\to 0} h_\varepsilon^S(X)$.

5.2 Classical symbolic dynamics

In this section Σ is a finite alphabet which is the simplest case of compact alphabet. The theory of finite alphabet shift spaces is the core of symbolic dynamics. A broad part of this theory can be explained in an elementary manner with characterization of the principal objects (shift space, entropy, conjugacy ...) based on finite factors and without referring to topology (see [LM95]). We will lift to the timed case several results of this section.

5.2.1 Characterization with finite factors

Definition 1. *Given a family $O = (O_n)_{n\in\mathbb{N}^*}$ where for each $n \in \mathbb{N}$, $O_n \subseteq \mathcal{C}^n$, we denote by X_O the set of bi-infinite words not having factors in O : $X_O = \{x \in \mathcal{C}^{\mathbb{Z}} \mid \forall i \in \mathbb{Z}, \forall n \in \mathbb{N}, x_{[i..i+n]} \notin O_n\}$.*

We have also the dual definition

Definition 2. *Given a family $F = (F_n)_{n\in\mathbb{N}^*}$ where for each $n \in \mathbb{N}$, $F_n \subseteq \mathcal{C}^n$, we denote by $\mathcal{B}(F)$ the set of bi-infinite words whose authorized factors are those of F : $\mathcal{B}(F) = \{x \in \mathcal{C}^{\mathbb{Z}} \mid \forall i \in \mathbb{Z}, \forall n \in \mathbb{N}, x_{[i..i+n]} \in F_n\}$*

The following theorem permits to define shift spaces with factors.

Theorem 21. *A set is a shift space iff it can be defined as a X_O iff it can be defined as a $\mathcal{B}(F)$.*

The entropy can also be characterized with factors as follows:

Proposition 20. *The topological entropy of a sub-shift $X \subseteq \mathcal{C}^{\mathbb{Z}}$ for a finite $\mathcal{C}$ is:*

$$\lim_{n\to+\infty} \frac{1}{n} \log_2 |X_n|$$

5.2.2 Edge and sofic shifts

Let $G = (Q, \Delta)$ be a finite graph with possibly multiple edges between two nodes, Σ be a finite alphabet and $\texttt{Lab} : \Delta \to \Sigma$ a labelling function on edges. The couple $(G, \texttt{Lab})$ is called a labeled graph.

A finite (resp bi-infinite) path of G is a finite (resp bi-infinite) sequence of consecutive edges δ_i such that for all $i \in \{1, \ldots, n-1\}$ (resp $i \in \mathbb{Z}$) ${\delta_i}^+ = {\delta_{i+1}}^-$ (where we recall the notation that a transition δ starts from a vertex δ^- and ends in the vertex δ^+).

The set $\texttt{PATH}(G)$ of bi-infinite paths of a labeled graph G is a sub-shift of $(\Delta^{\mathbb{Z}}, \sigma)$ called *edge shift* of G. The *sofic shift* of the *labeled graph* $(G, \texttt{Lab})$ is the set of bi-infinite words that label bi-infinite paths of G:

$$\texttt{Lab}(\texttt{PATH}(G)) = \{(\texttt{Lab}(\delta_i))_{i\in\mathbb{Z}} \mid \forall i \in \mathbb{Z}, {\delta_i}^+ = {\delta_{i+1}}^-\},$$

it is a sub-shift of Σ^Z.

5.2.3 The language point of view.

In the next chapter we will recall several notions of constrained channel coding based on languages (of finite words) and their entropies. There is a direct correspondence between shift spaces and languages we explain now.

A language L is *factorial* whenever for every word $w \in L$ every factor of it is in the language. A language L is *extensible* whenever for every word $w \in L$ there exist letters $a, b \in \Sigma$ such that $wa \in L$ and $bw \in L$.

These two conditions are usual in the context of coding and can be justified in practice as follows. If we can encode (decode) some long word w (e.g. a movie file), then we want also to encode (decode) its contiguous fragments (e.g. a short scene in the middle of the movie). On the other hand, some extension of w should correspond to a longer movie.

A labelled graph can be seen as a finite automaton whose states are all initial and final. This property ensures factoriality. Every bi-infinite path of the automaton visits only vertices that have both ongoing and outgoing edges. Vertices that do not satisfy this condition can be pruned out without loss of generality. The resulting automaton has only states with ongoing and outgoing edges. This ensures extensibility of the language. Thus every regular, factorial and extensible language is the language of allowed block of a sofic shift. We call such a language *sofic*. Entropy can be defined directly for sofic languages L:

$$h(L) = \lim_{n \to +\infty} \frac{1}{n} \log_2 |L_n|. \tag{5.1}$$

For a sofic language L recognized by a given automaton, its entropy $h(L)$ can be effectively computed using linear algebra. In particular if $L = \Sigma^*$ for a k-letter alphabet Σ then $h(L) = \log_2 k$. Finally, the intuitive meaning of the entropy is the amount of information (in bits per symbol) in typical words of the language.

5.3 Compact alphabet shift space

5.3.1 Factor based characterization of shift spaces.

We already defined shift spaces as shift invariant and closed subsets of $\mathcal{C}^{\mathbb{Z}}$ for some metric compact alphabet $\mathcal{C}$. Here we generalize from finite to compact alphabet shift spaces the factor based characterization of such shift spaces (see [LM95] and Theorem 21 recalled above).

Definition 3. *Given a family $O = (O_n)_{n \in \mathbb{N}^*}$ where all O_n are open sets of $\mathcal{C}^n$, we denote by X_O the set of bi-infinite words whose forbidden factors are those of O : $X_O = \{x \in \mathcal{C}^{\mathbb{Z}} \mid \forall i \in \mathbb{Z}, \forall n \in \mathbb{N}, x_{[i..i+n]} \notin O_n\}$.*

We have also the dual definition

Definition 4. *Given a family $F = (F_n)_{n \in \mathbb{N}^*}$ where all F_n are closed sets of $\mathcal{C}^n$, we denote by $\mathcal{B}(F)$ the set of bi-infinite words whose authorized factors are those of F : $\mathcal{B}(F) = \{x \in \mathcal{C}^{\mathbb{Z}} \mid \forall i \in \mathbb{Z}, \forall n \in \mathbb{N}, x_{[i..i+n]} \in F_n\}$*

Theorem 22. *A subset of $\mathcal{C}^{\mathbb{Z}}$ is a shift space iff it can be defined as a X_O iff it can be defined as a $\mathcal{B}(F)$.*

Proof. **Poof that X is a shift space $\Leftrightarrow$ it can be defined as a $\mathcal{B}(F)$**

$\Rightarrow$) Let X be a shift space. Let $F = (X_n)_{n \in \mathbb{N}^*}$. By definition $X \subseteq \mathcal{B}(F)$. To show the converse inclusion, we take an $x \in \mathcal{B}(F)$ and prove that it belongs to X. As $x_{[-n..n]} \in X_{2n+1}$,

there exists a bi-infinite word $x^n \in X$ such that $x^n_{[-n..n]} = x_{[-n..n]}$. The sequence $(x^n)_{n\in\mathbb{N}}$ converges to x. As this sequence takes its values in the closed set X, its limit x is also in X. It remains to prove that for all $n \in \mathbb{N}^*$, X_n is closed. It suffices to show that for every convergent sequence $(w^m)_{m\in\mathbb{N}}$ of X_n, its limit w belongs to X_n. For each n, there exists an $x^m \in X$ such that $x^m_{[0..n-1]} = w^m$. The sequence x^m of the compact $C^{\mathbb{Z}}$ admits a subsequence which converges toward an $x \in X$. We have $x_{[0..n-1]} = \lim_n x^m_{[0..n-1]} = \lim_n w^m = w$ and thus w belongs to X_n as a factor of $x \in X$. □

⇐) Let $X = \mathcal{B}(F)$. X is shift invariant. Let us show that X is closed. It suffices to show that for every convergent sequence $(x^m)_{m\in\mathbb{N}}$ of X, its limit x belongs to X. For all $n \in \mathbb{N}^*$, $m \in \mathbb{N}$ and $i \in \mathbb{Z}$, $x^m_{[i..i+n-1]} \in F^n$. As F^n is closed, $x_{[i..i+n-1]} = \lim x^m_{[i..i+n-1]}$ belongs to F^n. All factors of x belongs to F thus $x \in \mathcal{B}(F) = X$ which is thus closed. □

Poof that X is a shift space ⇔ it can be defined as a X_O

⇒) We take $F = (X_n)_{n\in\mathbb{N}^*}$ as above. We define $O = (\mathcal{C}^n \setminus X_n)_{n\in\mathbb{N}^*}$. For each n, O_n is relatively open in $\mathcal{C}^n$ since X_n is closed. By definition of O, $\mathcal{B}(F) = X_0$ which is equal to X. □

⇐) Let $X = X_O$. X is shift invariant. Let us show that X is closed. It suffices to show that for every convergent sequence $(x^m)_{m\in\mathbb{N}}$ of X, its limit x belongs to X. For all $n \in \mathbb{N}$ and $i \in \mathbb{Z}$, we have $x_{[i..i+n-1]} = \lim x^m_{[i..i+n-1]} \notin O^n$ since O^n is open. Thus $x \in X_0$. □

Example 4. *Let $\mathcal{C} = [0,1] \times \{a\}$ and the set of forbidden factors be given by $O_2 = \{(a,t)(a,t') \mid t + t' > 1\}$. The shift space X_O is the set $\{(a,t_i)_{i\in\mathbb{Z}} \mid t_i + t_{i+1} \leq 1\}$.*

5.3.2 An infinite topological entropy

Topological entropy is very useful to compare dynamical systems. Unfortunately it is infinite in our case of interest, i.e. for shift spaces on infinite alphabet. Such fact was already known and given as a motivating remark in [LW00]. It is a straightforward consequence of the two following lemmas:

Lemma 25. *If $\mathcal{C}$ has infinite cardinality then $\mathcal{S}_\varepsilon(\mathcal{C},\delta) \to_{\varepsilon\to 0} +\infty$.*

Lemma 26. *Let $(\mathcal{C},\delta)$ be a compact metric space. Then*

$$h^S_\varepsilon((\mathcal{C}^{\mathbb{Z}},\bar{\delta}),\sigma) \geq \log_2(\mathcal{S}_\varepsilon(\mathcal{C},\delta)).$$

The fact that topological entropy for the shift spaces we are interested in is infinite can seem problematic. In the two next sections (sections 5.3.3 and 5.3.4) we will see two ways to circumvent this problem.

5.3.3 Entropy of a mesurable shift

The alphabets of interest Σ, $[0,M]$, $[0,M]\times\Sigma$ have natural measures. The counting measure for Σ, the Lebesgue measure (i.e. 1-dimensional-volume) for $[0,M]$ and the product measure of the two for $\mathbb{A} = [0,M] \times \Sigma$.

We suppose that the metric compact set $\mathcal{C}$ is also endowed with a "natural" measure $\mu_{\mathcal{C}}$. The set $\mathcal{C}^n$ has the product measure $\mu_{\mathcal{C}}^n$. For example the measure on Σ^n is the counting measure, that on $[0,M]^n$ is the n-dimensional Lebesgue measure (volume) and the measure on $([0,M]\times\Sigma)^n \cong [0,M]^n \times \Sigma^n$ also called volume is the product of the two preceding measures. As already observed in other chapters of this thesis, a subset $\mathbb{L}_n$ of $([0,M]\times\Sigma)^n$ can be seen as a formal sum of subsets $\mathbb{L}_w \subseteq [0,M]^n$ indexed by words $w \in \Sigma^n$. The volume of $\mathbb{L}_n$ is just the sum of the volumes of $\mathbb{L}_w$:

$$\texttt{Vol}(\mathbb{L}_n) = \texttt{Vol}\left(\cup_{w\in\Sigma^n}\{w\}\times\mathbb{L}_w\right) = \sum_{w\in\Sigma^n}\texttt{Vol}(\mathbb{L}_w).$$

We define the entropy of a sub-shift X of $\mathcal{C}^{\mathbb{Z}}$ as

$$\mathcal{H}(X) = \lim_{n\to+\infty}\frac{1}{n}\log\mu_{\mathcal{C}}^n(X_n). \tag{5.2}$$

This limit is well defined. Indeed, the set inclusion $X_{n+m} \subseteq X_m X_n$ for $m,n \geq 0$ implies the inequality $\mu_{\mathcal{C}}^{n+m}(X_{n+m}) \leq \mu_{\mathcal{C}}^m(X_m)\mu_{\mathcal{C}}^n(X_n)$. Thus the sequence $(\log\mu_{\mathcal{C}}^n(X_n))_{n\in\mathbb{N}}$ is sub-additive $\log\mu_{\mathcal{C}}^{n+m}(X_{n+m}) \leq \log\mu_{\mathcal{C}}^m(X_m) + \log\mu_{\mathcal{C}}^n(X_n)$ and we use (as in the proof of Proposition 2.3.3) Fekete's Lemma on sub-additive sequences [Fek23].

For example for finite alphabet shift spaces this definition coincides with the classical characterization of entropy (Proposition 20). The entropy of a sub-shift of $([0,M]\times\Sigma)^{\mathbb{Z}}$ is the (volumetric) entropy introduced in [AD09a] and recalled in Chapter 2.

5.3.4 Keeping the ε-entropy

Another way of circumventing the problem of the infinite topological entropy is to consider an asymptotic expansion of the ε entropy instead of its limit when ε tends to 0.

One of our main theorems (Theorem 25) states that for timed shift spaces considered, this asymptotic expansion is of the form $h_\varepsilon(X) = \log_2(\frac{1}{\varepsilon}) + \mathcal{H}(X) + o(1)$ where $\mathcal{H}(X)$ is the entropy defined above (5.2).

One can interpret as in [AD09b] $\mathcal{H}(X)$ as the average information for each event and $\log_2(\frac{1}{\varepsilon})$ as the information necessary to represents with precision ε the time between two events.

ε-entropies defined using factors

The distance $\bar{d}_n$ used in the definition of ε-entropies is quite uneasy to deal with. We give here definitions of ε-entropies based on distances d_n defined using finite factors of $\mathcal{C}^n$. For this distance ε-balls are just hypercubes of side ε.

Proposition 21. *Let X be a metric compact alphabet shift space. The N-ε-entropy (resp S-ε-entropy) defined on bi-infinite words is equal to the following ε-entropy of X defined using finite factors:*

$$h_\varepsilon^N(X) = \limsup_{n\to\infty}\frac{1}{n}\log_2(\mathcal{N}_\varepsilon(X_n,\delta^n));$$

$$h_\varepsilon^S(X) = \limsup_{n\to\infty} \frac{1}{n} \log_2(\mathcal{S}_\varepsilon(X_n, \delta^n)).$$

This proposition is a consequence of the four following lemmas.

Lemma 27. *For all $\varepsilon > 0$: $\mathcal{N}_\varepsilon(X_n, \delta^n) \leq \mathcal{N}_\varepsilon(X, \bar{\delta}_n)$.*

Proof. Let R be an ε-net of X for $\bar{\delta}_n$. We define $R' = \{x_{[0..n-1]} |\ x \in R\}$. We will show that R' is an ε-net of X_n. For all $y \in X_n$, there exists $y' \in X$ such that $y'_{[0..n-1]} = y$. There exists $x \in R$ such that $\bar{\delta}_n(x, y') \leq \varepsilon$. In particular, for all $i \in \{0, .., n-1\}, \bar{\delta}(\sigma^i(x), \sigma^i(y)) \leq \varepsilon$ thus $\max_{i\in\{0,..,n-1\}} \delta(x_i, y_i) < \varepsilon$ i.e $\delta^n(x, y) \leq \varepsilon$. □

Lemma 28. *For all $\varepsilon > 0$, there exists l such that $\mathcal{N}_\varepsilon(X_{n+2l}, \delta^{n+2l}) \geq \mathcal{N}_\varepsilon(X, \bar{\delta}_n)$.*

Proof. Let $l \in \mathbb{N}$ such that $\max_{|i|>l} \frac{\mathtt{diam}(\mathcal{C})}{2^{|i|}} \leq \varepsilon$. Let R_{n+2l} be an ε-net of X_{n+2l}. For all $x \in R_{n+2l}$, we choose $\hat{x} \in X$ such that $\hat{x}_{[-l..n+l-1]} = x$ and we define $\hat{R} = \{\hat{x} \mid x \in R\}$. We will show that $\hat{R}$ is an ε-net of X for $\bar{\delta}_n$. Let $y \in X$, there exists $x \in R_{n+2l}$ such that $\delta^{n+2l}(y_{[-l..n+l-1]}, x) \leq \varepsilon$. Therefore, for all $k \in \{0..n-1\}$: $\max_{i\in\{-l...l\}} \frac{\delta(x_{i+k}, y_{i+k})}{2^{|i|}} \leq \varepsilon$ and thus $\sup_{i\in\mathbb{Z}} \frac{\delta(x_{i+k}, y_{i+k})}{2^{|i|}} \leq \varepsilon$ i.e $\bar{\delta}_n(x, y) \leq \varepsilon$ □

Lemma 29. *For all $\varepsilon > 0$: $\mathcal{S}_\varepsilon(X_n, \delta^n) \leq \mathcal{S}_\varepsilon(X, \bar{\delta}_n)$.*

Proof. Let S be an ε-net of X_n for δ^n. For all $x \in S$, we choose $\hat{x} \in X$ such that $\hat{x}_{[0..n-1]} = x$ and we define $\hat{S} = \{\hat{x} \mid x \in S\}$. We have, for all $\hat{x}, \hat{y} \in \hat{S}$ $\bar{\delta}_n(\hat{x}, \hat{y}) = \max_{k\in\{0...n-1\}} \sup_{i\in\mathbb{Z}} \frac{\delta(\hat{x}_{i+k}, \hat{y}_{i+k})}{2^{|i|}} \geq \max_{k\in\{0..n-1\}} \delta(x_k, y_k) = \delta^n(x, y) > \varepsilon$. Thus $\hat{S}$ is ε-separated. □

Lemma 30. *For all $\varepsilon > 0$, there exists l such that $\mathcal{S}_\varepsilon(X_{n+2l}, \delta^{n+2l}) \geq \mathcal{S}_\varepsilon(X, \bar{\delta}_n)$.*

Proof. Let $l \in \mathbb{N}$ such that $\max_{|i|>l} \frac{\mathtt{diam}(\mathcal{C})}{2^{|i|}} \leq \varepsilon$. Let S a ε-net of X for $\bar{\delta}_n$. We define $S' = \{x_{[-l..l+n-1]} \mid x \in S\}$. We have, for all $x, y \in S$, $\bar{\delta}_n(x, y) = \max_{k\in\{0...n-1\}} \sup_{i\in\mathbb{Z}} \frac{\delta(x_{i+k}, y_{i+k})}{2^{|i|}} > \varepsilon$. The terms of indices less than $-l$ and greater than $n + l - 1$ are not taken into account as by definition of l they cannot be greater than ε. Therefore $\delta^{n+2l}(x_{[-l..l+n-1]}, y_{[-l..l+n-1]}) \geq \bar{\delta}_n(x, y) > \varepsilon$ and S' is ε-separated. □

5.4 Timed edge shift and timed sofic shift

We define in this section timed sofic shifts which are a way to see regular timed languages as compact alphabet shift spaces. This permits to bring tools from symbolic dynamics as entropy, metric mean dimension, codings to timed automata theory. In the opposite direction, this permits to bring the continuous time of timed automata theory to the field of symbolic dynamics and coding.

5.4.1 Definitions

Timed edge shift

We have defined timed region graphs (TRG) and their closed versions (CTRG) in Chapter 2. CTRGs are the timed analogues of finite graphs. A CTRG yields two kinds of compact alphabet shift spaces, one is the set of bi-infinite runs (whose definition are recalled just below) the other called timed edge shift is the set of bi-infinite words of timed transitions that can be recognized by the CTRG.

A bi-infinite *run* of a (closed) timed region graph $\mathcal{G}$ is a bi-infinite word $(s_i, \alpha_i)_{i\in\mathbb{Z}} \in (\mathbb{S} \times \mathbb{A})^{\mathbb{Z}}$ such that $s_{i+1} = s_i \triangleright \alpha_i \neq \bot$ for all $i \in \mathbb{Z}$ (notation $\triangleright$ as well as finite runs has been defined in chapter 2).

Lemma 31. *The set* $\{(s, \alpha, s') \mid s' = s \triangleright \alpha\}$ *is a closed subset of* $\mathbb{S} \times \mathbb{A} \times \mathbb{S}$.

Proof. It suffices to remark that for every transition δ the set of tuples $(\vec{x}, t, \vec{x}')$ such that $(\delta^-, \vec{x}) \triangleright (t, \delta) = (\delta^+, \vec{x}')$ is a polytope. Indeed it is a bounded set defined by equations involving equality and non-strict inequality only (they have the following form $x' = x + t$, $x' = 0$, $A \leq x \leq B$, $x \leq y$, $A \leq x + t \leq B$). □

Proposition 22. *The set of bi-infinite runs of* $\mathcal{G}$, $\mathtt{Run}_{\mathcal{G}} = \{(s_i, \alpha_i)_{i\in\mathbb{Z}} \mid s_{i+1} = s_i \triangleright \alpha_i\}$ *is a sub-shift of* $(\mathbb{S} \times \mathbb{A})^{\mathbb{Z}}$.

Proof. This set is shift invariant. We show that it is closed. The set $\{(s, \alpha, s') \mid s' = s \triangleright \alpha\}$ is a closed subset of $\mathbb{S} \times \mathbb{A} \times \mathbb{S}$. For a fixed $j \in \mathbb{Z}$ the projection $(s_i, \alpha_i)_{i\in\mathbb{Z}} \mapsto (s_j, \alpha_j, s_{j+1})$ is continuous and thus the set of runs $(s_i, \alpha_i)_{i\in\mathbb{Z}}$ such that $s_{j+1} = s_j \triangleright \alpha_j$ is closed. The set $\mathtt{Run}_{\mathcal{G}}$ is the intersection for $j \in \mathbb{Z}$ of the closed sets described just above, it is thus closed. □

Proposition-definition 23. *The following set is a sub-shift of* $\mathbb{A}^{\mathbb{Z}}$ *called the* timed edge shift *of* $\mathcal{G}$ *and denoted by* $X_{\mathcal{G}}$*:*

$$X_{\mathcal{G}} = \{(\alpha_i)_{i\in\mathbb{Z}} \mid \exists (s_i)_{i\in\mathbb{Z}} \in \mathbb{S}^{\mathbb{Z}},\ \forall i \in \mathbb{Z}, s_{i+1} = s_i \triangleright \alpha_i\}.$$

Proof. This set is obtained by projecting $\mathtt{Run}_{\mathcal{G}}$ on the timed transition components. By continuity, the projected set is compact. □

Every bi-infinite run visits only locations that have both ongoing and outgoing edges. Location that do not satisfy this condition can be pruned out without loss of generality. From now on we consider without loss of generality only such pruned labelled graphs.

A timed edge or sofic shift is *fleshy* if so is its underlying TRG (see Chapter 2 for definition of fleshiness).

Timed sofic shift

Similarly to the finite case, when adding to a TRG $\mathcal{G}$ a labelling function $\mathtt{Lab} : \Delta \to \Sigma$ from the set of transition Δ to a finite alphabet of event Σ we obtain a *labelled timed region graph* LTRG $\mathcal{A} = (\mathcal{G}, \mathtt{Lab})$.

Abusing the notation we will extend the labelling function to timed letters and runs as follows: $\mathtt{Lab}(\alpha) = (t, \mathtt{Lab}(\delta))$ when $\alpha = (t, \delta)$ and $\mathtt{Lab}[(s_i, \alpha_i)_{i\in\mathbb{Z}}] = (s_i, \mathtt{Lab}(\alpha_i))_{i\in\mathbb{Z}}$.

As in classical symbolic dynamics a LTRG is called right resolving if every two different transitions have pairwise incompatible guards (this is the usual definition of determinism without the condition of the uniqueness of an initial state). Finally a labelled closed timed region graph (LCTRG) is called right resolving if it is obtained from a right resolving LTRG by taking the closure of guards and regions.When dealing with LCTRG, there can be a non determinism on border of guards, however it remains of null volume. For instance the LCTRG on figure 5.1 is right resolving with a null volume non determinism for $x = 1$.

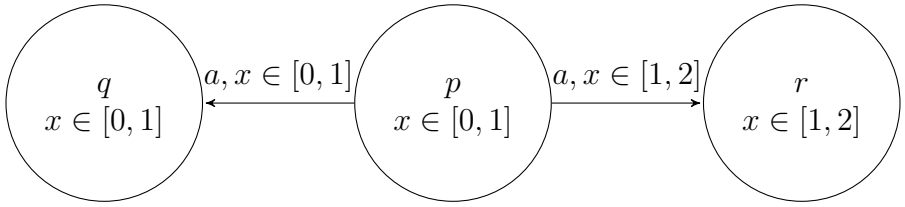

Figure 5.1: A right resolving LCTRG

Proposition-definition 24. *Let $\mathcal{A} = (\mathcal{G}, \mathtt{Lab})$ be a labeled closed timed region graph then the following set is a sub-shift of $(\mathbb{S} \times \mathtt{Lab}(\mathbb{A}))^{\mathbb{Z}}$ called the timed sofic shifts of $\mathcal{A}$ and denoted by $X_{\mathcal{A}}$:*

$$X_{\mathcal{A}} = \mathtt{Lab}(X_{\mathcal{G}}) = \{\mathtt{Lab}[(\alpha_i)_{i\in\mathbb{Z}}] \mid (\alpha_i)_{i\in\mathbb{Z}} \in (\alpha_i)_{i\in\mathbb{Z}} \in X_{\mathcal{G}}\}$$

Indeed since labelling is a continuous mapping, $X_{\mathcal{A}}$ is a shift space as $X_{\mathcal{G}}$.

We are mainly interested in *right-resolving* timed sofic shift: the shift spaces associated to right resolving LCTRGs.

The LCTRGs depicted in figure 5.2 are right-resolving, they recognize the timed sofic shift spaces of Examples 5, 6, 7 and 8.

5.4.2 The timed language point of view

Here we relate shift spaces of bi-infinite timed words (the focus of the chapter) to languages of finite timed words (used in the rest of the thesis). We also prove that edge and sofic shifts have the same entropy in case of determinism.

A right-resolving labelled (closed) timed region graph can be seen as a (closed) region-split BDTA (see [AD09a] and section 2.3 of the present thesis) with all states of entry regions being initial and finals. This latter property ensures factoriality of the language as defined and motivated in the discrete case (section 5.2.3). Moreover, the language is extensible (as the timed region graph is assumed to be pruned). Such timed language recognized by right-resolving LCTRG are called *sofic*. They will be used in the next chapter.

Recall from Chapter 2 that the language of a timed region graph is $\mathbb{L}(\mathcal{G}) = \{(\vec{t}, \pi) \mid \vec{t} \in P_\pi\}$. This language is exactly the set of allowed factors of $X_{\mathcal{G}}$. It is extensible and factorial.

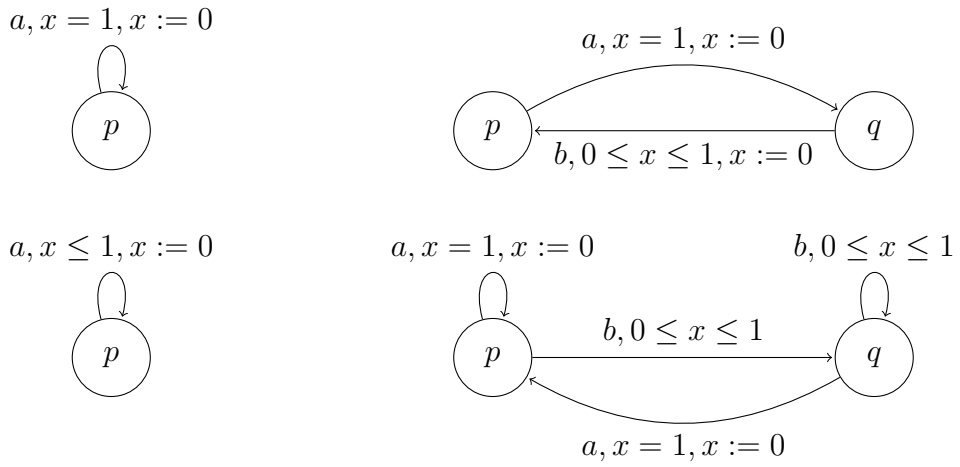

Figure 5.2: LCTRGs for Examples 5, 6 (first row) and for Examples 7 and 8 (second row)

The following proposition states that entropy of a timed sofic language is equal to that of the underlying TRG $\mathcal{G}$.

Proposition 23. *Given a right-resolving LCTRG $\mathcal{A} = (\bar{\mathcal{G}}, \mathtt{Lab})$, its entropy is equal to that of its underlying TRG $\mathcal{G}$:*

$$\mathcal{H}(\mathbb{L}(\mathcal{A})) = \lim_{n\to\infty} \frac{1}{n} \log_2 \left(\sum_{w\in\Sigma^n} \mathtt{Vol}(P_w) \right) = \lim_{n\to\infty} \frac{1}{n} \log_2 \left(\sum_{\pi\in\Delta^n} \mathtt{Vol}(P_\pi) \right) = \mathcal{H}(\mathbb{L}(\mathcal{G})).$$

Proof. The inequality $\mathcal{H}(\mathcal{A}) \leq \mathcal{H}(\mathcal{G})$ is easy to prove since for each w we have $P_w = \cup_{\pi\in\mathtt{Lab}^{-1}(w)} P_\pi$ and then for every $n \in \mathbb{N}$:

$$\mathtt{Vol}(\mathbb{L}_n(\mathcal{A})) = \sum_{w\in\Sigma^n} \mathtt{Vol}(P_w) \leq \sum_{w\in\Sigma^n} \sum_{\pi\in\mathtt{Lab}^{-1}(w)} \mathtt{Vol}(P_\pi) = \mathtt{Vol}(\mathbb{L}_n(\mathcal{G})).$$

For the converse inequality, everything would be very simple if the above union were disjoint. This is not the case due to freedom on the initial state. Indeed, let us consider for instance the right resolving LCTRG depicted in Figure 5.1. The timed transition $(0.5, a)$ taken from $(p, 0.2)$ leads to $(q, 0.7)$ while taken from $(p, 0.8)$ leads to $(r, 1.3)$.

However the polytopes $P_{\pi_{(2)}}(s)$, $P_{\pi'_{(2)}}(s)$ associated with two distinct paths $\pi_{(2)} \neq \pi'_{(2)}$ and a same starting state s are disjoint. Then, if all clocks are reset during a path $\pi_{(1)}$, for every two distinct paths $\pi_{(2)}, \pi'_{(2)}$ we have $P_{\pi_{(1)}\pi_{(2)}} \cap P_{\pi_{(1)}\pi'_{(2)}} = \emptyset$. We divide in two groups the set of paths of a given length l (l is a parameter that we will tune later):

- the set $R(l)$ of paths which reset all its clocks;
- the set of other paths. For paths in this latter set, it holds that $\mathtt{Vol}(P_\pi) \leq \frac{1}{l!}$

The n^{th} volume associated with $\mathcal{G}$ is

$$\mathtt{Vol}(\mathbb{L}_n(\mathcal{G})) = \sum_{\pi\in\Delta^n} \mathtt{Vol}(P_\pi) = \sum_{\pi_{(1)}\in R(l)} \sum_{\pi_{(2)}\in\Delta^{n-l}} \mathtt{Vol}(P_{\pi_{(1)}\pi_{(2)}}) + \sum_{\pi_{(1)}\notin R(l)} \sum_{\pi_{(2)}\in\Delta^{n-l}} \mathtt{Vol}(P_{\pi_{(1)}\pi_{(2)}})).$$

We will denote by S_1 and S_2 the two sums above. S_2 is upper bounded by $\frac{|\Delta|^n}{l!}$. For each $w \in \Sigma^n$ we have

$$P_w = \bigcup_{\pi_{(1)}\in R(l)} \biguplus_{\substack{\pi_{(2)}\in\Delta^{n-l} \\ \pi_{(1)}\pi_{(2)}\in \mathtt{Lab}^{-1}(w)}} P_{\pi_{(1)}\pi_{(2)}}$$

and then

$$\mathtt{Vol}(P_w) \geq \max_{\pi_{(1)}\in R(l)} \sum_{\substack{\pi_{(2)}\in\Delta^{n-l} \\ \pi_{(1)}\pi_{(2)}\in \mathtt{Lab}^{-1}(w)}} \mathtt{Vol}(P_{\pi_{(1)}\pi_{(2)}})) \geq \frac{1}{|\Delta|^l} \sum_{\pi_{(1)}\in R(l)} \sum_{\substack{\pi_{(2)}\in\Delta^{n-l} \\ \pi_{(1)}\pi_{(2)}\in \mathtt{Lab}^{-1}(w)}} \mathtt{Vol}(P_{\pi_{(1)}\pi_{(2)}}).$$

We sum over all w and deduce that $\mathtt{Vol}(\mathbb{L}_n(\mathcal{A})) \geq S_1/|\Delta|^l$. Remark that $\mathcal{H}(X_\mathcal{A}) \geq \mathcal{H}(X_\mathcal{G}) - \log_2(|\Delta|)$ since

$$\mathtt{Vol}(\mathbb{L}_n(\mathcal{A})) = \sum_{w\in\Sigma^n} \mathtt{Vol}(P_w) \geq \max_{\pi\in\Delta^*} \mathtt{Vol}(P_\pi) \geq \frac{\mathtt{Vol}(\mathbb{L}_n(\mathcal{G}))}{|\Delta|^n}.$$

Hence $\mathcal{H}(\mathbb{L}(\mathcal{A})) = -\infty$ iff $\mathcal{H}(\mathbb{L}(\mathcal{G})) = -\infty$. We suppose now that $\mathcal{H}(\mathbb{L}(\mathcal{A})) > -\infty$. In particular $\mathtt{Vol}(\mathbb{L}_n(\mathcal{A}))$ behaves like an exponent in the following sense: for every $a < 2^{\mathcal{H}(\mathbb{L}(\mathcal{A}))} < b$ it holds that $b^n >> \mathtt{Vol}(\mathbb{L}_n(\mathcal{A})) >> a^n$.

Recap that $\mathtt{Vol}(\mathbb{L}_n(\mathcal{G})) = S_1 + S_2$, $S_2 \leq \frac{|\Delta|^n}{l!}$and $S_1 \leq \mathtt{Vol}(\mathbb{L}_n(\mathcal{A}))|\Delta|^l$, hence

$$\mathtt{Vol}(\mathbb{L}_n(\mathcal{G})) = S_1 + S_2 \leq \mathtt{Vol}(\mathbb{L}_n(\mathcal{A})) \left(|\Delta|^l + \frac{|\Delta|^n}{l!\mathtt{Vol}(\mathbb{L}_n(\mathcal{A}))} \right). \tag{5.3}$$

We choose l such that $l << n << \log_2(l!)$ e.g. l such that $n = l\log_2(\log_2(l))$. With such an l the quantity $\frac{1}{n}\log_2(|\Delta|^l + \frac{|\Delta|^n}{l!\mathtt{Vol}(\mathbb{L}_n(\mathcal{A}))})$ tends to 0 and then taking $\lim_{n\to\infty} \frac{1}{n}\log_2(.)$ in (5.3) yields $\mathcal{H}(\mathbb{L}(\mathcal{A})) \leq \mathcal{H}(\mathbb{L}(\mathcal{A}))$.

□

Thus, in case of determinism the computability of entropy for a timed sofic shift reduces to the computability of entropy of its underlying timed region graph. That is why, in the following, we restrict our attention w.l.o.g. to timed region graphs (without labelling) and their corresponding timed edge shifts.

5.4.3 Discretization

Several definitions of ε-entropy for general compact alphabet shift spaces were given in section 5.3.4. Here we work with *timed shift spaces*: the sub-shifts of $\mathbb{A}^{\mathbb{Z}}$. In this particular case we give a simpler definition of ε-entropy which turns out to be computable for timed sofic shift and asymptotically equal to the other ε-entropies under some hypotheses (i.e. thickness and fleshiness). This new definition of ε-entropy is based on discretization of the timed shift space we explore now.

We call *ε-discrete* the different objects involving delays and clocks multiple of ε (i.e vector of delays of $\mathbb{R}^n$, timed words, bi-infinite timed words, runs, etc.). The *ε-discretization* of a set B denoted by B_ε is the set of its ε-discrete element. For instance, for $\mathbb{A} = [0, M] \times \Delta$, $\mathbb{A}_\varepsilon = \{0, \varepsilon, \ldots, M\varepsilon\} \times \Delta$; for $X \subseteq \mathbb{A}^{\mathbb{Z}}$, $X_\varepsilon = X \cap \mathbb{A}_\varepsilon^{\mathbb{Z}}$; for $P \subseteq \mathbb{R}^n$, $P \subseteq \varepsilon\mathbb{Z}^n$.

Given an ε-discrete point $\vec{t} = (t_1, \ldots, t_n) \in \mathbb{R}^n$, its ε-North-East-neighborhood is the hypercube $\mathcal{B}_\varepsilon^{NE}(\vec{t}) = \{(u_1, \ldots, u_n) \mid u_i \in [t_i, t_i + \varepsilon],\ i = 1..n\}$. This notion extends to timed words as follows $\mathcal{B}_\varepsilon^{NE}(\vec{t}, w) = \mathcal{B}_\varepsilon^{NE}(\vec{t}) \times \{w\}$. as well as to sets of ε-discrete elements $\mathcal{B}_\varepsilon^{NE}(B_\varepsilon) = \cup_{b \in B_\varepsilon} \mathcal{B}_\varepsilon^{NE}(b)$.

Discretization of a shift space and its entropy

The following properties permit a reduction from timed shift spaces to discrete ones.

Proposition 24. *Given a subshift X of $\mathbb{A}^{\mathbb{Z}}$ then X_ε is a subshift of $\mathbb{A}_\varepsilon^{\mathbb{Z}}$.*

We define the ε-entropy of a shift X as the (topological) entropy of the shift X_ε:

$$h_\varepsilon(X) =_{\text{def}} \lim_{n \to \infty} \frac{1}{n} \log_2 |X_{\varepsilon,n}|.$$

The discretization of a timed edge shift $X_\mathcal{G}$ is the sofic shift of a finite labeled graph $\mathcal{G}_\varepsilon$ obtained from $\mathcal{G}$ by a discretization of its timed transitions and states as follows: $\mathcal{G}_\varepsilon = ((Q_\varepsilon, \Delta_\varepsilon), \mathtt{Lab}_\varepsilon)$ with $Q_\varepsilon = \mathbb{S} \cap (Q \times \{0, \varepsilon, \ldots, M\varepsilon\}^d)$, $\mathtt{Lab}_\varepsilon : \Delta_\varepsilon \to \mathbb{A}_\varepsilon$ and there is a transition $s \to s'$ in Δ_ε labeled by α iff $s \triangleright \alpha = s'$.

Proposition 25. *Let $\mathcal{G}$ be a CTRG, then $\mathcal{G}_\varepsilon$ is right resolving and*

$$X_{(\mathcal{G}_\varepsilon)} = (X_\mathcal{G})_\varepsilon.$$

Proof. Let Y be the sofic shift recognized by $\mathcal{G}_\varepsilon$. It is straightforward that $Y \subseteq X_\varepsilon$. To prove the converse inclusion we take $(\alpha_i)_{i \in \mathbb{Z}} \in (X_\mathcal{G})_\varepsilon$ and show that $(\alpha_i)_{i \in \mathbb{Z}} \in Y$. By definition of $(X_\mathcal{G})_\varepsilon$, each α_i is ε-discrete i.e. of the form $\alpha_i = (k_i\varepsilon, \delta_i)$ with $k_i \in \{0, \ldots, M/\varepsilon\}$ and $(\alpha_i)_{i \in \mathbb{Z}}$ is obtained by projecting states of an infinite run of $\mathcal{G}$, say $(s_i, \alpha_i)_{i \in \mathbb{Z}}$. We denote by x_i the value of the clock x at the index i of this run. Our objective is to transform the value of x_i for all clocks x and indexes i in such a way that the new values are multiple of ε and the guards are still satisfied. For every clock x, we denote by $\mathtt{fr}(x)$ the index of first reset of x (possibly equal to $-\infty$ if the clock is reset infinitely often in the past or $+\infty$ if the clock is never reset).

Since all the delays are multiple of ε then so is x_i for $i \geq \mathtt{fr}(x)$. Remark that for all $i < \mathtt{fr}(x)$ we have $x_i = x_{\mathtt{fr}(x)} - \sum_{i \leq l < \mathtt{fr}(x)} k_l \varepsilon$. As all $x_j \in \varepsilon\mathbb{N}$ for $j \in \mathbb{Z}$, it holds that $k_l = 0$ for every l lower than a position $\mathtt{fp}(x)$ where it is positive for the first time (here also $\mathtt{fp}(x)$ can take values $-\infty$, $+\infty$). We have thus the three possible cases for x_i:

- $x_i = x_{\mathtt{fp}(x)}$ if $i \leq \mathtt{fp}(x)$;
- $x_i = x_{\mathtt{fp}(x)} + \sum_{l=\mathtt{fp}(x)}^{i-1} k_l \varepsilon$ if $\mathtt{fp}(x) < i \leq \mathtt{fr}(x)$;
- x_i is multiple of ε if $i > \mathtt{fr}(x)$.

It remains to choose a new value for $x_{\mathtt{fp}(x)}$ that is multiple of ε. The guards on the path $(\delta_i)_{i \in Z}$ give inequalities of the form $A_i \leq x_i \leq B_i$. The lower bound for $\varepsilon^{-1} x_{\mathtt{fp}(x)}$ is

$$\sup\Big(\sup_{i \leq \mathtt{fp}(x)} \varepsilon^{-1} A_i, \sup_{\mathtt{fp}(x) < i \leq \mathtt{fr}(x)} \varepsilon^{-1} A_i - \sum_{l=\mathtt{fp}(x)}^{i-1} k_i \Big).$$

This lower bound is an integer since it is a supremum over a set of integers. A symmetric reasoning can be used for the upper bound. An arbitrary choice between the lower and upper bounds gives a new value for $\varepsilon^{-1} x_{\mathtt{fp}(x)}$ which is an integer. This choice does not affect the delays nor the values of the other clocks, it permits to have a new run satisfying the same constraints. One can repeat this operation until all the clocks are ε-discrete in all positions and then we are done. □

As a corollary the computation of the ε-entropy of a timed sofic shift reduces to the computation of the entropy of a (finite alphabet) sofic shift:

Corollary 1 (A symbolic dynamics version of [AD09b], Theorem 3)**.** *Let $\mathcal{G}$ be a timed region graph, its ε-entropy is the topological entropy of the sofic shift $\mathcal{G}_\varepsilon$:*

$$h_\varepsilon(X_\mathcal{G}) = h(X_{\mathcal{G}_\varepsilon})$$

In particular, $h_\varepsilon(X_\mathcal{G})$ can be computed as the logarithm of the spectral radius of the adjacency matrix of the graph $\mathcal{G}_\varepsilon$ (This matrix has order $O(|Q|/\varepsilon^d)$ where d is the number of clocks).

Discretizing the volumetric entropy

Theorem 25. *Let $\mathcal{G}$ be a fleshy thick timed region graph then its volumetric entropy can be approximated by its ε-entropy as follows:*

$$h_\varepsilon = \log_2(1/\varepsilon) + \mathcal{H} + o(1)$$

The proof uses geometrical arguments on polytopes associated to paths and requires several definitions we give now. Given a polytope P, we denote by NP its N-fold dilated copy, i.e. $\{N\vec{t} \mid \vec{t} \in P\}$ and by $E(P) = P \cap \mathbb{Z}^n$ the set of points with integer coordinates in P. We call a contiguous polytope *d-fat* if it is d contiguous and there exists an integer

point in the interior of dP (called an *internal point*). In other words, a polytope is d-fat if there exists a point $(u_1, \cdots, u_n) \in \mathbb{R}^n$ such that $\sum_{i=j}^{k} u_i \in [dA+1, dB-1]$ for all inequalities $\sum_{i=j}^{k} t_i \in [A, B]$ defining the polytopes and if $k - j + 1 \leq d$ for all such inequalities.

The $\geq$ direction of Theorem 25 is the following lemma:

Lemma 32. *For thick timed sofic shifts:*

$$h_\varepsilon \geq \mathcal{H} + \log_2(1/\varepsilon). \tag{5.4}$$

Proof. We will bound the volume $\texttt{Vol}(X_n)$ by the number of discrete points $|X_{\varepsilon,n}|$. For this, we will use a beautiful theorem on counting points in polytopes:

Theorem 26 (Ehrhart, see [BR07])**.** *For integer N and an integer polytope $P \subset \mathbb{R}^n$ (i.e. whose vertices have integer coordinates), the number of integer points $|E(NP)|$ is a polynomial in N with non negative coefficients of degree n and whose coefficient of the highest degree is the volume of P.*

We deduce directly from this theorem that for each path π of length n and $\varepsilon = \frac{1}{N}$ the following holds: $\texttt{Vol}(P_\pi)N^n \leq |E(NP_\pi)|$. Summing over all paths of length n and taking $\lim_{n\to\infty} \frac{1}{n} \log_2(.)$ in both sides of the inequality, we get $\mathcal{H} + \log_2 \frac{1}{\varepsilon} \leq h_\varepsilon$. □

Upper bounding h_ε by $\mathcal{H} + \log_2 \frac{1}{\varepsilon} + o(1)$ is more involved, and we first give a sketch of the proof. Details of the proof are given later.

Sketch of the proof. We fix several integer parameters: b, c, d, e (they have to be adjusted in order to obtain the required estimate). Let π be a path of a length n (It is fleshy by assumption). At every b transitions, we insert in π a forgetful cycle of length c (it exists by virtue of Lemma 14). Thus we obtain a slightly longer path π' (its length is $n' \approx n(1+c/b)$), whose polytope P'_π is e-fat.We have three inequalities:

1. The first one:
$$|P_{\pi,\varepsilon}| \leq |P_{\pi',\varepsilon}|$$
is proved by constructing an injection from the left-hand side ε-discrete set to the right-hand side one.

2. We choose ε' slightly smaller than ε (another parameter to adjust) and consider the polytope P^- obtained from $P_{\pi'}$ by pushing all its facets inside by the amount[2] $\delta = \varepsilon' e$. Using fatness of $P_{\pi'}$, it is possible to build an injection from its ε-discrete points to ε'-discrete points of P^- (the latter is a bit smaller but its discrete points are slightly denser).
$$|P_{\pi',\varepsilon}| \leq \left|P^-_{\varepsilon'}\right|.$$

[2]i.e. by replacing each constraint $\sum_{i=j}^{k} t_i \in [A, B]$ in the definition of $P_{\pi'}$ as a (closed) contiguous polytope by $\sum_{i=j}^{k} t_i \in [A+\delta, B-\delta]$ (see also [AD09b]).

3. Taking an ε'-cube at every ε'-discrete point of P^-, we get a set included in $P_{\pi'}$ (this requires e-contiguity of $P_{\pi'}$). Passing to volumes we conclude that

$$\varepsilon'^{n'} \left| P^-_{\varepsilon'} \right| \leq V_{\pi'}.$$

Combining the three inequalities we get:

$$|P_{\pi,\varepsilon}| \leq \left(\varepsilon'^{-n'} V_{\pi'} \right),$$

and with an appropriate choice of parameters, ε' and n' can be made very close to ε and n. Summing up over π and taking $\lim_{n\to\infty} \frac{1}{n} \log_2(.)$ in the previous inequality, we obtain the required result. □

Corollary 2. *For a timed region graph $\mathcal{G}$, $\mathcal{H}$ is computable as function of $\mathcal{G}$. Consequently, $\mathcal{H}$ is a computable real (i.e. one can compute its approximation with any wanted precision).*

Proof. Using Theorem 6 of Chapter 3 one can decide whether $\mathcal{G}$ is thick or not. If it is thin then $\mathcal{H} = -\infty$. Otherwise, the automaton is thick and it just remains to compute the discrete entropy $h(X_{\mathcal{G}_\varepsilon})$ (see Theorem 1) for the wanted precision. □

Details of the proof of Theorem 25

We begin by some auxiliary results used in the core of the proof.

Lemma 33. *If a contiguous polytopes contains a $1/N$ discrete and internal point then it is NM-fat.*

Proof. Let $(t_1, \ldots, t_n)$ be a $1/N$ discrete and internal point of the polytope. This point is also a $\frac{1}{NM}$ discrete and internal point of the polytope. Equations of the polytopes are of the form $A \leq \sum_{i=j}^{k} t_i \leq B$ with $B \leq M$. As for every $i \in \{1, \ldots, n\}$, $t_i > 0$ and t_i is multiple of $\frac{1}{N}$ then $t_i \geq \frac{1}{N}$ and $\frac{k-j+1}{e} \leq \sum_{i=j}^{k} t_i \leq M$. We deduce the result: lengths $k - j + 1$ of contiguous sums are bounded by NM. □

The following two discretization lemmas are from [AMP98].

Lemma 34. *If $\vec{x}, \vec{x}'$ are ε-discrete and $P_{\overline{\pi}}(\vec{x}, \vec{x}') \neq \emptyset$ then $P_{\overline{\pi},\varepsilon}(\vec{x}, \vec{x}') \neq \emptyset$.*

Lemma 35. *Every contiguous polytope P of dimension n has an $\frac{1}{m}$-discrete internal point for all $m > n$.*

The constant c occurring in the following definition is that defined in Proposition 14.

Lemma 36 (fattening the polytopes)**.** *Let $b \in \mathbb{N}$, for every $n \in \mathbb{N}$ there exists an injection ϕ from path of length n to path of length $n' = n + \lfloor n/b \rfloor c = n(1 + O(b))$ and such that polytopes of these paths are e-fat with $e =_{def} M(c+1)(b+1) = O(b)$ and satisfy:*

$$|P_{\pi,\varepsilon}| \leq \left| P_{\phi(\pi),\varepsilon} \right|.$$

Proof. Let π be a path of length $n = mb+r$ with $0 \leq r \leq b-1$. We will insert in π at every b letters a forgetful cycle f_i such that the polytope associated to the created word is e-fat with $e = M(b+1)(c+1)$ (and thus contain more discrete points).

Let $\pi = \pi_{(1)}\pi_{(2)}\dots\pi_{(m)}\pi_{(m+1)}$ where $\pi_{(1)},\dots,\pi_{(m)}$ are words of length b (and thus $|\pi_{(m+1)}| = r$). For all $i \in \{1,\dots,m\}$, there exists a forgetful cycle f_i of length c on the region $\mathbf{r}_i$ between $\pi_{(i)}$ and $\pi_{(i+1)}$.

We define $\pi' = \phi(\pi)$ by $\phi(\pi) = \pi_{(1)}f_1\pi_{(2)}\dots\pi_{(m-1)}f_{m-1}\pi_{(m)}f_m\pi_{(m+1)}$. Function ϕ is an injection from E^n to $E^{n'}$ with $n' = n + mc \sim n(1 + c/b)$. This injection can be extended to ε-discrete words because with Lemma 34, for each couple of ε-discrete states of $\mathbf{r}_i$ we can choose delays labelling f_i to join each other. Therefore we have the inequality $|P_{\pi,\varepsilon}| \leq |P_{\pi',\varepsilon}|$.

By Lemma 35, for each i one can find a $\frac{1}{b+1}$-discrete and internal run on $\pi_{(i)}$ starting from a state s_i and ending in a state s'_i. One can also find a $1/[(b+1)(c+1)]$-discrete and internal run on f_i from s'_i to s_{i+1}. We have described a $1/[(b+1)(c+1)]$-discrete and internal run on π'. The polytope is thus e-fat with $e = M(b+1)(c+1)$ by virtue of Lemma 33. □

For every e-contiguous polytope, and a discretization step $\varepsilon \leq 1/2e$, we denote by $P^{-e\varepsilon}$ the polytope defined from P by replacing all inequalities $\sum_{i=j}^{k} t_i \in [A,B]$ involved in the definition of P by $\sum_{i=j}^{k} t_i \in [A+e\epsilon, B-e\varepsilon]$.

Lemma 37. *If P is an e-fat contiguous polytope, $\varepsilon \leq 1/2e$ the inverse of a positive integer and $\varepsilon' = \frac{\varepsilon}{1+e^2\varepsilon}$, then the following inequality holds:*

$$|P_\varepsilon| \leq \left|P_{\varepsilon'}^{-e\varepsilon'}\right|.$$

Proof. As $P_{\pi'}$ is e-fat, there exists $\vec{u} \in \mathbb{N}^n$ such that for each equation $\sum_{i=j}^{k} t_i \in [A,B]$ defining P it holds that

$$eA + 1 \leq \sum_{i=j}^{k} u_i \leq eB - 1. \tag{5.5}$$

The mapping $\vec{t} \mapsto \frac{\varepsilon'}{\varepsilon}\vec{t} + \varepsilon' e\vec{u}$ is injective, we show that it maps P_ε into $P_{\varepsilon'}^{-e\varepsilon'}$, which yields the expected inequality on cardinalities $|P_\varepsilon| \leq \left|P_{\varepsilon'}^{-e\varepsilon'}\right|$.

By definition $\vec{t} \in P_\varepsilon$ iff $\vec{t}/\varepsilon \in \mathbb{N}^n$ and

$$A \leq \sum_{i=j}^{k} t_i \leq B. \tag{5.6}$$

Now, one can remark that $\frac{1}{\varepsilon} + e^2 = \frac{1}{\varepsilon'}$ and then combining inequalities as follows $\varepsilon'[\frac{1}{\varepsilon} \times (5.6) + e \times (5.5)]$ gives

$$A + e\varepsilon' \leq \sum_{i=j}^{k} \frac{\varepsilon'}{\varepsilon} t_i + \varepsilon' u_i \leq B - e\varepsilon'.$$

We conclude that $\vec{t} \mapsto \frac{\varepsilon'}{\varepsilon}\vec{t} + \varepsilon'\vec{u} \in P_{\varepsilon'}^{-e\varepsilon'}$ for every $\vec{t} \in P_\varepsilon$. □

Lemma 38 ([AD09b])**.** *If P is e-contiguous then $\mathcal{B}_\varepsilon^{NE}(P_\varepsilon^{-e\varepsilon}) \subseteq P$ and passing to volumes: $|P_\varepsilon^{-e\varepsilon}| \leq \mathtt{Vol}(P)\varepsilon^n$.*

End of the proof. The three inequalities of the sketch of the proof are given by lemma 36, 37 and 38. Combining the three inequalities we get:

$$|P_{\pi,\varepsilon}| \leq \left(\varepsilon'^{-n'}\right) V_{\pi'}.$$

Recall that $\phi : \pi \mapsto \pi'$ is an injection and thus

$$|L_{n,\varepsilon}| = \sum_{\pi \in \Delta^n} |P_{\pi,\varepsilon}| \leq \left(\varepsilon'^{-n'}\right) \sum_{\pi' \in E^{n'}} V_{\pi'} = \left(\varepsilon'^{-n'}\right) \mathtt{Vol}(L_{n'}).$$

If we take $\lim_{n\to\infty} \frac{1}{n} \log_2(.)$ in the previous inequality we obtain

$$h_\varepsilon \leq \left(1 + \frac{c}{b}\right)\left(\log_2\left(\frac{1}{\varepsilon'}\right) + \mathcal{H}\right).$$

The right-hand side is equal to $\log_2 \frac{1}{\varepsilon} + \mathcal{H} + \log_2(1 + e^2\varepsilon)) + \frac{c}{b} O(\log_2 \frac{1}{\varepsilon})$. We choose $b = \varepsilon^{-\frac{1}{3}} (\log_2 \frac{1}{\varepsilon})^{\frac{1}{3}}$, then $\log_2(1 + e^2\epsilon)$ and $\frac{c}{b} \log_2 \frac{1}{\varepsilon}$ are $O\left(\varepsilon^{\frac{1}{3}} (\log_2 \frac{1}{\varepsilon})^{\frac{2}{3}}\right)$. Indeed $e = O(b)$ and then $\log_2(1 + e^2\epsilon) = O(b^2\varepsilon) = O\left(\varepsilon^{\frac{1}{3}} (\log_2 \frac{1}{\varepsilon})^{\frac{2}{3}}\right)$. □

Over and under ε-discretization

In [AD09a], an over and under-approximation of the language were designed for timed automata satisfying the progress cycle condition (mentioned above in Chapter 3 and 4). This condition states that there exists $D > 0$ such that each path of the TRG of length greater than D resets all the clocks. In that case we say that the TRG is D-progressive. Remark that if a TRG is D-progressive for some D then it is also D'-progressive for every $D' \geq D$. We call a progressive timed sofic shift the shift spaces of a right-resolving LCTRG which is D-progressive.

The following theorem is a corollary of the previous proof. It is based on ideas of [AD09b], improving this work by stating the convergence of under and over-approximation when the discretization step tends to 0.

Theorem 27 (over and under approximation)**.** *Let S be a progress timed sofic shift. If $\mathcal{H}(S) > -\infty$ then for all positive small enough ε, one can compute ε-discrete sofic shifts S_ε^- and S_ε^+ that verify $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^-) \subseteq S \subseteq \mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$, and $\mathcal{H}(S) + \log \frac{1}{\varepsilon} = h(S_\varepsilon^-) + o(1) = h(S_\varepsilon^+) + o(1)$.*

Sketch of the proof. Let $(\mathcal{G}, \mathtt{Lab})$ be the timed region graph recognizing S. We take $e = \varepsilon^{-\frac{1}{3}} (\log_2 \frac{1}{\varepsilon})^{\frac{1}{3}}$ as in the previous proof and ε small enough to have S satisfying the e-progress condition.

under-approximation We define the $e\varepsilon$-under-approximation of $\mathcal{G}$ denoted by $\mathcal{G}^{-e\varepsilon}$ from $\mathcal{G}$ by replacing every constraint of the form $x \in [A, B]$ by $x \in [A, B - e\varepsilon]$. $\mathcal{G}^{-e\varepsilon}$ is not truly a LCTRG since it has non integer bounds in its guards and since it is no more decomposed in regions. Nevertheless one can define runs and shift space $S^{-e\varepsilon}$ associated to this object in

the same way as for LCTRG. The discretization procedure of section 5.4.3 yields a sofic shift denoted by $S_\varepsilon^- =_{\text{def}} S_\varepsilon^{-e\varepsilon}$. We must show that $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^-) \subseteq S$ and $\mathcal{H}(S)+\log\frac{1}{\varepsilon} = h(S_\varepsilon^-)+o(1)$. The former result is a consequence of [AD09b]. The latter is a corollary of the previous proof. Indeed for each n it holds that:

$$|S_n, \varepsilon| \leq \sum_\pi \left|P_{\pi,\varepsilon'}^{-e\varepsilon'}\right| \leq \left|S_{n,\varepsilon'}^{-e\varepsilon'}\right| \tag{5.7}$$

where the definition of ε' and the first inequality is given in Lemma 37 while the second one is obtained by set inclusion the shrunk polytope $P_\pi^{-e\varepsilon'}$ is always included in the corresponding polytope for $\mathcal{G}^{-e\varepsilon}$. Taking $\lim_{n\to\infty}\frac{1}{n}\log_2(.)$ in (5.7) we get $h_\epsilon(S) = h(S_\varepsilon^-)+o(1)$ which yields by virtue of Theorem 25: $\mathcal{H}(S)+\log\frac{1}{\varepsilon} = h(S_\varepsilon^-)+o(1)$

over-approximation the over-approximation can be proven in a similar way. We define the $e\varepsilon$-over-approximation of $\mathcal{G}$ denoted by $\mathcal{G}^{+e\varepsilon}$ from $\mathcal{G}$ by replacing every constraint of the form $x \in [A, B]$ by $x \in [A \dot{-} e\varepsilon, B]$ (where for two numbers $x, y \geq 0$, $x \dot{-} y = \max(0, x-y)$). We denote $S^{+e\varepsilon}$ the shift space associated to $\mathcal{G}^{+e\varepsilon}$. Let $S_\varepsilon^+ = S_\varepsilon^{+e\varepsilon}$. We show that $S \subseteq \mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$ and $\mathcal{H}(S)+\log\frac{1}{\varepsilon} = h(S_\varepsilon^+)+o(1)$ The former result is a consequence of [AD09b].

Lemma 36 and 37 can be adapted to upper bound $|P_{\pi,\varepsilon}^{+e\varepsilon}|$ by $|P_{\pi',\varepsilon'}|$. Thus, For every n it holds that:

$$\left|S_{n,\varepsilon}^{+e\varepsilon}\right| \leq \sum_{\pi\in\Delta^n} \left|P_{\pi,\varepsilon}^{+e\varepsilon}\right| \leq |S_{n,\varepsilon'}| \tag{5.8}$$

where the first inequality is obtained by set inclusion: the "bloated" polytope $P_\pi^{+e\varepsilon'}$ always includes the corresponding polytope for $\mathcal{G}^{+e\varepsilon}$. Taking $\lim_{n\to\infty}\frac{1}{n}\log_2(.)$ in (5.8) we get $h_\epsilon S = h(S_\varepsilon^+)+o(1)$ which yields by virtue of Theorem 25: $\mathcal{H}(S)+\log\frac{1}{\varepsilon} = h(S_\varepsilon^+)+o(1)$

We pose $S_\varepsilon^- = S_\varepsilon^{-e\varepsilon}$ and $S_\varepsilon^+ = S_\varepsilon^{+e\varepsilon}$. The sequence of inclusions is proved in [AD09b] as well as the following sequence of inequalities $h(S_\varepsilon^-) \leq \mathcal{H}(S)+\log\frac{1}{\varepsilon} \leq h(S_\varepsilon^+)$.

As a corollary of the previous proof $h(\varepsilon) \leq h(S_\varepsilon^{-e\varepsilon})+o(1)$ (since the e-fat polytopes involved in Lemmas 36 and 37 form a subset of all the polytopes defining $S^{-e\varepsilon}$).

Lemma 36 and 37 can be adapted to upper bound $|P_{\pi,\varepsilon}^{+e\varepsilon}|$ by $|P_{\pi',\varepsilon'}|$. This yields the inequalities on entropies $h(S_\varepsilon^+) \leq h(S_\varepsilon)+o(1)$ which concludes the proof.

□

5.4.4 Metric mean dimension

Definitions

The *metric mean dimension* [LW00] of a dynamical system $((X,d),f)$ is defined by:

$$\mathbf{mdim}(X) = \liminf_{\varepsilon\to 0} \frac{\log_2 h_\varepsilon^S(X)}{\log_2(1/\varepsilon)}$$

One can replace $h_\varepsilon^S(X)$ by $h_\varepsilon^R(X)$ in the preceding definition using Lemma 24.

Moreover for timed sofic shifts the different ε-entropies can be related as follows:

Proposition 26. *For all $\varepsilon' \geq \varepsilon$ the following inequalities hold:*

$$h_{2\varepsilon'} \leq h^S_{2\varepsilon} \leq h^N_{\varepsilon} \leq h_{\varepsilon}$$

Proof. The points of $X_{n,\varepsilon'}$ are 2ε separated which proves the first inequality. The second inequality is a straightforward corollary of Lemma 24. To prove the third inequality it suffices to prove that for all $n \in \mathbb{N}$, $X_{n,\varepsilon}$ is an ε-net of X_n. We will adapt a method used in [HMP92]. The setting of [HMP92] considers increasing sequences of dates when events occur instead of delays between events. There is a one-to-one correspondence ϕ between n-uplet of delays and n-uplets of dates defined by $\phi(t_1, ..., t_n) = (t_1, t_1 + t_2, ..., t_1 + t_2 + ... + t_n)$ and with ϕ^{-1} defined by $\phi^{-1}(T_1, ..., T_n) = (T_1, T_2 - T_1, ..., T_n - T_{n-1})$. One can remark that ε-discrete points are also in one to one correspondence by ϕ. The guards along a path give the following inequations on dates $T_k - T_j \in [A, B]$ (this corresponds to a constraint $x \in [A, B]$ checked at the k^{th} transition and with the last reset of x done in the j^{th} transition).

For all real T we denote by $[T]$ the closest multiple of ε to T ($|[T] - T| \leq \frac{\varepsilon}{2}$): $[T] = \begin{cases} \varepsilon\lfloor T/\varepsilon \rfloor & \text{if } T \leq \varepsilon\lfloor T/\varepsilon \rfloor + \frac{\varepsilon}{2} \\ \varepsilon(\lfloor T/\varepsilon \rfloor + 1) & \text{otherwise} \end{cases}$

One can remark that $T_k - T_j \in [A, B]$ implies $[T_k] - [T_j] \in [A, B]$.

Let $(t_1, \delta_1), ..., (t_n, \delta_n) \in X_n$. We denote by $(T_1, ..., T_n) = \phi(t_1, ..., t_n)$ and thus $([T_1], ..., [T_n])$ satisfy the constraints $[T_k] - [T_j] \in [A, B]$. We denote by $(u_1, ..., u_n) = \phi^{-1}([T_1], ..., [T_n])$ then $(u_1, \delta_1), ..., (u_n, \delta_n) \in X_{n,\varepsilon}$. Since $|u_1 - t_1| = |[T_1] - T_1| \leq \frac{\varepsilon}{2}$ then for all $i \in \{2, .., n\}$ we have $|u_i - t_i| = |([T_{i+1}] - [T_i]) - (T_{i+1} - T_i)| \leq |[T_{i+1}] - T_{i+1}| + |[T_i] - T_i| \leq \varepsilon$. Finally every timed word of X_n is at most ε far apart from a timed word of $X_{\varepsilon,n}$. $X_{\varepsilon,n}$ is thus an ε net for X_n which concludes the proof. □

Corollary 3. *The metric mean dimension of a timed sofic shift can be defined with h_ε:*

$$\mathbf{mdim} = \liminf_{\varepsilon \to 0} \frac{\log_2 h_\varepsilon}{\log_2(1/\varepsilon)} = \liminf_{\varepsilon \to 0} \limsup_{n \to \infty} \frac{\log_2 \left(\sum_{\pi \in \Delta^n} |P_{\pi,\varepsilon}|\right)}{n \log_2(1/\varepsilon)}. \tag{5.9}$$

Remark that if $X \subseteq Y$ then $\mathbf{mdim}(X) \leq \mathbf{mdim}(Y)$ and that $\mathbf{mdim}(\mathbb{A}^{\mathbb{Z}}) = 1$ (where $\mathbb{A} = [0, M] \times \Sigma$ is our alphabet of interest). Thus every subshift of $\mathbb{A}^{\mathbb{Z}}$ has a mean dimension lower or equal to 1.

Examples

The following examples of timed sofic shifts are recognized by LCTRGs depicted in Figure 5.2.

Example 5 (zero choice)**.** *Let $\mathbb{A} = [0, 1] \times \{a\}$ and the set of forbidden factors be given by $O_1 = \{(a, t) \mid t < 1\}$. The only element of the shift space X_O is $(1, a)^{\mathbb{Z}}$. This shift space has metric mean dimension zero.*

Example 6 (One half a choice)**.** *Let $\mathbb{A} = [0, 1] \times \{a, b\}$ and the set of forbidden factors be given by $O_2 = \{(a, t)(b, t') \mid t < 1, t' \in [0, 1]\} \cup \{(l, t)(l, t') \mid l \in \{a, b\},\ t, t' \in [0, 1]\}$. The shift*

space X_O is the set of bi-infinite words of the form $[(a_i,t_i)(b_{i+1},t_{i+1})]_{i\in 2\mathbb{Z} \text{ or } 2\mathbb{Z}+1}$ with $t_i=1$. Its metric mean dimension is $\mathbf{mdim}=\frac{1}{2}$. *This corresponds to the intuition that a full choice can be made half of the time, since delays before events a are always in the 0-dimensional singleton $\{1\}$ while delays before events b are to be in the 1-dimensional interval $[0,1]$.*

Examples 7 and 8 below are more subtle and involve Zeno phenomena and simplices as we have seen in Chapter 3. To treat these examples we need first a couple of results:

Lemma 39 (Few points in a simplex)**.** *The number of ε-discrete points in a simplex described by inequalities $0\le u_1\le\cdots\le u_n\le M$ (resp. by inequalities $\sum_{i=1}^n u_i\le M$ and $u_i\ge 0$) is $\binom{n+M/\varepsilon}{n}$ and $(1/n)\log_2\left[\binom{n+M/\varepsilon}{n}\right]\to_{n\to+\infty}0$.*

Proof. There is $\binom{n+M/\varepsilon}{n}$ possibilities to choose n indices $i_1<\ldots<i_n$ among $\{1,\ldots,n+M/\varepsilon\}$. For $j=1..n$ define $u_j=(i_j-j)\varepsilon$ and get $0\le u_1\le\cdots\le u_n\le M$. It remains to remark that the mapping $(i_1,\ldots,i_n)\mapsto(t_1,\ldots,t_n)$ is a bijection (one can check that the following function is an inverse for it: $u_j\mapsto u_j/\varepsilon+j$ for $j=1..n$). □

Example 7 (Zeno: zero choice in the average)**.** *Let $\mathbb{A}=[0,1]\times\{b\}$ and the set of forbidden factors be given by $O_n=\{(b,t_1)\cdots(b,t_n)\mid t_1+\ldots+t_n>1\}$ ($n\ge 2$). The shift space X_O is the set of bi-infinite word of the form $(b,t_i)_{i\in\mathbb{Z}}$ satisfying the (bi-infinite) Zeno condition $\sum_{i\in\mathbb{Z}}t_i\le 1$. The number of discrete points in $P_{\pi,\varepsilon}$ for the only path π of length n is given by Lemma 39 above: $|P_{\pi,\varepsilon}|=\binom{k+1/\varepsilon}{k}$. Plugging this in (5.9) yields a metric mean dimension zero. Intuitively there are less and less choices as n increases.*

Example 8 (Thin, yet full mean dimensional)**.** *Let $\mathbb{A}=[0,1]\times\{a,b\}$ and the set of forbidden factors be given by $O_1=\{(a,t)\mid t<1\}$ and $O_n=\{(b,t_1)\cdots(b,t_n)\mid t_1+\ldots+t_n>1\}$. Every bi-infinite words of X_O has its delays corresponding to the event a equals to 1 (as in example 5) and the sum of delays of blocks of consecutive b is bounded by 1 (as in example 7). Every words containing an a yields a volume 0, the only word of length n that contributes to $\mathtt{Vol}(X_n)$ is b^n it yields a volume: $\mathtt{Vol}(X_n)=1/n!$ and thus an entropy $\mathcal{H}=-\infty$. However this shift space has metric mean dimension 1 and thus many discrete points. Indeed, for every positive integer m, the path in $(a^{m-1}b)^*$ yields a metric mean dimension equal to $(m-1)/m$ and thus* $\mathbf{mdim}\ge 1-1/m$ *for every $m>0$.*

Metric mean dimension and thickness.

Here we prove that when it is fleshy, a timed sofic shift is thick ($\mathcal{H}>-\infty$) if and only if it has a full metric mean dimension ($\mathbf{mdim}=1$). The condition of fleshiness is necessary as there exists thin ($\mathcal{H}=-\infty$) full metric mean dimensional fleshy timed sofic shift as we have seen with example 8.

Theorem 28. *For fleshy timed sofic shift, thickness is equivalent to maximal metric dimension.*

$$\mathcal{H}>-\infty \text{ iff } \mathbf{mdim}=1.$$

Proof of $\mathcal{H} > -\infty \Rightarrow \mathbf{mdim}_M = 1$

The statement $\mathcal{H} > -\infty \Rightarrow \mathbf{mdim}_M = 1$ is obtained by taking $\liminf_{\varepsilon\to 0} \frac{\log_2(.)}{\log_2(1/\varepsilon)}$ in both sides of inequality (5.4) stated in Lemma 32.

Proof of $\mathbf{mdim}_M = 1 \Rightarrow \mathcal{H} > -\infty$ (difficult part)

This proof is an adaptation of results of Chapter 3 where we replace volumes of sets by cardinalities of ε-discretization of the corresponding sets. We refer to this chapter when a definition is needed e.g. monoid of orbit graph, forgetful idempotent, factorization forest, etc.

By virtue of Theorem 5 of Chapter 3 the existence of a forgetful cycle is a necessary and sufficient condition to have thickness ($\mathcal{H} > -\infty$). We will thus prove that if there is no forgetful cycle then $\mathbf{mdim} < 1$.

The idea of the proof is very similar to that of Theorem 5. Path that contains several times a non-forgetful idempotent has polytope with few discrete points (Proposition 27). The Simon's factorization forest theorem (Theorem 4) ensures that in any long enough path an idempotent is iterated. Thus if there is no forgetful cycle, there is roughly few discrete points. This formally corresponds to a mean dimension lower than 1.

Proposition 27. *Let $\pi_1, \dots, \pi_k$ be k cycles of Δ^* such that $\mu(\pi_1), \dots, \mu(\pi_k)$ are all equal to a same non forgetful idempotent of $\mathcal{M}$, then $|P_{\pi_1\dots\pi_k,\varepsilon}| \leq (1 + M/\varepsilon)^{n-k}\binom{k+M/\varepsilon}{k}$ where $n = |\pi_1| + \cdots + |\pi_k|$.*

The proof is omitted as it follows the same line as that of Lemma 11.

Now we introduce a function $L(n, h, \varepsilon)$ (whose second parameter will be clarify below) similar to that of (5.9) yet easier to handle. It is related to mean dimension in Lemma 40 and asymptoticly upper bounded in Proposition 28.

Let $\mathtt{LC}(\pi, \varepsilon) =_{\text{def}} \log_2 |P_{\pi,\varepsilon}| - |\pi| \log_2(1 + M/\varepsilon)$. This function is subadditive and non-positive, i.e. $\mathtt{LC}(\pi_1\pi_2) \leq \mathtt{LC}(\pi_1) + \mathtt{LC}(\pi_2) \leq 0$. Let $L(n, h, \varepsilon)$ be the maximum of $\mathtt{LC}(\pi, \varepsilon)$ over paths π of length n that do not contain forgetful idempotents and admit a factorization forest of height at most h. In particular, $L(n, h(\mathcal{M}), \varepsilon) = \max_{\pi\in\Delta^n} \log_2 |P_{\pi,\varepsilon}| - n \log_2(1 + M/\varepsilon)$ where $h(\mathcal{M})$ is the height of the monoid $\mathcal{M}$ (defined in Chapter 3). The mean dimension is related to this latter quantity as follows:

Lemma 40. *If $\mathcal{M}$ does not contain forgetful idempotent then*

$$\liminf_{\varepsilon\to 0} \limsup_{n\to\infty} \frac{L(n, h(\mathcal{M}), \varepsilon)}{n \log_2(1/\varepsilon)} = \mathbf{mdim} - 1.$$

Proof. We relate the left-hand side of the equality to prove to the right-hand side of (5.9). We remark that $\max_{\pi\in\Delta^n} |P_{\pi,\varepsilon}| \leq \sum_{\pi\in\Delta^n} |P_{\pi,\varepsilon}| \leq |\Delta^n| \max_{\pi\in\Delta^n} |P_{\pi,\varepsilon}|$. Then, $L(n, h(\mathcal{M}), \varepsilon) + n \log_2(1 + M/\varepsilon) \leq \log_2 \left(\sum_{\pi\in\Delta^n} |P_{\pi,\varepsilon}|\right) \leq L(n, h(\mathcal{M}), \varepsilon) + n \log_2(1 + M/\varepsilon) + n \log_2(\Delta)$.
We divide by $n \log_2(1/\varepsilon)$, take $\liminf_{\varepsilon\to 0} \limsup_{n\to\infty}(.)$ and obtain the expected result. □

It remains to prove that the quantity $\frac{L(n,h(\mathcal{M}),\varepsilon)}{n\log_2(1/\varepsilon)}$ becomes negative when $n \to +\infty$ and $\varepsilon \to 0$. This is implied by the following proposition.

Proposition 28. *If $\mathcal{M}$ does not contain forgetful idempotents then for any height h, there exists $\alpha_h > 0$ such that for all $n > \frac{1}{\alpha_h}$, for all $\varepsilon > 0$, the inequality $L(n,h,\varepsilon) \leq -\alpha_h n \log_2(1+M/\varepsilon)$ holds.*

Proof. We will define α_h by induction on the height h. Let a be a factorization forest of height h with n leaves and $\pi_1, \ldots, \pi_k$ be the children of the root. We distinguish two disjoint cases:

1. There are more than $m =_{\text{def}} n\alpha_{h-1}$ subtrees having less than $1/\alpha_{h-1}$ leaves.
2. There are less than $m = n\alpha_{h-1}$ subtrees with less than $1/\alpha_{h-1}$ leaves. Here the juicy part (sons with enough leaves to satisfy induction hypothesis) has more than $\frac{n}{2}$ leaves.

- In the first case: root is an idempotent node and we can apply Lem. 27:

$$\mathtt{LC}(\pi) \leq \log_2\left((1+M/\varepsilon)^{n-k}\right) + \log_2 \binom{k+M/\varepsilon}{k} - n\log_2(1+M/\varepsilon)$$

which after simplifications yields:

$$\mathtt{LC}(\pi) \leq -k\log_2(1+M/\varepsilon) + \log_2 \binom{k+M/\varepsilon}{k} \leq -m\log_2(1+M/\varepsilon) + \log_2 \binom{m+M/\varepsilon}{m}.$$

The constant α_h can be chosen small enough such that for every $n \geq \frac{1}{\alpha_h}$:

$$\mathtt{LC}(\pi) \leq -m\log_2(1+M/\varepsilon) + \log_2 \binom{m+M/\varepsilon}{m} \leq -n\alpha_h \log_2(1+M/\varepsilon).$$

- In the second case: $\mathtt{LC}(\pi) \leq \sum_{i=1}^{k} L(n_i,h_i,\varepsilon) \leq \sum_{n_i \geq N_{h-1,2C}} L(n_i,h_i,\varepsilon)$. We apply the induction hypothesis:

$$\mathtt{LC}(\pi) \leq -\alpha_{h-1}\log_2(1+M/\varepsilon) \sum_{n_i \geq \alpha_{h-1}} n_i$$
$$\leq -\alpha_{h-1}\frac{n}{2}\log_2(1+M/\varepsilon) \leq -\alpha_h n \log_2(1+M/\varepsilon)$$

where α_h is chosen such that $\alpha_h \leq \alpha_{h-1}/2$.

□

At the end, when $\mathcal{H} = -\infty$ then $\mathbf{mdim} \leq 1 - \alpha_{h(\mathcal{M})} < 1$. □

5.5 Sliding block codes

In this section $\mathcal{C}$ and $\mathcal{C}'$ denotes two metric compact alphabet shift-spaces (with respective distances d and d'), X and Y denotes sub-shifts of $\mathcal{C}^{\mathbb{Z}}$ and $\mathcal{C}'^{\mathbb{Z}}$ respectively. Given a function ψ from X to $\mathcal{C}'$ we denote by $\psi^\infty : X \to \mathcal{C}'^{\mathbb{Z}}$ defined by $(\psi^\infty(x))_i = \psi^\infty(\sigma^i(x))$. Such functions are those that commute with the shifts i.e. $\sigma_{\mathcal{C}'^{\mathbb{Z}}} \circ \phi = \phi \circ \sigma_X$. We denote by $\mathcal{F}(X, \mathcal{C}')$ the function from X to $\mathcal{C}'$ and by $\mathcal{SC}(X, \mathcal{C}'^{\mathbb{Z}})$ the function from X to $\mathcal{C}'^{\mathbb{Z}}$ that commutes with the shift.

Lemma 41. *The mapping $\psi \mapsto \psi^\infty$ is a bijection from $\mathcal{F}(X, \mathcal{C}')$ to $\mathcal{SC}(X, \mathcal{C}'^{\mathbb{Z}})$ whose inverse is $\phi \mapsto (x \mapsto \phi(x)_0)$. Moreover ψ is continuous iff so is ψ^∞.*

Proof. By definition of ψ^∞ it holds that $\psi^\infty(x)_0 = \psi(x)$ and thus the two mappings defined above are mutual inverses. If ψ is continuous then for every sequence $(x^n)_{n\in\mathbb{N}}$ of elements of X, the convergence $x^n \mapsto_{n\to+\infty} x \in X$ implies that for every $i \in \mathbb{N}$, $\psi^\infty(x^n)_i = \psi^\infty(\sigma^i(x^n)) \mapsto_{n\to+\infty} \psi^\infty(\sigma^i(x)) = \psi^\infty(x)_i$. This means that ψ^∞ is continuous when so is ψ. The converse is straightforward. □

Every continuous functions from a shift space $X \subseteq \mathcal{C}^{\mathbb{Z}}$ to a shift space $Y \subseteq \mathcal{C}'^{\mathbb{Z}}$ that commutes with the shift is called a *morphism*. By virtue of the preceding lemma, every morphism is of the form ψ^∞ with ψ a continuous function from X to $\mathcal{C}'$.

We say that ψ is a $(2n+1)$-block function when for every x, $\psi(x)$ depends only on the $(2n+1)$-central factor $x_{[-n..n]}$ i.e. there exists a function $f : \mathcal{C}^{2n+1} \to \mathcal{C}'$ such that for every x, $\psi(x) = f(x_{[-n..n]})$. One can remark that ψ is continuous iff so is f. A function ϕ that is equal to some ψ^∞ with ψ a (continuous) block function is called a (continuous) *sliding block code*.

The following famous theorem gives a characterization of the morphisms of finite-alphabet shift spaces as sliding block codes.

Theorem 29 (Curtis-Hedlund-Lyndon)**.** *Let X and Y be two finite-alphabet shift spaces. A function $\varphi : X \to Y$ is a morphism if and only if it is a sliding block code.*

The following proposition below shows that there are morphisms which are not sliding (finite) block codes and thus that the Curtis-Hedlund-Lyndon theorem does not hold for compact alphabet shift spaces in general.

Proposition 29. *Let $\psi : [0,1]^{\mathbb{Z}} \to [0,1]$ defined by $\psi(x) = \frac{1}{3}\sum_{i\in\mathbb{Z}} \frac{x_i}{2^{|i|}}$ then ψ^∞ is an endomorphism of $[0,1]^{\mathbb{Z}}$ (which is not a sliding block code).*

Proof. We show that ψ maps converging sequences to converging sequences and is thus continuous. Let $(x^n)_{n\in\mathbb{N}}$ be a sequence of bi-infinite words of $[0,1]^{\mathbb{Z}}$ that converges toward a bi-infinite word $x \in [0,1]^{\mathbb{Z}}$. We show that $\sum_{i\in\mathbb{Z}} \frac{x^n_i}{2^{|i|}} \to_{n\to+\infty} \sum_{i\in\mathbb{Z}} \frac{x_i}{2^{|i|}}$. For every $i \in \mathbb{N}, n \in \mathbb{N}$, $\frac{x^n_i}{2^{|i|}} \leq \frac{1}{2^{|i|}}$ and thus we are done by applying the dominate convergence theorem. Thus ψ is continuous and by virtue of Lemma 41 ψ^∞ is an endomorphism of $[0,1]^{\mathbb{Z}}$. □

Even if the Curtis-Hedlund-Lyndon theorem does not hold for compact alphabet shift spaces, Theorem 30 below states that we can approximate every morphism by continuous sliding block codes (the target shift space must be a full shift $\mathcal{C}'^{\mathbb{Z}}$). We first state a similar result for function from $\mathcal{C}^{\mathbb{Z}}$ to $\mathcal{C}'$.

Lemma 42. *Every continuous function from X to $\mathcal{C}'$ is a uniform limit of block continuous functions from X to $\mathcal{C}'$.*

Proof. Let $\psi \in \mathcal{F}(X, \mathcal{C}')$ be a continuous function. The n^{th} *truncation* $f_n : x \mapsto x_{[-n..n]}$ is a continuous function from X to $\mathcal{C}^{2n+1}$. For $n \in \mathbb{N}$, let $g_n : X_{2n+1} \to X$ be a function such that $g_n(w)$ is an element of X with central factor w i.e. $g_n(w)_{[-n..n]} = w$ (construction of g_n requires the axiom of choice). The function ψ_n is a $(2n+1)$-block continuous function. It remains to prove that $(\psi_n)_{n\in\mathbb{N}}$ converges toward ψ. As ψ is continuous between two compacts, it is also uniformly continuous. Thus for an arbitrary ε, let $\delta \to \bar{d}(x, x') \leq \delta$ implies that $d'(\psi(x), \psi(x'))$. We take $n \geq \log_2(\texttt{diam}(\mathcal{C})/\delta) - 1$ so that for every $x \in \mathcal{C}^{\mathbb{Z}}$, $\bar{d}(g_n \circ f_n(x), x) \leq \texttt{diam}(\mathcal{C})2^{-(n+1)} \leq \delta$ and thus $d'(\psi_n(x), \psi(x)) \leq \varepsilon$. We are done the sequence $(\psi_n)_{n\in\mathbb{N}}$ of continuous sliding block codes uniformly converges toward ψ. □

Theorem 30. *Every morphism from a shift space X to a full shifts $\mathcal{C}'^{\mathbb{Z}}$ is the uniform limit of continuous sliding block codes.*

Proof. By Lemma 41 every morphism is of the form ψ^∞ with ψ a continuous function of $\mathcal{F}(X, \mathcal{C}')$. By Lemma 42 just above there exists a sequence $(\psi_n)_{n\in\mathbb{N}}$ of continuous block functions that uniformly converges toward ψ. For every x and i, the i^{th} coordinates of $\psi^\infty(x)$ and $\psi_n^\infty(x)$ are $\psi(\sigma^i(x))$ and $\psi_n(\sigma^i(x))$ respectively. Thus, for every x:

$$\sup_{x\in\mathcal{C}^{\mathbb{Z}}} \bar{d'}\left(\psi^\infty(x), \psi_n^\infty(x)\right) = \sup_{x\in\mathcal{C}^{\mathbb{Z}}} \sup_{i\in\mathbb{Z}} \frac{1}{2^{|i|}} d'\left(\psi(\sigma^i(x)), \psi_n(\sigma^i(x))\right) \leq \sup_{y\in\mathcal{C}^{\mathbb{Z}}} d'\left(\psi(y), \psi_n(y)\right).$$

We can conclude: the sequence of continuous sliding block codes $(\psi_n^\infty)_{n\in\mathbb{N}}$ converges toward the morphism ψ^∞. □

5.6 Conclusion and perspectives

In this chapter we designed a general symbolic dynamics framework for the volumetry of timed languages. We close the problem partially left open in [AD09b] of computability of the entropy by discretization of the entropy. We adapted to sofic timed shift the metric mean dimension of Lindenstrauss, Weiss and Gromov [LW00]. Finally we study morphisms of compact alphabet shift spaces.

5.6.1 Open problems

What is the good notion of morphism/sliding block codes?

Here we gave a very general definition of morphisms as functions that preserve the structure of dynamical system: they are continuous, in order to preserve compactness, and they commute

with the shift, in order to preserve dynamics. This definition is maybe too general and it would be an interesting task to define morphism that preserves more structural aspects i.e. distance, and even volume measure. It would be nice to define morphisms that leave the class of timed sofic shift invariant.

In the next chapter, motivated by coding purposes, we add conditions on these functions to loosely preserve distances and volume measures. Yet they do not leave the sofic timed shifts (and languages) invariant.

A machine-independent characterization of timed sofic shifts

The definition of a timed sofic shift as given in this chapter requires a "machine" (the LCTRG $(\mathcal{G}, \texttt{Lab})$). There is a beautiful machine-independent characterization of sofic shifts as the shift spaces that have only a finite number of follower sets (see [LM95]). This results corresponds to the characterization of regular languages as languages that have a finite number of Nerode's equivalence classes. In [BL12], the authors introduce an equivalence relation playing the role of the Nerode's equivalence for timed languages. Their main results state that deterministic regular timed languages are those for which the set of equivalence classes is orbite finite (a notion introduced and explained by these authors). It would be interesting to marry the new theory of [BL12] with ours and get a machine-independent characterization of timed sofic shift.

What about shifts of finite type?

Fundamental objects of symbolic dynamics are so-called shifts of finite types (SFT): the shift spaces that can be defined with finite sets of forbidden factors. In fact such shifts are conjugated to edge shifts. That is why we are able to lift a broad part of results to the timed case without referring to SFTs (but referring to graphs and edge shifts). The question of what would be a good notion of timed SFT is still open.

About metric mean dimension

In [AD10], Asarin and Degorre proposed that timed automata with non-fleshy transitions should be associated two size measures. The first one is the asymptotic proportion of fleshy transions in the "best" runs (here, let us call it "structural mean dimension" and denote it by **smdim**). The second one is the volumetric entropy adapted to this dimension (let us denote it by H_v).

[AD10] characterizes the structural mean dimension as the spectral radius in the Max-Plus algebra of a kind of adjacency matrix associated to the automaton (its coefficients are 1 if there is a fleshy transition, 0 if there are only non fleshy ones and $-\infty$ if no transition exists). However this dimension does not detect Zeno behaviors as in Example 7. Indeed, in this example, **smdim** $= 1$, yet according Weiss' and Lindenstrauss' intuition of mean dimension, it should be 0. Our notion of metric mean dimension, directly adapted from [LW00], captures such Zeno behaviours.

It is shown in [AD10] that provided that $\mathbf{smdim} > 0$ and $H_v > -\infty$ a kind of ϵ-entropy is equal to $\mathbf{smdim} \log_2(1/\varepsilon) + H_v + o(1)$. This results is similar to our Theorem 25 yet several questions remain open to unify our work with that of [AD10].

- How to characterize $H_v > -\infty$? We must adapt for this, theory of Chapter 3 to the case where non-fleshy transitions are unavoidable.
- How the ε-entropy of [AD10] is related to that defined here.
- What is the case of equality? It is easy to see that the mean dimension exposed in the present chapter is always upper-bounded by that of Asarin and Degorre.

There are also open questions dealing with decidability.

- Is there an algebraic characterization of the metric mean dimension?
- Is metric mean dimension computable?
- What real numbers are metric mean dimensions of timed region graphs?

Chapter 6

Toward a Timed Theory of Channel Coding

Abstract of the chapter

The classical theory of constrained-channel coding deals with the following questions: given two languages representing a source and a channel, is it possible to encode source messages to channel messages, and how to realize encoding and decoding by simple algorithms, most often transducers. The answers to this kind of questions are based on the notion of entropy.

In the current chapter, the questions and the results of the classical theory are lifted to timed languages. Using the notion of entropy of timed languages introduced by Asarin and Degorre, the question of timed coding is stated and solved in several settings.

Chapter structure

In section 6.1 we recall basic notions of the discrete theory of constrained-channel coding. In section 6.2 we state our main results on timed theory of constrained-channel coding. In section 6.3 we discuss the rationale, perspectives and applications of this work.

6.1 Theory of channel coding for finite alphabet languages

In this section we give an elementary exposition of some basic notions and results from the theory of constrained-channel coding, see [MRS98, LM95, BPR09, BBM+10] for more details. We refer the reader to the sections 5.2.3 for the definition of sofic languages and their entropy.

6.1.1 Terminology

Most of our coding functions have a special property defined below.

Definition 5 (almost injective)**.** *A (partial) function $\phi : \Sigma^* \to \Gamma^*$ is called* almost injective *with delay $D \in \mathbb{N}$, if for any n and $w, w' \in \Sigma^n$, and $u, u' \in \Sigma^D$ it holds that*

$$\phi(wu) = \phi(w'u') \Rightarrow w = w'.$$

Intuitively, if such a function is used to encode messages, then knowing the code of some message wu one can decode w, i.e. the whole message except its last D symbols. Thus the decoding is possible with delay D. This can be formalized as follows:

Definition 6 (almost inverse)**.** *For an almost injective function $\phi : \Sigma^* \to \Gamma^*$ with delay D its D-*almost inverse *family of functions $\psi_n : \Gamma^* \to \Sigma^n$ is characterized by the following property: for any $w \in \Sigma^n$ and $v \in \Gamma^*$,*

$$w = \psi_n(v) \Leftrightarrow \exists u \in \Sigma^D : \ \phi(wu) = v.$$

Lemma 43. *If the domain of an almost injective function ϕ is extensible and ψ_n is its almost inverse, then ψ_n is a surjection to this domain (constrained to words of length n).*

6.1.2 Coding: the basic case

Let A and A' be two alphabets (source and channel alphabets), $S \subset A^*$ and $C \subset A'^*$ factorial extensible languages and $D \in \mathbb{N}$. The aim is to encode any source message $w \in S$ to a channel message $\phi(w) \in C$. The latter message can be transmitted over the channel.

Definition 7. *An (S, C)-encoding with delay D is a function $\phi : S \to C$ (total on S but not necessarily onto C) such that*

- *it is length preserving: $\forall w \in S, \ |\phi(w)| = |w|$,*
- *it is almost injective with delay D.*

The first condition means that the information is transmitted in real-time (with the transmission rate 1). The second one permits decoding.

A natural question is to find necessary and sufficient conditions on S and C for an (S, C)-encoding (with some delay) to exist. This question can be addressed by comparing the entropy of the languages S and C. Roughly, the channel language should contain at least as much information per symbol as the source language. Formally, we define the *information inequality*:

$$h(S) \leq h(C). \tag{II1}$$

Proposition 30. *Let S and C be factorial and extensible languages. If an (S, C)-encoding exists then* (II1) *necessarily holds.*

Proof. Let $\phi : S \to C$ be an (S, C)-encoding with delay D. By Lemma 43 its almost inverse ψ maps C onto S. More precisely, for every n we have $\psi_n(C_{n+D}) = S_n$. Hence, the cardinalities should satisfy: $|S_n| \leq |C_{n+D}|$. Finally we have

$$\lim_{n\to\infty} \frac{1}{n} \log |S_n| \leq \lim_{n\to\infty} \frac{1}{n} \log |C_{n+D}|$$

and the expected inequality $h(S) \leq h(C)$. □

Figure 6.1: A sofic automaton of a channel C (left) and a $(\{0,1\}^*, C)$-encoder (right)

Thus, (II1) is necessary for existence of the coding. For sofic languages (if the inequality is strict) it is also sufficient. Moreover, the encoding can be realized by a sort of finite-state machine. We present this fundamental result in the following form which is essentially the finite-state coding theorem from [MRS98], Theorem 10.3.7 in [LM95].

Theorem 31. *Let S and C be sofic languages. If the strict version of* (II1) *holds, then there exists an (S, C)-encoding realized by a finite-state transducer which is right-resolving on input and right-closing on output*[1]*.*

The reader is now motivated to get through a couple of definitions.

Definition 8 (transducer)**.** *A transducer is a tuple $\tau = (Q, A, A', \Delta, \mathcal{I}, \mathcal{O})$ with a finite set Q of control states; finite input and output alphabets A and A'; a set of transitions Δ (each transition δ has a starting state δ^- and an ending state δ^+); input and output labeling functions $\mathcal{I} : \Delta \to A$ and $\mathcal{O} : \Delta \to A'$.*

The transducer is said to be *right-resolving* on input whenever for each state $q \in Q$ every two different edges starting from q have different input labels. For such a transducer, the input automaton with a fixed initial state i, i.e. $\mathcal{A}_i = (Q, A, \Delta, \mathcal{I}, i, Q)$ is deterministic, we denote by S_i the language of this automaton.

The transducer is *right-closing* on output with delay D whenever every two paths π and π' of length $D+1$ with the same output label and the same starting state q always have the same initial edge $\pi_1 = \pi'_1$.

A transducer τ satisfying both properties performs the encoding process in a natural way: an input word w of S is read from a state i along a path π_w, and this path determines the output word $\mathcal{O}(\pi_w)$. The function $\phi_i : S_i \to A'^*$ defined as $w \mapsto \mathcal{O}(\pi_w)$ is length preserving and almost injective with delay D.

Example 9. *Consider the source language $S = \{0,1\}^*$ and the channel language C recognized by the sofic automaton on the left of Figure 6.1. The language C is composed by all the words on $\{a, b, c\}$ that do not contain any block bc. The entropy of the source is $h(S) = 1$, and the one of the channel is $h(C) = 2\log\left[(1+\sqrt{5})/2\right] \approx 1.3885$. The information inequality $h(S) < h(C)$ holds and we can encode S in C using the transducer on the right of Figure 6.1.*

[1]Such a transducer is also called a finite-state (S, C)-encoder.

6.1.3 Other coding settings

Similarly to the previous section, other coding settings can be considered. For example, we can transmit information over a channel with some rate $\alpha = \frac{p}{q}$, when q letters of the channel message correspond to p letters of the source message (the previous section corresponds thus to the case $\alpha = 1$).

Definition 9. *An (S, C)-encoding with rate $\alpha \in \mathbb{Q}^+$ and delay D is a function $\phi : S \to C$ (total on S and not necessarily onto C) such that*

- *it is of rate α, i.e. $\forall w \in S,\ \lceil \alpha |\phi(w)| \rceil = |w|$;*
- *it is almost injective (with delay D).*

In this setting, the information inequality takes the form:

$$\alpha h(S) \leq h(C), \tag{II2}$$

and it is a necessary and almost sufficient condition for the code to exist:

Proposition 31. *Let S and C be factorial and extensible languages. If an (S, C)-encoding with rate α exists, then* (II2) *necessarily holds.*

Proposition 32. *Let S and C be sofic languages. If a strict inequality* (II2) *holds, then an (S, C)-encoding of rate α exists. Moreover, it can be realized by a finite-state transducer of rate α.*

We skip here a natural definition of such a transducer.

6.2 Timed coding

Similarly to classical results presented in section 6.1, we will consider several settings for transmission of timed words over a channel. For every setting we will formulate an information inequality, and show that it is necessary and, with some additional hypotheses, sufficient for a coding to exist.

On timed language considered

In this chapter we consider (progressive) timed sofic language as defined in section 5.4.2, i.e. recognized by (progressive) right-resolving labelled closed timed region graphs.

As usual, we work with alphabets of the form $A = [0, M] \times \Sigma$ called here *k-M-alphabet* where Σ is a k-letter alphabet and M a positive integer bound, so that every letter in A corresponds to a real-valued delay (seen as a data) in $[0, M]$ and a discrete event in Σ.

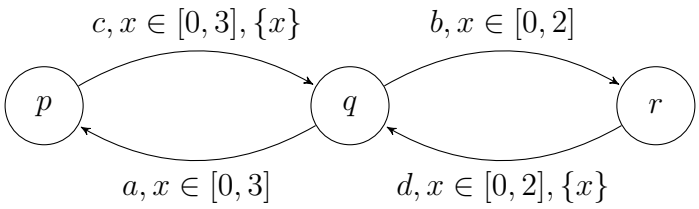

Figure 6.2: An abstract representation of a LCTRG (in fact locations p and r correspond to multiples location of a LCTRG depending whether $x \in [0,1]$, $x \in [1,2]$ or $x \in [2,3]$)

Example 10. *In this chapter we will consider the labelled closed timed region graph given in Figure 6.2. Polytopes associated to words ac and bd are* $P_{ac} = \{(t_1, t_2) \mid t_1 + t_2 \leq 3\}$ *and* $P_{bd} = \{(t_1, t_2) \mid t_1 + t_2 \leq 2\}$*; and their volumes are respectively* 4.5 *and* 2*; and thus* $L_2(q)$*, the sublanguage of* L_2 *of words accepted by runs starting from* q*, has volume* 6.5*. We have* $L_{2n}(q) = (L_2(q))^n$ *whose volume is* 6.5^n*, the entropy of the whole language* $\mathcal{H}(L)$ *is thus at least* $0.5 \log 6.5$ *(in fact it is exactly* $0.5 \log 6.5$*).*

Discretization of languages and entropy

We will use discretization in a three-step reduction scheme:

1. discretize the timed languages S, C with a sampling rate ε to obtain $S_\varepsilon, C_\varepsilon$ (as in section 5.4.3);

2. use a classical coding theorem (Theorem 31) with $S_\varepsilon, C_\varepsilon$;

3. go back to timed languages by taking ε-NE-neighborhood of S_ε and C_ε (as defined in section 5.4.3).

The following lemma is the main tool for this reduction scheme, it is a recap of Theorem 27 of Chapter 4.

Lemma 44. *Let* S *be a progressive timed sofic language. If* $\mathcal{H}(S) > -\infty$ *then for all positive small enough* ε*, one can compute* ε*-discrete sofic languages* S_ε^- *and* S_ε^+ *that verify* $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^-) \subseteq S \subseteq \mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$*, and* $\mathcal{H}(S) + \log \frac{1}{\varepsilon} = h(S_\varepsilon^-) + o(1) = h(S_\varepsilon^+) + o(1)$.

6.2.1 Timed source, discrete channel, approximate transmission

In practice, timed and data words are often transmitted via discrete (finite alphabet) channels. For example, a timed log of events in an operating system (a timed message) can be stored as a text file (ASCII message). The delays in the timed word cannot be stored with infinite precision, thus the coding is necessarily approximate. More precisely, the set of timed source messages w of a length n is infinite, while the set of discrete channel messages

of the same length is finite. For this reason, the coding cannot be injective, and necessarily maps many timed words to a same discrete word. It is natural to require that all the timed words with the same code are ε-close to each other. This justifies Definition 10 below. We give first some notation. For two timed words of same length $w = (t_1, a_1)\dots(t_n, a_n)$ and $w = (t'_1, a'_1)\dots(t'_n, a'_n)$, the distance $d(w, w')$ between w and w' is equal to $+\infty$ if $a_1 \dots a_n \neq a'_1 \dots a'_n$, otherwise it is $\max_{1\leq i\leq n} |t_i - t'_i|$. Let A be a k-M alphabet, Σ' be a finite alphabet, S be a factorial extensible measurable timed language on A, C be a factorial extensible language on Σ', and α, ε be positive reals and D be a non negative integer.

Definition 10. *Similarly to Definition 5 we say that a partial function $\phi : A^* \to \Sigma'^*$ is* almost approximately injective *with precision ε and delay D if*

$$\forall n \in \mathbb{N}, w, w' \in A^n \ \forall u, u' \in A^D : \ \phi(wu) = \phi(w'u') \Rightarrow d(w, w') < \varepsilon.$$

Its almost inverse *is a multi-valued function family $\psi_n : \Sigma'^* \to A^n$ characterized by the following property: for any $w \in A^n$ it holds that $w \in \psi_n(v)$ if and only if some $u \in A^D$ yields $\phi(wu) = v$.*

Lemma 45. *Given an almost approximately injective function ϕ with precision ε and delay D, let ψ_n be its almost inverse family. Then for every v the diameter of $\psi_n(v)$ is at most ε. If the domain of ϕ is extensible, then the image of ψ_n coincides with this domain (constrained to length n).*

Definition 11. *An (S, C)-encoding of a rational rate α, precision ε and delay D is a function $\phi : S \to C$ (total on S) such that*

- *it is of rate α: i.e. $\forall w \in S, \ \lceil \alpha |\phi(w)| \rceil = |w|$;*
- *it is almost approximately injective with precision ε and delay D.*

The information inequality for this setting has the form:

$$\alpha(\mathcal{H}(S) + \log(1/\varepsilon)) \leq h(C), \tag{II3}$$

which corresponds to information contents of S equal to $\mathcal{H}(S) + \log(1/\varepsilon)$, see the formula for Kolmogorov complexity of timed words in [AD09b].

Proposition 33. *For a factorial extensible measurable timed language S and a factorial extensible discrete language C the following holds. If an (S, C)-encoding of rate α, precision ε, and some delay D exists then necessarily* (II3) *must be satisfied.*

Proof. Let ϕ be an (S, C)-encoding of rate α, precision ε and delay D, and ψ its almost inverse. By Lemma 45, for every n it holds that $S_n = \psi_n(C_{\lfloor (n+D)/\alpha \rfloor})$. This leads to an inequality on volumes

$$\mathtt{Vol}(S_n) \leq \sum_{v \in C_{\lfloor (n+D)/\alpha \rfloor}} \mathtt{Vol}(\psi_n(v)).$$

Any $\psi(v)$ has a diameter $\leq \varepsilon$ and thus is included in a cube of side ε and volume ε^n. We have:

$$\texttt{Vol}(S_n) \leq \varepsilon^n |C_{\lfloor (n+D)/\alpha \rfloor}|.$$

Thus

$$\frac{\alpha}{n} \log \texttt{Vol}(S_n) \leq \frac{\alpha}{n} \log \varepsilon^n |C_{\lfloor (n+D)/\alpha \rfloor}| = \alpha \log \varepsilon + \frac{\lfloor (n+D)/\alpha \rfloor}{n/\alpha} \frac{|C_{\lfloor (n+D)/\alpha \rfloor}|}{\lfloor (n+D)/\alpha \rfloor}.$$

Taking the limit as n tends to infinity we obtain $\alpha \mathcal{H}(S) \leq \alpha \log \varepsilon + h(C)$ and then (II3) holds. □

We strengthen a little bit (II3) to have a (partial) converse result for timed sofic languages.

Proposition 34. *For a progressive timed sofic language S satisfying with $\mathcal{H}(S) > -\infty$, there exists a function R_S such that $\lim_{x \to 0} R_S(x) = 0$ and the following holds. Whenever the entropy of a sofic discrete language C verifies the inequality $\alpha(\mathcal{H}(S) + \log(1/\varepsilon) + R_S(\varepsilon)) < h(C)$, then there exists an (S, C)-encoding of rate α, precision ε and some delay D. Moreover it can be realized by a "real-time transducer" sketched below in the proof.*

Sketch. Let S be a timed sofic language. For $\varepsilon > 0$, let S_ε^+ be its ε-discretized over-approximation given by Lemma 44. We define

$$R_S(\varepsilon) = h(S_\varepsilon^+) - \mathcal{H}(S) - \log(1/\varepsilon),$$

it satisfies the required condition: $R_S(\varepsilon) = o(1)$ (see Lemma 44). Let C be a sofic discrete language such that

$$\alpha(\mathcal{H}(S) + \log(1/\varepsilon) + R_S(\varepsilon)) < h(C),$$

we prove that an (S, C)-encoding of rate α, precision ε and some delay D exists. Lemma 44 gives us

$$S \subseteq \mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+) \text{ and } \alpha h(S_\varepsilon^+) = \alpha(\mathcal{H}(S) + \log(1/\varepsilon) + R_S(\varepsilon)) < h(C).$$

Thus by Proposition 32 an (S_ε^+, C)-encoding of rate α and some delay D exists and can be realized by a finite-state transducer τ_ε of rate α and delay D. If we replace for each transition its input label $(a, k\varepsilon)$ by the label a and the guard $t \in [k\varepsilon, (k+1)\varepsilon]$, we obtain a real-time transducer τ with input $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$ whose output is in C. The injectivity of τ_ε ensures that τ realizes an approximately injective function with precision ε. □

6.2.2 Timed source, timed channel, exact transmission

Another natural setting is when a timed message is transmitted via a timed channel. In this case, the coding can be exact (injective). For the moment we consider length-preserving transmission (see section 6.2.4 for faster and slower transmission).

Let A, A' be a k-M and a k'-M' alphabet, S and C factorial extensible measurable timed languages on these alphabets, and $D \in \mathbb{N}$.

Let ℓ and σ be positive rationals. A function $f : A'^n \to A^*$ is said to be ℓ-Lipshitz whenever for all x, y in its domain, $d(f(x), f(x')) \leq \ell\, d(x, x')$. We call a function σ-piecewise ℓ-Lipshitz if its restriction to each cube of the standard σ-grid on A'^n is ℓ-Lipshitz.

We can now state the definition of an (S, C)-encoding:

Definition 12. *An (S, C)-encoding with delay D (and step σ) is a function $\phi : S \to C$ such that*

- *it is length preserving: $|\phi(w)| = |w|$,*
- *it is almost injective (with delay D),*
- *no time scaling: the almost inverse ψ_n are σ-piecewise 1-Lipshitz.*

The last condition rules out a possible cheating when for instance all the time delays are divided by 1000 before transmission over the channel. We will come back to this issue in section 6.2.3.

The information inequality in this setting takes a very simple form:

$$\mathcal{H}(S) \leq \mathcal{H}(C). \tag{II4}$$

The necessary condition for existence of a coding has a standard form (for technical reasons we require the channel to be sofic):

Proposition 35. *If for a factorial extensible measurable timed language S and a progressive timed sofic language C an (S, C)-encoding of delay D exists, then* (II4) *holds.*

Proof. We consider first the most interesting case when $\mathcal{H}(C) > -\infty$. We will prove that for any $\zeta > 0$ the inequality $\mathcal{H}(S) \leq \mathcal{H}(C) + \zeta$ holds. Suppose ϕ is an (S, C)-encoding of delay D (and step σ), and ψ its almost inverse. By Lemma 43, for any natural n we have $S_n \subset \psi_n(C_{n+D})$.

Since C is sofic, Lemma 44 applies, and for a fixed ϵ, each C_n can be covered by C_n^+, a union of $K_{n,\varepsilon}$ cubes of size ε with $\mathcal{H}(C) + \log \frac{1}{\varepsilon} = \lim_{n\to\infty} \frac{\log K_{n,\varepsilon}}{n} + o(1)$. We choose ε dividing σ and small enough such that

$$\mathcal{H}(C) + \log \frac{1}{\varepsilon} > \lim_{n\to\infty} \frac{\log K_{n,\varepsilon}}{n} - \zeta. \tag{6.1}$$

Thus we have $S_n \subset \psi_n(C^+_{n+D})$, and, passing to volumes we get

$$\mathtt{Vol}(S_n) \leq \mathtt{Vol}(\psi_n(C^+_{n+D})) \leq K_{n+D,\varepsilon}\varepsilon^n, \tag{6.2}$$

indeed, since ψ_n is 1-Lipshitz on each ε-cube, ψ_n-image of each such cube has a diameter $\leq \varepsilon$ and thus a volume $\leq \varepsilon^n$. Passing to logarithms, dividing by n and taking the limit as $n \to \infty$ in (6.2) we get

$$\mathcal{H}(S) \leq \lim_{n\to\infty} \frac{\log K_{n+D,\varepsilon}}{n + D} + \log \varepsilon,$$

and applying inequality (6.1) we obtain

$$\mathcal{H}(S) \leq \mathcal{H}(C) + \log \frac{1}{\varepsilon} + \zeta + \log \varepsilon = \mathcal{H}(C) + \zeta,$$

which concludes the proof for the case when $\mathcal{H}(C) > -\infty$. The remaining case $\mathcal{H}(C) = -\infty$ is a simple corollary of the previous one. □

As usual, when both timed languages S and C are sofic and the information inequality (II4) strict, the converse holds.

Proposition 36. *If for progressive timed sofic languages S and C it holds that $H(S) < H(C)$, then there exists an (S, C)-encoding (with some delay D). Moreover it can be realized by a "real-time transducer" described below in the proof.*

Sketch. Let S and C be timed sofic languages whose entropies verify $-\infty < \mathcal{H}(S) < \mathcal{H}(C)$. We prove that an (S, C)-encoding with some delay D exists. Let C_ε^- and S_ε^+ be as in Lemma 44, i.e. such that

$$S \subseteq \mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+);\ \mathcal{B}_\varepsilon^{NE}(C_\varepsilon^-) \subseteq C;$$
$$\mathcal{H}(S) + \log \frac{1}{\varepsilon} = h(S_\varepsilon^+) + o(1);\ \mathcal{H}(C) + \log \frac{1}{\varepsilon} = h(C_\varepsilon^-) + o(1).$$

The discretization step ε can be chosen small enough such that $h(S_\varepsilon^+) < h(C_\varepsilon^-)$. Thus by Theorem 31 a finite-state $(S_\varepsilon^+, C_\varepsilon^-)$-encoder τ_ε exists.

We replace each transition δ_ϵ of τ_ϵ with input label $(a, k\varepsilon)$ and output label $(b, l\varepsilon)$ by a transition δ with input label a, guards $t \in [k\varepsilon, (k+1)\varepsilon]$, output label b and increment/decrement $c(\delta) = (l-k)\varepsilon$. We obtain what we call a *real-time transducer* τ. Its input language is $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$ and its output language is included in $\mathcal{B}_\varepsilon^{NE}(C_\varepsilon^-) \subseteq C$. The encoding is performed as follows: a timed word $(t_1, a_1) \dots (t_n, a_n)$ is in the input language if there is a path $\delta_1 \dots \delta_n$ such that $\mathcal{I}(\delta_i) = a_i$ and t_i satisfies the guard of δ_i: $t_i \in [k\varepsilon, (k+1)\varepsilon]$; the output timed word is in this case $(t'_1, b_1) \dots (t'_n, b_n)$ with $t'_i = t_i + c(\delta_i)$, $b_i = \mathcal{O}(\delta_i)$.

The collection of cubes $\mathcal{B}_\varepsilon^{NE}(w)$, $w \in S_\varepsilon^+$ (resp. $w \in C_\varepsilon^-$) forms a partition of timed languages $\mathcal{B}_\varepsilon^{NE}(S_\varepsilon^+)$ (resp. $\mathcal{B}_\varepsilon^{NE}(C_\varepsilon^-)$), they are cubes of the standard σ-grid with the step $\sigma = \epsilon$. Transducer τ only translates cubes. Translations are 1-Lipshitz and thus the last condition of an (S, C)-encoding holds.

The remaining degenerate case when $-\infty = \mathcal{H}(S) < \mathcal{H}(C)$ is an easy corollary of the non-degenerate one. □

The following example illustrates the construction of the transducer. Let the source timed language be $S = ([0, 1] \times \{e, f\})^*$ and the channel timed language C be recognized by the automaton on Figure 6.2. We have seen that the entropy $\mathcal{H}(C)$ is at least $0.5 \log 6.5$ and thus $\mathcal{H}(C) > \mathcal{H}(S) = \log 2$. By Proposition 36 an (S, C)-encoding exists. To realize this encoding we take $\varepsilon = 1$. There are four cubes included in the language $C_2(q)$: $([0, 1] \times [0, 1] \times \{ac\}$, $[0, 1] \times [0, 2] \times \{ac\}$, $[1, 0] \times [0, 1] \times \{ac\}$, $[0, 1] \times [0, 1] \times \{bd\})$ while the cubes to encode

δ^-	δ^+	$\mathcal{I}(\delta)$	$\mathfrak{g}(\delta)$	$\mathcal{O}(\delta)$	$c(\delta)$
q	p_0	e	$[0,1]$	a	0
q	p_1	f	$[0,1]$	a	1
q'	p'_0	e	$[0,1]$	a	0
q'	r	f	$[0,1]$	b	1
p_0	q	e	$[0,1]$	c	0
p_0	q'	f	$[0,1]$	c	0
p'_0	q	e	$[0,1]$	c	1
p'_0	q'	f	$[0,1]$	c	1
p_1	q	e	$[0,1]$	c	0
p_1	q'	f	$[0,1]$	c	0
r	q	e	$[0,1]$	d	0
r	q'	f	$[0,1]$	d	0

Table 6.1: The coding transducer

(language S_2) are $[0,1]\times\{ee\}$, $[0,1]\times\{ef\}$, $[0,1]\times\{fe\}$, $[0,1]\times\{ff\}$. The transducer will repeatedly map four "input cubes" to four "output cubes".

We build an automaton for discrete words C^-_ε (as in Lemma 44, such words correspond to "output cubes") in Figure 6.3. Then, as usual in coding, we first split the state p_0 and then the state q (each in two copies) to obtain an automaton with constant outdegree 2 (Figure 6.4). This automaton accepts the same language C^-_ε, and can be transformed to the desired transducer just by adding input letters and increment/decrement to its transition. The transitions of the transducer are given in Table 6.1.

6.2.3 A variant: scaling allowed

In some situations, timed data can be scaled for coding, which leads to a new term in the information inequality. Let $\lambda > 0$ be a rational bound on scaling factor. We modify Definition 12 by replacing 1-Lipshitz by $1/\lambda$-Lipshitz.

Definition 13. *An (S,C)-encoding with delay D, scaling λ and step σ is a function $\phi : S \to C$ such that*

- *it is length preserving: $|\phi(w)| = |w|$,*
- *it is almost injective (with delay D),*
- *it has scaling at most λ: the almost inverse ψ_n are σ-piecewise $1/\lambda$-Lipshitz.*

The information inequality for this case becomes:

$$\mathcal{H}(S) \le \mathcal{H}(C) + \log\lambda. \tag{II5}$$

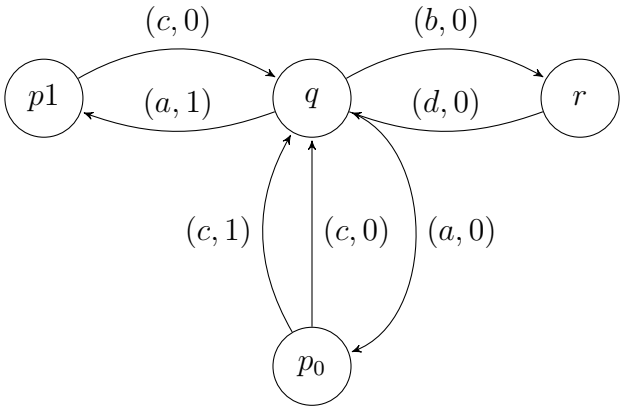

Figure 6.3: $\mathcal{A}_\varepsilon^-$: an automaton recognizing C_ε^- with $\varepsilon = 1$.

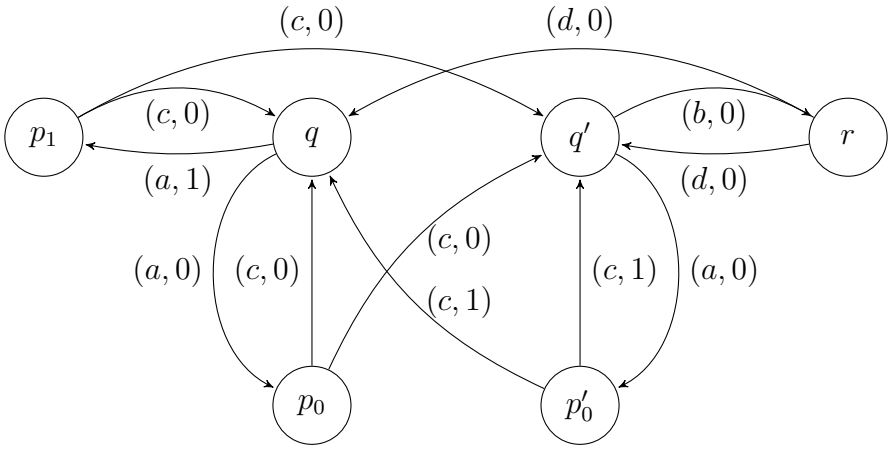

Figure 6.4: The split form of $\mathcal{A}_\varepsilon^-$.

The problem of coding with scaling can be easily reduced to the one considered in the previous section. Indeed, for a timed language C let λC be the same language with all times multiplied by λ (the entropy of this language is $\mathcal{H}(\lambda C) = \mathcal{H}(C) + \log \lambda$). A function ϕ is an (S, C)-encoding with scaling λ if and only if the "λ-scaled" function $\lambda\phi$ is an $(S, \lambda C)$-encoding without scaling. Using this reduction, the results below are corollaries of Proposition 35-36.

Proposition 37. *If for factorial extensible measurable timed language S and progressive timed sofic language C an (S, C)-encoding with scaling λ and delay D exists then* (II5) *holds.*

Proposition 38. *If for progressive timed sofic languages S and C (with $\mathcal{H}(S) > -\infty$) the strict version of* (II5) *holds, then there exists an (S, C)-encoding with scaling λ (with some delay D). Moreover it can be realized by a "real-time transducer".*

6.2.4 A speedup and a slowdown lead to a collapse

For untimed channels, transmission with some rate $\alpha \neq 1$leads to the factor α in information inequalities (II2), (II3). Unfortunately, for timed channels this does not work: any rate $\alpha \neq 1$ leads to a collapse of the previous theory.

Definition 14. *An (S, C)-encoding with rational rate α, delay D and step σ is a function $\phi : S \to C$ such that*

- *its rate is α, i.e. $\forall w \in A^*,\ \lceil \alpha |\phi(w)| \rceil = |w|$;*
- *it is almost injective (with delay D);*
- *no time scaling: its almost inverse ψ is σ-piecewise 1-Lipschitz.*

For $\alpha > 1$ no coding is possible, and for $\alpha < 1$ it always exists. More precisely, the two following propositions hold.

Proposition 39. *For factorial measurable timed languages S and C if $\mathcal{H}(S) > -\infty$ and $\alpha > 1$, then no (S, C)-encoding with rate α exists.*

Sketch. The proof follows the same lines as that of Proposition 35. Suppose ϕ is such an (S, C)-encoding, and ψ its almost inverse. By Lemma 43, for any natural n we have $S_n \subset \psi_n(C_{\lfloor (n+D)/\alpha \rfloor})$. Each C_n can be covered by C_n^+, a union of $K_{n,\varepsilon}$ cubes of size ε satisfying inequality (6.1) We have $S_n \subset \psi_n(C^+_{\lfloor (n+D)/\alpha \rfloor})$, and, passing to volumes we get $\mathtt{Vol}(S_n) \leq \mathtt{Vol}(\psi_n(C^+_{\lfloor (n+D)/\alpha \rfloor})) \leq K_{\lfloor (n+D)/\alpha \rfloor, \varepsilon} \varepsilon^n$. Passing to logarithms, dividing by n and taking the limit as $n \to \infty$ we get

$$\mathcal{H}(S) \leq \alpha^{-1} \lim_{n \to \infty} \frac{\log K_{\lfloor (n+D)/\alpha \rfloor, \varepsilon}}{\lfloor (n+D)/\alpha \rfloor} + \log \varepsilon,$$

and applying inequality (6.1) we obtain

$$\alpha \mathcal{H}(S) \leq \mathcal{H}(C) + \log(1/\varepsilon) + \zeta + \alpha \log \varepsilon = \mathcal{H}(C) + \zeta - (\alpha - 1) \log(1/\varepsilon).$$

Choosing ε small enough makes the inequality wrong. This contradiction concludes the proof. □

Proposition 40. *For timed sofic languages S and C (with $\mathcal{H}(C) > -\infty$) and any $\alpha < 1$ there exists an (S, C)-encoding with rate α (and some delay D). Moreover it can be realized by a kind of timed transducer.*

The construction is non-trivial and uses spare time durations in the channel message to transmit discrete information.

6.3 Conclusion and perspectives

In the previous section, we have established several results on timed channel coding following the standard scheme: a setting of information transmission – information inequality – coding existence theorem – synthesis of an encoder/decoder. We believe that this approach can be applied to various situations of data transmission (and compression). We also consider it as a justification of the previous research on entropy of timed languages [AD09a, AD09b, AD10]. In this concluding section, we explain some of our choices and immediate perspectives of this approach.

The time is not preserved. In the central Definition 12 and Proposition 35,36, we consider codings of timed words that preserve the number of events, and not their duration. This choice is compatible to the general idea of dealing with data words (in our case, sequences of letters and real numbers), and less so with the standard timed paradigm. We use again the example of Figure 6.2 to illustrate this feature. For $n \in \mathbb{N}$, the timed word $w = [(0.5, e)(0.5, f)]^n$ is encoded to the timed word $w' = [(0.5, a)(1.5, c)]^n$, both have $2n$ events. However, the duration of w is $(0.5 + 0.5)n = n$, while the duration of w' is $(0.5 + 1.5)n = 2n$.

Other settings to explore. It would be still interesting to explore coding functions preserving durations. On the theoretical side, a more detailed analysis for transmission speeds different from 1, as in section 6.2.4 would probably lead to information inequalities instead of a collapse. Many other settings of transmission of information could be practically relevant: approximated transmission of a timed source on a timed channel; coding of a discrete source on a timed channel; coding using transducers of a fixed precision; coding of other kinds of data languages. On the other hand, more physical models of timed and data channels would be interesting to study, one can think of a discrete channel with a fixed baud rate coupled with an analog channel with a bounded frequency bandwidth. Finally, some special codes, such as sliding-window, error-correcting etc., should be explored for timed and data languages.

What is a timed transducer? In the classical theory of constrained channel coding, several kinds of transducers are used for encoding/decoding, such as those leading to a sliding-block window decoding. Here, we have realized our codes by using some very restricted ad hoc timed transducers. They behave like timed (in our case, real-time) automata on the input, and "print" letters and real numbers on the output; we believe that this is the correct approach. However, the right definition of a natural class of timed transducers adequate to coding remains an open question. As a preliminary definition, we suggest timed automata that output, on each transition, a letter and a real number (which is an affine combination of clock values). While reading a timed word, such a transducer would output another timed word. We illustrate this informal definition with the example of mutually inverse transducers τ_1 and τ_2 (encoder and decoder) on Figure 6.5. Let us consider a run of the transducer τ_2 on

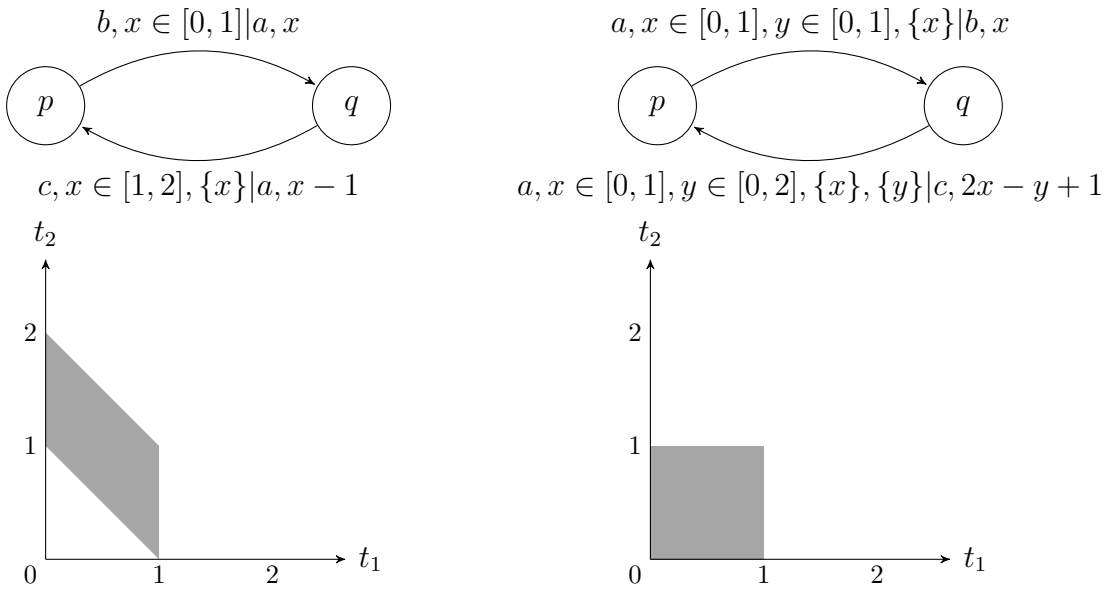

Figure 6.5: Top: transducers $\tau_1 : S \to C$ and $\tau_2 : C \to S$. Bottom: languages S_2 and C_2.

the timed word $(t_1, b)(t_2, c)\ldots$ We start from q with $x = 0, y = 0$, after reading (t_1, b) the value of x and y is t_1, we fire the transition, the output is (t_1, a), we pass in q where we read (t_2, c), the value of x is t_2 and the value of y is $t_1 + t_2$, we fire the transition, the output is $(2t_2 - (t_1 + t_2) + 1, a) = (t_2 - t_1 + 1, a)$, we pass in p etc.

For τ_1, the input language of timed words of length 2 starting from p is $S_2(p) = \{(t_1, b)(t_2, c) \mid t_1 \in [0,1],\ t_1 + t_2 \in [1,2]\}$, while for the second transducer it is $C_2(p) = \{(t_1, a)(t_2, a) \mid t_1 \in [0,1],\ t_2 \in [0,1]\}$. These two languages are depicted in Figure 6.5, they have the same volume. It would be impossible to realize this kind of encoding using "rectangular" transducers as in section 6.2.

How to improve code synthesis? The encoders in section 6.2 are not completely satisfactory: they use non-integer guards even when the source and the target language are defined using integer timed automata. We believe that this issue can be avoided using a broader class of transducers as suggested just above.

Applications. In practice, when transmitting (or storing) information containing discrete events and real-valued data, all the information is first converted to the digital form and next encoded for transmission or storage. Our paradigm combines both steps and, in principle, provides better bounds and codes. However, more research is needed to come up with useful practical codes.

Chapter 7

Generating functions of timed languages

Abstract of the chapter

In order to study precisely the growth of timed languages, we associate to such a language a generating function. These functions (tightly related to volume and entropy of timed languages) satisfy compositionality properties and, for deterministic regular timed languages, can be characterized by integral equations. We provide procedures for closed-form computation of generating functions for some classes of timed automata and regular expressions.

Chapter structure

In section 7.1 we introduce a formalism (inspired by [BP02]) for timed and clock languages, introduce volume functions of such languages, and investigate the properties of these functions. In section 7.2 we introduce generating functions of timed languages and investigate their general properties. In section 7.3 we explain how to compute generating functions for several subclasses of timed automata. We summarize the contributions and discuss the directions of future work in section 7.4.

7.1 Preliminaries

7.1.1 Clock languages and timed languages

In this chapter, we study timed languages (mostly regular) using an approach based on clock languages introduced in [BP02]. We present this approach in a slightly different form along with a multi-stage semantics. The general idea is as follows: we are interested in timed languages. Timed languages are obtained as projections of clock languages. Clock languages are homomorphic images of discrete "triplet languages". Triplet languages, in turn, can be generated by automata or regular expressions. Below we define formally all these notions and illustrate them on a running example.

An alphabet of *timed events* is the product $\mathbb{R}^+ \times \Sigma$ where Σ is a finite alphabet. The meaning of a timed event (t, a) is that t is the time delay before the event a. A *timed word* is a sequence of timed events and a *timed language* is just a set of timed words.

Inspired by [BP02], we enrich timed words and languages with d-dimensional clock vectors. As usual, a *clock* is a variable which takes values in $\mathbb{R}^+$ which is (in our setting) bounded by a positive integer M. A *clock word* is a tuple whose components are a starting clock vector $\vec{x}$, a timed word $\mathbf{w}$ and a final clock vector clock vector $\vec{x}$, it is denoted by $\vec{x} \xrightarrow{\mathbf{w}} \vec{y}$. Two clock words $\vec{x} \xrightarrow{\mathbf{w}} \vec{y}$ and $\vec{x}' \xrightarrow{\mathbf{w}'} \vec{y}'$ are said to be compatible if $\vec{y} = \vec{x}'$, in this case we define their product by $\vec{x} \xrightarrow{\mathbf{w}} \vec{y} \cdot \vec{x}' \xrightarrow{\mathbf{w}'} \vec{y}' = \vec{x} \xrightarrow{\mathbf{w}\mathbf{w}'} \vec{y}'$. A *clock language* is a set of clock words. The product of two clock languages $\mathcal{L}$ and $\mathcal{L}'$ is

$$\mathcal{L} \cdot \mathcal{L}' = \{c \cdot c' \mid c \in \mathcal{L},\ c' \in \mathcal{L}',\ c \text{ and } c' \text{ compatible}\}. \tag{7.1}$$

The neutral element $\mathcal{E}$ is $\{\vec{x} \xrightarrow{\epsilon} \vec{x} \mid \vec{x} \in \mathbb{R}^d\}$ and the Kleene star of a language $\mathcal{L}$ is as usual $\mathcal{L}^* = \bigcup_k \mathcal{L}^k$ with $\mathcal{L}^0 = \mathcal{E}$.

A clock language $\mathcal{L}$ is said to be *deterministic* whenever for each clock word the final clock vector is uniquely determined by the starting clock vector and the timed word, in other words there exists a function $\sigma_{\mathcal{L}} : \mathbb{R}^d \times (\mathbb{R}^+ \times \Sigma)^* \to \mathbb{R}^d$ such that for any clock word $\vec{x} \xrightarrow{\mathbf{w}} \vec{y}$ of $\mathcal{L}$, we have that $\vec{y} = \sigma_{\mathcal{L}}(\vec{x}, \mathbf{w})$. In the following, we work with deterministic clock languages[1].

To a clock language we associate its timed projections. Given $\mathcal{L}$, we define $\mathcal{L}(\vec{x}, \vec{x}')$ as the timed language leading from $\vec{x}$ to an element lower than $\vec{x}'$:

$$\mathcal{L}(\vec{x}, \vec{x}') = \{w \mid \exists \vec{y} \in \mathbb{R}^d,\ \vec{x} \xrightarrow{\mathbf{w}} \vec{y} \in \mathcal{L} \text{ and } \vec{y} \leq \vec{x}'\}$$

where $\vec{y} \leq \vec{x}'$ means that for every $i \in \{1, \ldots, d\}$, $y_i \leq x'_i$. We also define the timed language

$$\mathcal{L}(\vec{x}) = \{w \mid \exists \vec{y} \in \mathbb{R}^d,\ \vec{x} \xrightarrow{\mathbf{w}} \vec{y} \in \mathcal{L}\}$$

as the language starting from $\vec{x}$. Note that $\mathcal{L}(\vec{x}) = \mathcal{L}(\vec{x}, \vec{M})$ where $\vec{M} = (M, \ldots, M)$ is the greatest clock vector possible.

7.1.2 From timed automata to triplet, clock and timed languages

In this section, following [BP02], we give a convenient representation of timed automata (such as those on Figure 7.1) and their languages.

Triplets and timed automata. We define *timed automata* as finite automata over the finite alphabet $\mathcal{T}$ of *triplets*. These triplets are tuples $\langle a, \mathfrak{g}, \mathfrak{r} \rangle$ with: a a letter in Σ; $\mathfrak{g}$ a conjunction of constraints $x_i \bowtie c$ ($i \in \{1..d\}$, $c \in \{0..M\}$, $\bowtie \in \{<, >, \leq, \geq\}$) called *guard* and $\mathfrak{r} \subseteq \{1..d\}$ a set of indices of clocks to be reset. We suppose moreover that guards are such that all the clocks remain bounded by M.

[1]Such are clock languages associated to deterministic timed automata. However, a product of two deterministic clock languages can be non-deterministic, and we will explicitly rule out this situation.

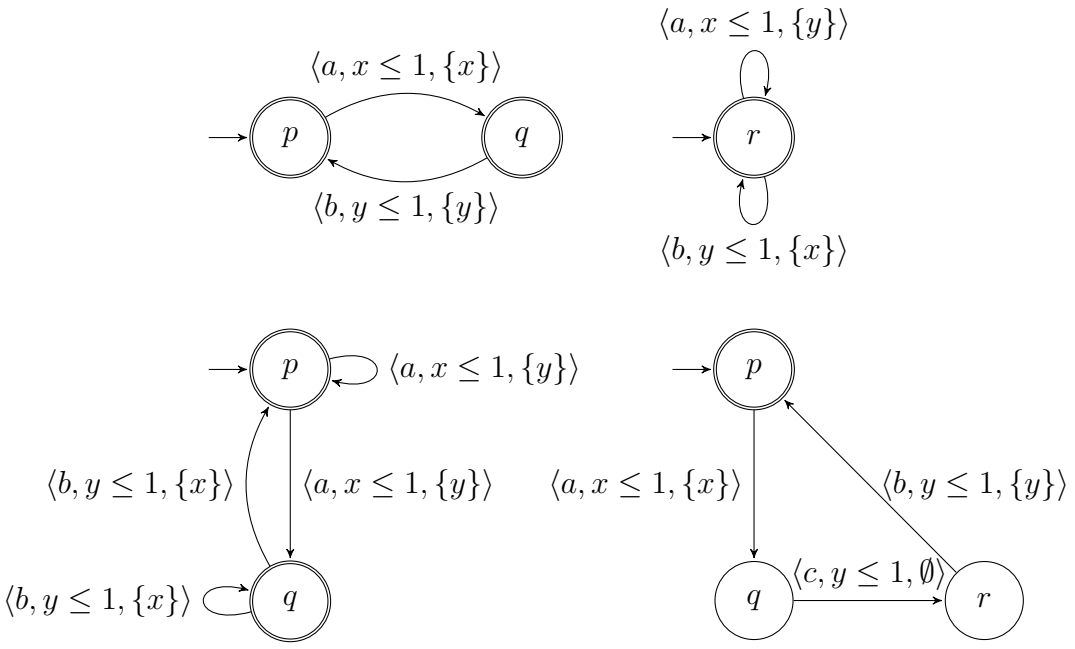

Figure 7.1: Timed automata. First line: $\mathcal{A}_1$, $\mathcal{A}_2$; second line: $\mathcal{A}_3$, $\mathcal{A}_4$.

Clock semantics of triplets. Informally, a triplet $\langle a, \mathfrak{g}, \mathfrak{r}\rangle$ corresponds to the following behaviours: starting from some clock vector $\vec{x}$ let some time t elapse (all the clocks advance by t), check that $\mathfrak{g}$ is satisfied, emit a and update the clocks according to $\mathfrak{r}$. Formally, the clock language of this triplet is $\mathcal{L}(\langle a, \mathfrak{g}, \mathfrak{r}\rangle) = \{\vec{x} \xrightarrow{(t,a)} \mathfrak{r}(\vec{x}+t) \mid \vec{x}+t \models \mathfrak{g}\}$. Here, for a clock vector $\vec{x} = (x_1, \ldots, x_d)$, we denote by $\vec{x}+t$ the vector $(x_1+t, \ldots, x_d+t)$. Clock vectors are updated as follows: $\mathfrak{r}(y_1, \ldots, y_d) = (y'_1, \ldots, y'_d)$ with $y'_i = 0$ if $i \in \mathfrak{r}$ and $y'_i = y_i$ otherwise.

This definition can be extended to all triplet words by: $\mathcal{L}(\epsilon) = \mathcal{E}$ and $\mathcal{L}(\pi_1 \ldots \pi_n) = \mathcal{L}(\pi_1) \ldots \mathcal{L}(\pi_n)$ (using the product of clock languages as defined in (8.5)). Finally for a language $L \subseteq \mathcal{T}^*$, we define $\mathcal{L}(L) = \{\mathcal{L}(\pi) \mid \pi \in L\}$. In fact, $\mathcal{L}$ is a morphism between the two Kleene algebras of triplet and clock languages.

Timed automata and their languages. Given a *timed automaton* $\mathcal{A}$, its *discrete semantics* L is the language of triplet words accepted by $\mathcal{A}$ seen as a finite automaton over $\mathcal{T}$; its *clock semantics* is $\mathcal{L}_\mathcal{A} =_{\text{def}} \mathcal{L}(L)$ and its *timed semantics* is $\mathbb{L}_\mathcal{A} =_{\text{def}} \mathcal{L}_\mathcal{A}(\mathbf{0})$. The timed automata considered in this chapter are assumed to be deterministic in the sense of [AD94], i.e. outgoing transitions from the same state with the same letter must have pairwise incompatible guards. Clock languages of deterministic automata are also deterministic.

A timed regular expression is defined as expression over the finite alphabet $\mathcal{T}$. Its discrete, clock and timed semantics are defined similarly to the automata. This multi-stage timed semantics is equivalent to the usual semantics of timed automata and timed regular expressions.

Example 11 (Running example of the Chapter)**.** *Automata $\mathcal{A}_2$ and $\mathcal{A}_3$ on Figure 7.1 have the same discrete semantics*[2] *which is captured by a regular expression:*

$$(\langle a, x \leq 1, \{y\}\rangle + \langle b, y \leq 1, \{x\}\rangle)^*.$$

An example of a clock word recognized by the automaton is $(0.5, 0.8) \xrightarrow{(0.3,a)(0.1,a)(0.9,b)} (0, 0.9)$. *The timed language recognized* $\mathbb{L}_{\mathcal{A}_2} = \mathbb{L}_{\mathcal{A}_3}$ *is:*

$$\{(t_1, a)\cdots(t_{n_1}, a)(t_{n_1+1}, b)\cdots(t_{n_2}, b)(t_{n_2+1}, a)\cdots(t_{n_3}, a)\cdots \mid \forall j \geq 0, \sum_{i=n_j+1}^{n_{j+1}} t_i \leq 1\}$$

with $n_0 = 0$ *and possibly* $n_1 = 0$.

Matrix notation. A convenient way to see automata is the matrix form. A timed automaton $\mathcal{A}$ over a set of control states Q and an alphabet of transitions $\mathcal{T}$ is uniquely described by three ingredients:

- a $Q \times Q$-matrix $\mathbb{T}$ whose element $\mathbb{T}_{qq'}$ is the set of triplets labelling transitions from q to q';
- a row vector $\mathbf{I}$ describing initial states: for each control state p, its element $I_p = \{\epsilon\}$ iff p is initial, and $\emptyset$ otherwise;
- a column vector $\mathbf{F}$ describing final states: for each control state q, its element $F_q = \{\epsilon\}$ iff q is final, and $\emptyset$ otherwise.

The coefficient $(\mathbb{T}^n)_{qq'}$ of $\mathbb{T}^n$ contains the language of all the triplet words of length n from q to q'. The q^{th} coordinates of the column vector $\mathbb{T}^n\mathbf{F}$ contains the language recognized from state q and $\mathbf{I}\mathbb{T}^n\mathbf{F}$ contains the language of triplet words of length n recognized by $\mathcal{A}$. For instance the matrices for $\mathcal{A}_3$ are:

$$\mathbf{I} = \begin{pmatrix} \{\epsilon\} & \emptyset \end{pmatrix};\ \mathbb{T} = \begin{pmatrix} \{\langle a, x \leq 1, \{y\}\rangle\} & \{\langle b, y \leq 1, \{x\}\rangle\} \\ \{\langle a, x \leq 1, \{y\}\rangle\} & \{\langle b, y \leq 1, \{x\}\rangle\} \end{pmatrix};\ \mathbf{F} = \begin{pmatrix} \{\epsilon\} \\ \{\epsilon\} \end{pmatrix}.$$

7.1.3 Volume(s) of timed and clock languages

Measurable timed languages and clock languages.

A timed language $\mathbb{L}$ is *measurable* if, for any word $w \in \Sigma^*$, the projection $\mathbb{L}_w = \{\vec{t} \in \mathbb{R}^{|w|} \mid (\vec{t}, w) \in \mathbb{L}\}$ is a Lebesgue-measurable subset of $\mathbb{R}^{|w|}$. A clock language $\mathcal{L}$ is measurable if it is deterministic and for every $w \in \Sigma^*$, $\sigma_{\mathcal{L}}(\cdot, (\cdot, w))$ is a Lebesgue-measurable function of $\mathbb{R}^d \times \mathbb{R}^{|w|} \to \mathbb{R}^d$. We remark that timed languages and deterministic clock languages obtained from a triplet word are measurable because their timed projections are polytopes.

[2]The difference will appear in section 7.3.1 since $\mathcal{A}_3$ is $1\frac{3}{4}$-clocks and $\mathcal{A}_2$ is not.

Volumes of a timed language [AD09a].

The sequence of volumes $(V_n(\mathbb{L}))_{n\in\mathbb{N}}$ associated to a measurable timed language is $V_n(\mathbb{L}) = \sum_{w\in\Sigma^n} \mathtt{Vol}(\mathbb{L}_w)$, where $\mathtt{Vol}$ is the hyper-volume (i.e. Lebesgue measure) in $\mathbb{R}^n$. For dimension 0 we define $V_0(\mathbb{L}) = 1$ if $\epsilon \in \mathbb{L}$, and $V_0(\mathbb{L}) = 0$ otherwise.

Now, for a clock language $\mathcal{L}$ and a word $w \in \Sigma^*$ of length $n \geq 0$, we define the clock language $\mathcal{L}(w) = \{\vec{x} \xrightarrow{(\vec{t},v)} \vec{x}' \mid v = w\}$.

Volumes constrained by initial and final clock vectors.

regular timed languages considered below come from clock languages (which themselves come from triplet languages). The information about clock vectors is crucial to compute the volume of timed languages in a compositional manner.

Thus we define parametric volumes depending on initial and final clock vectors as follows

$$V_n^2(\vec{x}, \vec{x}') = V_n(\mathcal{L}(\vec{x}, \vec{x}')).$$

We call this function the *cumulative volume function* (CVF)[3] of $\mathcal{L}$. We also allow the following notations: for a clock language $\mathcal{L}$ and a discrete events word w,

$$V^2_{\mathcal{L}(w)}(\vec{x}, \vec{x}') = V^2_{|w|}(\mathcal{L}(w)(\vec{x}, \vec{x}'));$$

and for a triplets word π,

$$V_\pi^2(\vec{x}, \vec{x}') = V_{|\pi|}(\mathcal{L}(\pi)(\vec{x}, \vec{x}')).$$

The notion of parametric volumes can also be applied to the clock language constrained only by starting clock vector $\mathcal{L}(\vec{x})$:

$$V_n^1(\vec{x}) = V_n(\mathcal{L}(\vec{x})).$$

Clearly $V_n^1(\vec{x}) = V_n^2(\vec{x}, \vec{M})$.

CVFs for a triplet word.

According to the following result, a CVF is easy to compute for a triplet word, and hence for a finite triplet language.

Proposition 41. *For a triplet word π the CVF V_π^2 is piecewise polynomial with rational coefficients of degree $\leq |\pi|$. The pieces are polytopes, and an expression of this function is computable.*

Sketch of proof. The clock language $\mathcal{L}(\pi)$ is a polytope in $\mathbb{R}^d \times \mathbb{R}^n \times \mathbb{R}^d$. Its timed section $\mathcal{L}(\pi)(\vec{x}, \vec{x}')$ is a polytope in $\mathbb{R}^n$ whose coefficients are linear functions of $\vec{x}$ and $\vec{x}'$. Volumes of such polytopes are computable and have the required form. □

[3] similarly to cumulative distribution functions in probability theory.

Composing CVFs.

In order to define a composition for CVF corresponding to the concatenation of triplet words and languages, we proceed as follows. We define composition of two functions of $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as:

$$V_1^2 \star V_2^2(\vec{x}, \vec{x}') = \int_{\vec{y}} V_2^2(\vec{y}, \vec{x}') V_1^2(\vec{x}, d\vec{y}),$$

where the integral is the Lebesgue-Stieltjes integral[4]. We also define

$$V_1^2 \star v(\vec{x}) = \int_{\vec{y}} v(\vec{y}) V_1^2(\vec{x}, d\vec{y}),$$

when v is defined on $\mathbb{R}^d$. Then we can state the key lemma (to transpose concatenation of words to the CVFs world):

Proposition 42. *For any measurable clock languages $\mathcal{L}_1$ and $\mathcal{L}_2$ and discrete words w_1 and w_2, $V^2_{\mathcal{L}_1(w_1)} \star V^2_{\mathcal{L}_2(w_2)}$ is well defined and satisfies: $V^2_{\mathcal{L}_1(w_1)} \star V^2_{\mathcal{L}_2(w_2)} = V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)}$.*

Proof. First recall that

$$\begin{aligned} V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)}(\vec{x}, \vec{x}') &= \mathtt{Vol}\{\vec{t} \mid \sigma_{\mathcal{L}_1\cdot\mathcal{L}_2}(\vec{x}, (\vec{t}, w_1 w_2)) \leq \vec{x}'\} \\ &= \int_{\vec{t}} \mathbf{1}_{\sigma_{\mathcal{L}_1\cdot\mathcal{L}_2}(\vec{x}, (\vec{t}, w_1 w_2)) \leq \vec{x}'} d\vec{t}, \end{aligned}$$

which gives

$$V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)}(\vec{x}, \vec{x}') = \int_{\vec{t_1}, \vec{t_2}} \mathbf{1}_{\sigma_{\mathcal{L}_2}(\sigma_{\mathcal{L}_1}(\vec{x}, (\vec{t_1}, w_1)), (\vec{t_2}, w_2)) \leq \vec{x}'} d\vec{t_1} d\vec{t_2}.$$

By Fubini's theorem this can be rewritten as

$$V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)}(\vec{x}, \vec{x}') = \int_{\vec{t_1}} \left(\int_{\vec{t_2}} \mathbf{1}_{\sigma_{\mathcal{L}_2}(\sigma_{\mathcal{L}_1}(\vec{x}, (\vec{t_1}, w_1)), (\vec{t_2}, w_2)) \leq \vec{x}'} d\vec{t_2} \right) d\vec{t_1}.$$

Applying again the formula for $V^2_{\mathcal{L}_2(w_2)}$, we get that

$$\begin{aligned} V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)}(\vec{x}, \vec{x}') &= \int_{\vec{t_1}} V^2_{\mathcal{L}_2(w_2)}(\sigma_{\mathcal{L}_1}(\vec{x}, (\vec{t_1}, w_1)), \vec{x}') d\vec{t_1} \\ &= \int_{\vec{x_1}} V^2_{\mathcal{L}_2(w_2)}(\vec{x_1}, \vec{x}')\, V^2_{\mathcal{L}_1(w_1)}(\vec{x}, d\vec{x_1}), \end{aligned}$$

as required. The last change of variables can be justified as follows. It has the form:

$$\int_{\vec{t_1}} u(\sigma_{\mathcal{L}_1}(\vec{x}, (\vec{t_1}, w_1))) = \int_{\vec{x_1}} u(\vec{x_1})\, V^2_{\mathcal{L}_1(w_1)}(\vec{x}, d\vec{x_1}) d\vec{t_1}, \tag{7.2}$$

and we have to prove that it holds for any measurable u. Indeed, whenever $u(\vec{y})$ is an indicator $\mathbf{1}_{\vec{y} \leq \vec{a}}$, both left-hand and right-hand sides equal $V^2_{\mathcal{L}_1(w_1)}(\vec{x}, \vec{a})$, thus (7.2) holds. Any other function u can be obtained as a limit of linear combinations of such indicators. □

[4]By definition, the Lebesgue-Stieltjes integral $\int f(\vec{x}) g(d\vec{x})$ is the Lebesgue integral of f wrt. the measure μ having cumulative distribution function g.

Volume functions in timed automata. Volume functions as automata admits a matrix notation as follows. We introduce a Q-vector $\mathbf{V}_n(\vec{x})$ of volumes of clock languages and a $Q \times Q$-matrix $\mathbb{V}(\vec{x}, \vec{x}')$ of cumulative volume functions of elements of the transition matrix $\mathbb{T}$: formally

$$\mathbf{V}_{n,q}(\vec{x}) = V_n(\mathcal{L}_q(\vec{x})) \text{ and } \mathbb{V}_{qq'}(\vec{x}, \vec{x}') = V_1(\mathcal{L}(\mathbb{T}_{qq'})(\vec{x}, \vec{x}')) \text{ with } \mathcal{L}_q = \mathcal{L}((\mathbb{T}^n\mathbf{F})_q).$$

It follows from the proposition above that the matrix element $(\mathbb{V}^{\star n})_{qq'}$ (of the matrix power wrt. $\star$) contains the CVF of $\mathcal{L}((\mathbb{T}^n)_{qq'})$, that is of the language of all the clock words of length n leading from q to q'. Finally, we get the formula for volumes:

$$\mathbf{V}_n = \mathbb{V}^{\star n} \star \mathbf{V}_F, \tag{7.3}$$

with $\mathbf{V}_F$ a column vector with $\mathbf{V}_{F,p} = 1$ if p is final, and $\mathbf{V}_{F,p} = 0$ otherwise.

The following property of volume functions will be used in the sequel.

Proposition 43. *In a timed automaton $\mathcal{A}$, for $n \geq 1$, the entries of $\mathbf{V}_n(\vec{x})$ and of $\mathbb{V}^{\star n}(\vec{x}, \vec{x}')$ are continuous wrt. the starting clock vector $\vec{x}$.*

Proof. We prove that for every triplet word π, the volume function V^2_π is continuous wrt. its first component. It is then straightforward to lift the result to the cumulative and non-cumulative volume functions of the n-language of an automaton (finite sum of triplet words).

For a triplet word π, $\mathcal{L}(\pi)(\vec{x}, \vec{x}')$ is actually a polytope, intersection of half-spaces H of equations of one of the following forms:

- either $x_i + s_j \bowtie c$,
- $x_i + s_{|\pi|} \bowtie x'_i$,
- $s_j - s_l \bowtie c$
- or $s_{|\pi|} - s_l \bowtie x'_i$

Choices of $i, j, \bowtie$ and c depend on H. Here s_j is the time since the beginning of the word after j transitions, i.e. $s_j = \sum_{i=1}^{j} t_i$, in particular s_0 is the constant 0. Note that the Jacobian of the change of variables $\vec{t} = (t_1, \ldots, t_{|\pi|})$ to $\vec{s} = (s_1, \ldots, s_{|\pi|})$ is 1. So we can write:

$$V^2_\pi(\vec{x}, \vec{x}') = \int \mathbf{1}_{\mathcal{L}(\pi)(\vec{x},\vec{x}')}(\vec{t})d\vec{t} = \int \prod_{H:\text{half-space}} \mathbf{1}_{H(\vec{s},\vec{x},\vec{x}')}d\vec{s},$$

thus, for any $i \in \{1, \ldots, d\}$:

$$\begin{aligned}\frac{\partial V^2_\pi(\vec{x}, \vec{x}')}{\partial x_i} &= \int \sum_{H_l : x_i \text{ appears}} \pm\delta(x_i + s_{j_i} - c_l \text{ or } x_i + s_{|\pi|} - x'_i) \prod_{H \neq H_l} \mathbf{1}_H d\vec{s} \\ &= \int \sum_{H_l : x_i \text{ appears}} \pm \prod_{H \neq H_l} \mathbf{1}_{H[x_i \leftarrow x'_i - s_{|\pi|} \text{ or } x_i \leftarrow c_l - s_{j_l}]} d\tilde{\vec{s}}.\end{aligned}$$

where $\tilde{\vec{s}}$ is the vector of coordinates of $\vec{s}$ different from j_l for all l such that $H_l : x_i + s_{j_l} \bowtie_{k_l} c_{k_l}$. We note that the Dirac's δs were all eliminated after the integration by s_{j_l} and that the expression that remains under the integral is a proper function of $\vec{x}, \vec{x}'$ and $\tilde{\vec{s}}$. This implies that $V^2_\pi(\vec{x}, \vec{x}')$ was actually continuous with respect to x_i for all i and thus continuous. □

7.2 Generating functions

7.2.1 Definitions

To study volume sequences associated to timed and clock languages we define their generating functions. As usual for generating functions, they allow recovering the sequence, its growth rate, momenta etc; and they have nice compositional properties. Given a timed language $\mathbb{L}$ its *generating function* is defined as follows:

$$f_{\mathbb{L}}(z) = \sum_k z^k V_k(\mathbb{L}).$$

Given a clock language $\mathcal{L}$, we define a *(parametric) generating function* with a given starting clock vector

$$f^1_{\mathcal{L}}(z, \vec{x}) = \sum_k z^k V_k(\mathcal{L}(\vec{x})) = f_L(z), \text{ with } \mathbb{L} = \mathcal{L}(\vec{x}).$$

For a clock language $\mathcal{L}$ we also define another *cumulative generating function* with a given starting clock vector and a bound on the final clock vector:

$$f^2(z, \vec{x}, \vec{x}') = \sum_k z^k V^2_k(\vec{x}, \vec{x}') = f_{\mathbb{L}}(z), \text{ with } \mathbb{L} = \mathcal{L}(\vec{x}, \vec{x}').$$

To summarize, we are interested in computing $f(z)$, but this computation will be based on $f^1(z, \vec{x})$, and sometimes on $f^2(z, \vec{x}, \vec{x}')$.

Given a timed automaton, timed and clock languages, and thus generating functions are naturally associated to its states, for example

$$f^1_q(z, \vec{x}) = \sum_k z^k V_k(\mathcal{L}_q(\vec{x})) = f_{\mathbb{L}}(z), \text{ with } \mathbb{L} = \mathcal{L}_q(\vec{x}).$$

Taken for all states, functions f_q and f^1_q form $|Q|$-dimensional vector functions $\mathbf{f}(z, \vec{x}), \mathbf{f}^1(z, \vec{x})$, while functions $f^2_{q,q'}$ form a $Q \times Q$-matrix function $\mathbb{f}^2(z, \vec{x}, \vec{x}')$.

7.2.2 Analytic characterization

Elementary properties. First, let us state the relations between the three kinds of generating functions:

Proposition 44. *The functions f, f^1, f^2 are related as follows:*

$$f(z) = f^1(z, \mathbf{0});$$
$$f^1(z, \vec{x}) = f^2(z, \vec{x}, \vec{M}).$$

By definition, f^2, f^1 and f are analytic functions of z. Since we consider timed automata with guards bounded by some constant M, all the volumes V_k (with any initial or final conditions) can be upper bounded by $(M|\Sigma|)^k$. This implies that convergence radius of

series for f^2, f^1 and f is at least $(M|\Sigma|)^{-1} > 0$. More precisely, the radius of convergence of f is

$$1/\limsup_{k\to\infty}(V_k(\mathbb{L}))^{1/k} = 2^{-\mathcal{H}(\mathbb{L})}$$

where $\mathcal{H}(\mathbb{L})$ is the volumetric entropy of $\mathbb{L}$.

For generating functions associated to timed automata, the following result is a straightforward corollary of Proposition 43:

Proposition 45. *Within its convergence radius, the generating function $\mathbf{f}^1(z,\vec{x})$ associated to a timed automaton $\mathcal{A}$ is continuous wrt. the starting clock vector $\vec{x}$.*

Integral equation for generating functions. Consider a timed automaton. Using formula (7.3), its generating function can be computed as follows:

$$\mathbf{f}^1(z,\vec{x}) = \sum_k z^k \mathbf{V}_k(\vec{x}) = \sum_k z^k \left(\mathbb{V}^{\star k} \star \mathbf{V}_F\right)(\vec{x}),$$

which implies our first main result.

Theorem 32 (Integral equation)**.** *In the interior of its convergence circle, the generating function $\mathbf{f}^1$ is the unique solution of the integral equation*

$$\mathbf{f}^1 - z\mathbb{V} \star \mathbf{f}^1 = \mathbf{V}_F. \tag{7.4}$$

Example 12 (Example 11, continued)**.** *For the automaton $\mathcal{A}_3$ (using the notation $x \dot{-} y$ for $\max(x-y,0)$):*

$$\mathbb{V} = \begin{pmatrix} \min(x',1) \dot{-} x\mathbf{1}_{y'\geq 0} & \min(y',1) \dot{-} y\mathbf{1}_{x'\geq 0} \\ \min(x',1) \dot{-} x\mathbf{1}_{y'\geq 0} & \min(y',1) \dot{-} y\mathbf{1}_{x'\geq 0} \end{pmatrix};\ \mathbf{V}_F = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Equation (7.4) gives: $f_p^1(z,x,y) = f_q^1(z,x,y) = 1 + z\int_x^1 f_p^1(z,x',0)dx' + z\int_y^1 f_q^1(z,0,y')dy'$. In section 7.3.1 below we develop a technique for solving such equations (for a subclass of automata including this one), and compute the generating function for this language.

7.2.3 Volumes, generating functions and functional analysis

In this section (which can be skipped by a reader not interested in functional analysis), similarly to [AD09a], we rephrase previous results in terms of the spectral theory of linear operators.

Given a timed automaton, consider the Banach space $\mathcal{F}$ of Q-vectors of continuous functions on clock valuations. Thus an element of $\mathcal{F}$ is a vector $\mathbf{v}$ whose components are continuous $v_q : [0;M]^d \to \mathbb{R}$, and $\mathcal{F} = C([0;M]^d)^Q$. The matrix $\mathbb{V}$ corresponds to an operator $\Psi : \mathcal{F} \to \mathcal{F}$ defined by $\Psi(\mathbf{v}) = \mathbb{V} \star \mathbf{v}$ (a variant of this operator plays the central role in [AD09a]). In terms of this operator, Proposition 43 and Equation (7.3) can be rephrased as follows:

Proposition 46 ([AD09a])**.** *Ψ is a bounded linear operator on $\mathcal{F}$ (represented by a matrix of integral operators). The volume vector can be obtained by iteration of this operator: $\mathbf{V}_n = \Psi^n(\mathbf{V}_F)$.*

Recall that, by definition, the *resolvent* of an operator A is $R(\lambda, A) = (A - \lambda I)^{-1}$; it is well defined when λ does not belong to the spectrum of A, in particular for $|\lambda| > \rho$, where ρ denotes the spectral radius. We obtain as a consequence of Theorem 32 another characterization of the generating function:

Proposition 47 (Generating function and resolvent)**.** *The generating function $\mathbf{f}^1$ satisfies the formula: $\mathbf{f}^1 = -z^{-1}R(z^{-1}, \Psi)\mathbf{V}_F$, which holds in the interior of the circle $|z| < \rho(\Psi)^{-1}$.*

7.2.4 Inductive characterization of generating functions

The form of generating functions of finite triplet languages follows from Proposition 41:

Proposition 48. *For a finite triplet language L with maximal word length ℓ, the generating functions f^2, f^1 are piecewise polynomial in $z, \vec{x}, \vec{x}'$ (pieces are polytopes in $\vec{x}, \vec{x}'$) of degree $\leq \ell$ wrt. z and wrt. $\vec{x}$ and $\vec{x}'$.*

More complex languages can be obtained from finite ones using Kleene algebra operations. As usual in the context of generating functions, we suppose that the operations are unambiguous. A language operation is *ambiguous* if a word of the resulting language can be obtained in several ways by composing different words from the operands. We consider first the simple case of timed languages.

Proposition 49. *Generating functions behave well for unambiguous operations on measurable timed languages: $f_{\mathbb{L}_1 \cup \mathbb{L}_2} = f_{\mathbb{L}_1} + f_{\mathbb{L}_2}$; $f_{\mathbb{L}_1 \cdot \mathbb{L}_2} = f_{\mathbb{L}_1} f_{\mathbb{L}_2}$; $f_{\mathbb{L}^*} = 1 + f_{\mathbb{L}} f_{\mathbb{L}^*}$ provided $\epsilon \notin \mathbb{L}$.*

However, in order to obtain general regular timed languages we need operations on clock languages, which are more involved.

Proposition 50. *Generating functions f^2 behave well for unambiguous operations on deterministic measurable clock languages (whenever the resulting language is also deterministic): $f^2_{\mathcal{L}_1 \cup \mathcal{L}_2} = f^2_{\mathcal{L}_1} + f^2_{\mathcal{L}_2}$; $f^2_{\mathcal{L}_1 \cdot \mathcal{L}_2} = f^2_{\mathcal{L}_1} \star f^2_{\mathcal{L}_2}$; $f^2_{\mathcal{L}^*} = \mathbf{1}_{\vec{x} \leq \vec{x}'} + f^2_{\mathcal{L}} \star f^2_{\mathcal{L}^*}$ provided $\mathcal{E} \cap \mathcal{L} = \emptyset$.*

Proof. • Union:

$$f^2_{\mathcal{L}_1 \cup \mathcal{L}_2}(\vec{x}, \vec{x}', z) = \sum_{w \in \Sigma^*} z^{|w|} V^2_{\mathcal{L}_1 \cup \mathcal{L}_2(w)}(\vec{x}, \vec{x}')$$

Since the union is unambiguous, $\forall w \in \Sigma^*$, $\mathcal{L}_1(w) \cap \mathcal{L}_2(w) = \emptyset$ and thus $\forall w \in \Sigma^*$, $V^2_{\mathcal{L}_1 \cup \mathcal{L}_2(w)} = V^2_{\mathcal{L}_1(w)} + V^2_{\mathcal{L}_2(w)}$. Finally :

$$f^2_{\mathcal{L}_1 \cup \mathcal{L}_2}(\vec{x}, \vec{x}', z) = f^2_{\mathcal{L}_1}(\vec{x}, \vec{x}', z) + f^2_{\mathcal{L}_2}(\vec{x}, \vec{x}', z)$$

- Product:

  Recall that:
  $$f^2_{\mathcal{L}_1\cdot\mathcal{L}_2}(\vec{x},\vec{x}',z) = \sum_{w\in\Sigma^*} z^{|w|} V^2_{\mathcal{L}_1\cdot\mathcal{L}_2(w)}(\vec{x},\vec{x}').$$

  Since the product is unambiguous,
  $$(\mathcal{L}_1\cdot\mathcal{L}_2)(w) = \biguplus_{\substack{w_1,w_2\in\Sigma^*\\ w=w_1w_2}} \mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2),$$

  and thus
  $$V^2_{\mathcal{L}_1\cdot\mathcal{L}_2(w)} = \sum_{\substack{w_1,w_2\in\Sigma^*\\ w=w_1w_2}} V^2_{\mathcal{L}_1(w_1)\cdot\mathcal{L}_2(w_2)} = \sum_{\substack{w_1,w_2\in\Sigma^*\\ w=w_1w_2}} V^2_{\mathcal{L}_1(w_1)} \star V^2_{\mathcal{L}_2(w_2)},$$

  with the last equality given by proposition 42. We deduce:
  $$\begin{aligned}\sum_{w\in\Sigma^*} z^{|w|} V^2_{\mathcal{L}_1\cdot\mathcal{L}_2(w)} &= \sum_{w\in\Sigma^*}\sum_{\substack{w_1,w_2\in\Sigma^*\\ w=w_1w_2}} z^{|w_1|}V^2_{\mathcal{L}_1(w_1)} \star z^{|w_2|}V^2_{\mathcal{L}_2(w_2)}\\ &= \sum_{w_1\in\Sigma^*}\sum_{w_2\in\Sigma^*} z^{|w_1|}V^2_{\mathcal{L}_1(w_1)} \star z^{|w_2|}V^2_{\mathcal{L}_2(w_2)}\\ &= \sum_{w_1\in\Sigma^*} z^{|w_1|}V^2_{\mathcal{L}_1(w_1)} \star \sum_{w_2\in\Sigma^*} z^{|w_2|}V^2_{\mathcal{L}_2(w_2)},\end{aligned}$$

  and finally:
  $$f^2_{\mathcal{L}_1\cdot\mathcal{L}_2} = f^2_{\mathcal{L}_1} \star f^2_{\mathcal{L}_2}.$$

- Kleene star:

  We have $\mathcal{L}^* = \mathcal{E} \cup \mathcal{L}\cdot\mathcal{L}^*$, the union is empty since $\mathcal{E}\cap\mathcal{L} = \emptyset$ and the product is unambiguous since the star is unambiguous thus using the two first items:
  $$f^2_{\mathcal{L}^*} = f^2_{\mathcal{E}} + f^2_{\mathcal{L}} \star f^2_{\mathcal{L}^*}.$$

  To complete the proof it remains to prove that $f^2_{\mathcal{E}}(x,x',z) = \mathbf{1}_{\vec{x}\le\vec{x}'}$. Recall that: $\mathcal{E} = \{\vec{x} \xrightarrow{\epsilon} \vec{x} \mid \vec{x}\in\mathbb{R}^d\}$ thus $\mathcal{E}(x,x') = \{\mathbf{w}\in(R^+\times\Sigma)^* \mid \exists\vec{y}\le\vec{x}',\ (\vec{x},\mathbf{w},\vec{y})\in\mathcal{E}\} = \{\mathbf{w}\in(R^+\times\Sigma)^* \mid \vec{x}\le\vec{x}', \mathbf{w}=\epsilon\}$ this set is equal to the 0-dimensional set $\{\epsilon\}$ whose volume is 1 iff $\vec{x}\le\vec{x}'$, otherwise it is empty and thus of volume 0.

□

Corollary 4. *Generating function f^1 for unambiguous compositions of clock languages (under the same hypotheses) can be computed as follows: $f^1_{\mathcal{L}_1+\mathcal{L}_2} = f^1_{\mathcal{L}_1} + f^1_{\mathcal{L}_2}$; $f^1_{\mathcal{L}_1\cdot\mathcal{L}_2} = f^2_{\mathcal{L}_1}\star f^1_{\mathcal{L}_2}$; $f^1_{\mathcal{L}^*} = 1 + f^2_{\mathcal{L}}\star f^1_{\mathcal{L}^*}$ provided $\mathcal{E}\cap\mathcal{L} = \emptyset$.*

7.3 Computing generating functions

The generating function of a timed language represented by an automaton is characterized by a system of integral equations (7.4). The generating function of a timed language represented by a regular expression can be found recursively from piecewise polynomial functions using operations $+, \star$ and solving fixpoint integral equations of Proposition 50 and Corollary 4. Unfortunately, both procedures involve computation of integrals, and solution of integral equations, for this reason, the result cannot be always presented by an explicit formula. Below we consider several subclasses of timed automata, for which generating functions can be obtained in closed form, or at least admit a simpler characterization.

7.3.1 Generating functions for particular classes of automata

System of equations.

Our closed-form solutions for subclasses of timed automata will be obtained using a variant of language equations.

Let $Q = G \cup B$ be a disjoint partition of the states of a timed automaton $\mathcal{A}$ into *good* and *bad*. We want to describe the vector $\overline{\mathbf{L}}$ of triplet languages L_q recognized from good states $q \in G$ only. This vector satisfies the equation:

$$\overline{\mathbf{L}} = \overline{\mathbb{T}} \cdot \overline{\mathbf{L}} + \overline{\mathbf{F}}, \tag{7.5}$$

where $\overline{\mathbb{T}}$ is a $G \times G$-matrix and $\overline{\mathbf{F}}$ is a G-vector of triplet languages. Their elements are defined as follows: $\overline{T}_{pq}$ consists of all words leading from p to q via bad states only; $\overline{F}_p$ consists of all words leading from p to a final state via bad states only[5].

Automata with regeneration.

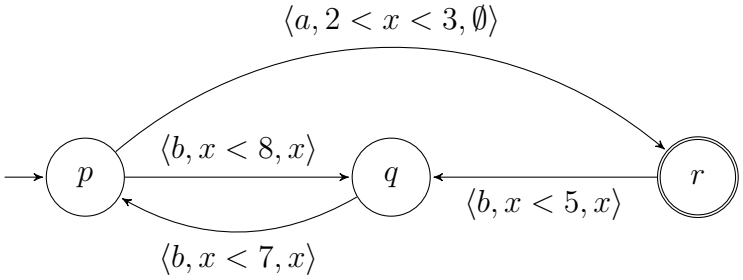

Figure 7.2: A regenerating automaton

Following [SV07], we call an automaton *regenerating* if there exists a partition $Q = G \cup B$ having two properties: (a) every cycle in the automaton contains a state in G (good); (b) all the transitions coming into a good state reset all clocks.

[5] if this final state is good the word should be ϵ.

W.l.o.g. we suppose that the initial state is good (this can be achieved by adding a new initial state). Condition (a) implies that no cycle is possible within bad states, and thus all the elements of $\overline{\mathbb{T}}$ and $\overline{\mathbf{F}}$ are finite triplet languages (with maximal word length $\leq |B|+1$). Condition (b) means that (7.5) can be rewritten in timed languages (instead of clock languages), since when entering in a good state all clocks are reset. This gives

$$\overline{\mathbf{L}}_{\text{timed}} = \overline{\mathbb{T}}_{\text{timed}} \cdot \overline{\mathbf{L}}_{\text{timed}} + \overline{\mathbf{F}}_{\text{timed}}. \tag{7.6}$$

Applying simple compositionality conditions for generating functions for timed languages (Proposition 49) we obtain that $\overline{\mathbf{f}} = \overline{\mathbb{f}}\ \overline{\mathbf{f}} + \overline{\mathbf{f}}_F$. Due to Proposition 48 all the coefficients (elements of matrix $\overline{\mathbb{f}}$ and vector $\overline{\mathbf{f}}_F$) are polynomials of z. Solving this linear $|G|$-dimensional system we express $\overline{\mathbf{f}}$ as a vector of rational functions of z:

$$\overline{\mathbf{f}} = \left(I - \overline{\mathbb{f}}\right)^{-1} \overline{\mathbf{f}}_F. \tag{7.7}$$

The generating function f of the timed language accepted by the automaton is just one element of this vector $\overline{\mathbf{f}}$. We conclude.

Theorem 33. *For a regenerating automaton the generating function $f(z)$ is a rational function.*

Example 13. *Consider a regenerating automaton on Figure 7.2. We choose good and bad states as follows: $G = \{p, q\}; B = \{r\}$. The system of equations on timed languages of good states takes the form*

$$\begin{pmatrix} L_p \\ L_q \end{pmatrix} = \begin{pmatrix} \emptyset & T_{pq} \\ T_{qp} & \emptyset \end{pmatrix} \cdot \begin{pmatrix} L_p \\ L_q \end{pmatrix} + \begin{pmatrix} F_p \\ \emptyset \end{pmatrix} \text{ with} \tag{7.8}$$

$$\begin{aligned} T_{pq} =& \{(t_1, a)(t_2, b) | 2 < t_1 < 3 \wedge t_1 + t_2 < 5\} \cup \{(t, b) | t < 8\}; \\ T_{qp} =& \{(t, b) | t < 7\}; \\ F_p =& \{(t, a) | 2 < t < 3\}. \end{aligned}$$

For generating functions this yields: $\begin{pmatrix} f_p \\ f_q \end{pmatrix} = \begin{pmatrix} 0 & 2.5z^2 + 8z \\ 7z & 0 \end{pmatrix} \cdot \begin{pmatrix} f_p \\ f_q \end{pmatrix} + \begin{pmatrix} z \\ 0 \end{pmatrix}$. *Solving this linear system we find the required $f_p(z) = 2z/(2 - 35z^3 - 112z^2)$. It converges for $|z| <$ 0.1309, its Taylor coefficients (i.e. volumes V_n for $n = 0..11$) are*

$$0; 1; 0; 56; 17\frac{1}{2}; 3136; 1960; 175922\frac{1}{4}; 164640; 9885946; 12298479\frac{3}{8}; 556494176.$$

Real-time automata.

We consider here transition-labeled real-time automata (t-RTA) from [Dim01]. They are automata, in which to any transition $p \xrightarrow{a} q$ is associated a time interval $[l, u]$. This transition can be taken after spending between l and u time units in p. Equivalently, a real-time

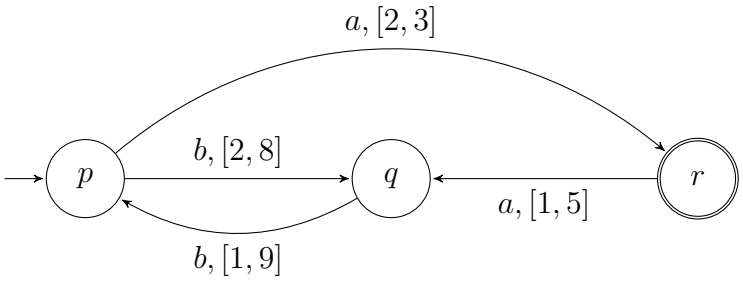

Figure 7.3: A real-time automaton

automaton can be seen as a timed automaton with only one clock, which is reset on any transition.

It is easy to see that real-time automata are regenerating (all their states are good). Thus equation (7.7) applies. Its coefficients can be found in a more explicit form:

$$\mathbf{f} = (I - z\mathbb{A})^{-1}\mathbf{V}_F, \tag{7.9}$$

where matrix A_{pq} is the sum of lengths of all time intervals associated to transitions from p to q, and, as before, $V_{Fq} = 1$ iff q is final, and 0 otherwise. This can be seen as a simplified version of the resolvent equation (7.4) for real-time automata: instead of a matrix of integral operators, a matrix of polynomials is inverted.

Example 14. *For the real-time automaton of Figure 7.3 equation (7.9) takes the form:*

$$\begin{pmatrix} f_p \\ f_q \\ f_r \end{pmatrix} = \left(I - z \begin{pmatrix} 0 & 6 & 1 \\ 8 & 0 & 0 \\ 0 & 4 & 0 \end{pmatrix} \right)^{-1} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

which gives the required generating function:

$$f_p = \frac{z}{1 - 48z^2 - 32z^3}$$

with convergence circle $|z| < 0.13812$ *and first 11 Taylor coefficients (volumes* V_n*):*

$$0; 1; 0; 48; 32; 2304; 3072; 111616; 221184; 5455872; 14188544; 268959744.$$

$1\frac{3}{4}$-clocks automata.

We call an automaton $1\frac{3}{4}$-*clocks* if there exists a partition of $Q = G \cup B$ into good and bad states having three properties:(a) every cycle in the automaton contains a good state; (b) the initial state is a good one; (c) for each good state p there is at most one clock $x_{i(p)}$ not reset by incoming transitions.

Similarly to regenerating automata, we apply equations (7.5), and observe that all the coefficients are finite triplet languages. Unfortunately, since some clocks are not reset, we cannot write an equation on timed languages similar to (7.6). Instead, we pass to clock languages and their generating functions, as in the general case. This gives:

$$\overline{\mathbf{f}^1} = \overline{\mathbb{f}^2} \star \overline{\mathbf{f}^1} + \overline{\mathbf{f}^1}_F, \tag{7.10}$$

an integral equation with piecewise polynomial coefficients. We notice that functions in the last equation depend on the clock vector $\vec{x} \in \mathbb{R}^d$ (or on two clock vectors $\vec{x}$,$\vec{x}'$), but in fact for any good state $p \in G$ only one clock $x_{i(p)}$ matters. This allows extracting simpler integral equations from (7.10), involving only functions of scalar argument.

We proceed as follows: given a G-vector $\mathbf{v}$ whose elements v_p are functions on $\mathbb{R}^d$, we define reduced functions on $\mathbb{R}$: $\widetilde{v}_p(x) = v_p(0,\dots,0,x,0,\dots,0)$, with the argument x at position $i(p)$. Reduced G-vector $\widetilde{\mathbf{v}}$ consists of reduced elements $\widetilde{v}_p$. Reduced versions of matrices are defined similarly.

The following identity is based on the requirement of clock resets:

Lemma 46. *For a $1\frac{3}{4}$-clocks automaton the following holds: $\widetilde{\overline{\mathbb{f}^2} \star \overline{\mathbf{f}^1}} = \widetilde{\overline{\mathbb{f}^2}} \star \widetilde{\overline{\mathbf{f}^1}}$.*

Equation (7.10), reduced to $\widetilde{\overline{\mathbf{f}^1}} = \widetilde{\overline{\mathbb{f}^2}} \star \widetilde{\overline{\mathbf{f}^1}} + \widetilde{\overline{\mathbf{f}^1}_F}$, implies that the reduced vector of generating functions is a solution of equations of the form:

$$\mathbf{f}(z,x) = (\mathbb{A} \star \mathbf{f})(z,x) + \mathbf{b}(z,x), \tag{7.11}$$

where all the coefficients are piecewise polynomial functions of z and a scalar argument x.

Lemma 47. *An integral equation of the form (7.11) can be transformed into a system of linear ordinary differential equation with piecewise polynomial coefficients (depending on x and z).*

Theorem 34. *For a $1\frac{3}{4}$-clocks automaton the generating function f can be obtained by solving a system of linear ordinary differential equations with piecewise polynomial coefficients.*

We notice that the theorem gives a rather explicit characterization of f, but not always a closed-form expression.

Example 15 (Running example completed)**.** *$\mathcal{A}_3$ is $1\frac{3}{4}$-clocks[6] with good states $G = \{p, q\}$ and no bad state $B = \emptyset$. The matrix $\mathbb{A}$ and the vector $\mathbf{b}$ are*

$$\mathbb{A} = \begin{pmatrix} z(\min(x',1) \dot{-} x) & z\min(x',1) \\ z\min(x',1) & z(\min(x',1) \dot{-} x) \end{pmatrix};\ \mathbf{b} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

We use equation (7.11) and remark that by symmetry of $\mathbb{A}$, the two generating functions $\widetilde{f^1_p}$ and $\widetilde{f^1_q}$ are equal to a unique function f^1 which satisfies $f^1(z,x) = z\int_x^1 f^1(z,x')\,dx' + z\int_0^1 f^1(z,x')\,dx' + 1$. Differentiating it one time wrt. x we obtain: $\frac{\partial f^1}{\partial x}(z,x) = -zf^1(z,x)$. The solution has the form $f^1(z,x) = A(z)e^{-zx}$. Remark that $f^1(z,0) - 1 = 2z\int_0^1 f^1(z,x')\,dx' = 2(f^1(z,1) - 1)$. We are done $f(z) = f^1(z,0) = A(z) = 1/(2e^{-z} - 1)$.

[6] $\mathcal{A}_3$ can be seen as $\mathcal{A}_2$ whose state is split to make it a $1\frac{3}{4}$-clock automaton. See also the region splitting of [AD09a] recalled in section 2.3.

Example 16 (Airy)**.** *Consider the automaton $\mathcal{A}_4$ of Figure 7.1. We choose $G = \{p, q\}$, $B = \{r\}$. As in the previous example we have $\mathbb{A}_{pq}(x, x') = \min(1 \dot{-} x, x')\mathbf{1}_{x' \geq 0}$. For $\mathbb{A}_{qp}$, we must compute the volume of the language $\mathcal{L}_{cb}(x, x') = \{t_1, t_2 \geq 0 \mid x + t_1 + t_2 \leq 1 \wedge t_1 + t_2 \leq x'\}$. This is the area of the right-angled triangle of equations $t_1, t_2 \geq 0, t_1 + t_2 \leq u$ where $u = min(1 \dot{-} x, x')$ i.e. $\min(1 \dot{-} x, x')^2/2$. We can now give the matrix $\mathbb{A}$ and the vector $\mathbf{b}$:*

$$\mathbb{A} = \begin{pmatrix} 0 & z \min(1 \dot{-} x, x')\mathbf{1}_{x' \geq 0} \\ z^2 \min(1 \dot{-} x, x')^2 \mathbf{1}_{x' \geq 0}/2 & 0 \end{pmatrix}; \ \mathbf{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Equations (7.11) on generating functions take the form:

$$\widetilde{f_p^1}(z, x) = z \int_0^{1-x} \widetilde{f_q^1}(z, x')\, dx'; \quad \widetilde{f_q^1}(z, x) = z^2 \int_0^{1-x} x' \widetilde{f_p^1}(z, x')\, dx' + 1.$$

Differentiating wrt. x we obtain:

$$\frac{\partial \widetilde{f_p^1}}{\partial x}(z, x) = -z\widetilde{f_q^1}(z, 1 - x); \quad \frac{\partial \widetilde{f_q^1}}{\partial x}(z, x) = -z^2(1 - x)\widetilde{f_p^1}(z, 1 - x).$$

Differentiate once again the former equation gives: $\frac{\partial^2 \widetilde{f_p^1}}{\partial x^2}(z, x) = z\frac{\partial \widetilde{f_q^1}}{\partial x}(z, 1 - x)$, combining with the latter one this gives:

$$\frac{\partial^2 f^1}{\partial x^2}(z, x) = -z^3 x f^1(z, x) \text{ with } f^1 = \widetilde{f_p^1}.$$

The solution has the form $f^1(z, x) = \alpha(z)\mathrm{Ai}(-zx) + \beta(z)\mathrm{Bi}(-zx)$ where Ai *and* Bi *are the Airy's functions and $\alpha(z)$, $\beta(z)$ two functions to be determined with the border conditions $f^1(z, 1) = 1$, $\frac{\partial f^1}{\partial x}(z, 0) = 0$. We obtain the following equations:*

$$\alpha(z)\mathrm{Ai}(-z) + \beta(z)\mathrm{Bi}(-z) = 1; \quad \alpha(z)\mathrm{Ai}'(0) + \beta(z)\mathrm{Bi}'(0) = 0.$$

Solving this system and simplifying using classical formulae for $\mathrm{Ai}(0)$, $\mathrm{Ai}'(0)$, $\mathrm{Bi}(0)$, $\mathrm{Bi}'(0)$ *and Euler's reflection formula, we obtain the final result:*

$$f(z) = f^1(z, 0) = \frac{4}{\pi} \cdot \frac{1}{\mathrm{Bi}'(0)\mathrm{Ai}(-z) - \mathrm{Ai}'(0)\mathrm{Bi}(-z)}.$$

Example 17. *The matrices defining $\mathcal{A}_1$ (the running example of the thesis) are:*

$$\mathbf{I} = \begin{pmatrix} \{\epsilon\} & \emptyset \end{pmatrix}; \ \mathbb{T} = \begin{pmatrix} \emptyset & \{\langle a, x \leq 1, \{x\}\rangle\} \\ \{\langle b, y \leq 1, \{y\}\rangle\} & \emptyset \end{pmatrix}; \ \mathbf{F} = \begin{pmatrix} \{\epsilon\} \\ \{\epsilon\} \end{pmatrix}.$$

The matrices of CVF are:

$$\mathbb{V} = \begin{pmatrix} 0 & \min(1 \dot{-} x, y' \dot{-} y)\mathbf{1}_{x' \geq 0} \\ \min(1 \dot{-} y, x' \dot{-} x)\mathbf{1}_{y' \geq 0} & 0 \end{pmatrix}; \ \mathbf{V}_F = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

The automaton $\mathcal{A}_1$ is $1\frac{3}{4}$-clocks, it suffices to choose $G = \{p, q\}$, $B = \emptyset$. The matrix $\mathbb{A}$ and the vector $\mathbf{b}$ are

$$\mathbb{A} = \begin{pmatrix} 0 & z\min(1 \dot{-} x, x')\mathbf{1}_{x'\geq 0} \\ z\min(1 \dot{-} x, x')\mathbf{1}_{x'\geq 0} & 0 \end{pmatrix}; \ \mathbf{b} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Equations (7.11) on generating functions take the form:

$$\widetilde{f_p^1}(z,x) = z\int_{x'=0}^{1-x} \widetilde{f_q^1}(z,x')\,dx' + 1; \quad \widetilde{f_q^1}(z,x) = z\int_{x'=0}^{1-x} \widetilde{f_p^1}(z,x')\,dx' + 1.$$

By symmetry the two generating functions $\widetilde{f_p^1}$ and $\widetilde{f_q^1}$ are equal to a unique function f^1 which satisfies $f^1(z,x) = z\int_{x'=0}^{1-x} f^1(z,x')\,dx' + 1$. Differentiating it twice wrt. x we obtain:

$$\frac{\partial f^1}{\partial x}(z,x) = -zf^1(z,1-x); \quad \frac{\partial^2 f^1}{\partial x^2}(z,x) = -z^2 f^1(z,x).$$

The solution has the form $f^1(z,x) = A(z)\cos zx + B(z)\sin zx$.

Using $\frac{\partial f^1}{\partial x}(z,0) = -zf^1(z,1) = -z$ we obtain $zB(z) = -z$ and thus $B(z) = -1$. Then $f^1(z,1) = 1$ implies $A(z)\cos z - \sin z = 1$ and thus $A(z) = \frac{1+\sin z}{\cos z} = \tan z + \sec z$. We can conclude

$$f(z) = f^1(z,0) = A(z) = \tan z + \sec z.$$

A reader acquainted with alternating permutations would recognized the exponential generating function of these permutations. This is no accident as we will see in the next chapter.

Example 18. *The automaton $\mathcal{A}_4$ is $1\frac{3}{4}$-clocks, it suffices to choose $G = \{p, q\}$, $B = \{r\}$. As in the previous example we have $\mathbb{A}_{pq}(x,x') = \min(1 \dot{-} x, x')\mathbf{1}_{x'\geq 0}$. For $\mathbb{A}_{qp}$, we must compute the volume of the language $\mathcal{L}_{cb}(x,x') = \{t_1, t_2 \geq 0 \mid x + t_1 + t_2 \leq 1 \wedge t_1 + t_2 \leq x'\}$. This is the area of the right triangle defined by equations $t_1, t_2 \geq 0, t_1 + t_2 \leq u$, where $u = min(1 \dot{-} x, x')$, i.e. $\frac{min(1 \dot{-} x, x')^2}{2}$. We can now give the matrix $\mathbb{A}$ and the vector $\mathbf{b}$:*

$$\mathbb{A} = \begin{pmatrix} 0 & z\min(1 \dot{-} x, x')\mathbf{1}_{x'\geq 0} \\ z^2\frac{min(1 \dot{-} x, x')^2}{2}\mathbf{1}_{x'\geq 0} & 0 \end{pmatrix}; \ \mathbf{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Equations (7.11) on generating functions take the form:

$$\widetilde{f_p^1}(z,x) = z\int_0^{1-x} \widetilde{f_q^1}(z,x')\,dx'; \quad \widetilde{f_q^1}(z,x) = z^2\int_0^{1-x} x'\widetilde{f_p^1}(z,x')\,dx' + 1.$$

Differentiating wrt. x we obtain:

$$\frac{\partial \widetilde{f_p^1}}{\partial x}(z,x) = -z\widetilde{f_q^1}(z,1-x); \quad \frac{\partial \widetilde{f_q^1}}{\partial x}(z,x) = -z^2(1-x)\widetilde{f_p^1}(z,1-x).$$

Differentiate once again the former equation gives: $\frac{\partial^2 \widetilde{f_p^1}}{\partial x^2}(z,x) = z\frac{\partial \widetilde{f_q^1}}{\partial x}(z,1-x)$, combining with the latter one this gives:

$$\frac{\partial^2 f^1}{\partial x^2}(z,x) = -z^3 x f^1(z,x) \text{ with } f^1 = \widetilde{f_p^1}.$$

The solution has the form $f^1(z,x) = \alpha(z)\mathrm{Ai}(-zx) + \beta(z)\mathrm{Bi}(-zx)$ *where* Ai *and* Bi *are the so-called Airy's functions and* $\alpha(z)$, $\beta(z)$ *two functions to be determined with the border conditions* $f^1(z,1) = 1$, $\frac{\partial f^1}{\partial x}(z,0) = 0$. *We obtain the following equations:*

$$\alpha(z)\mathrm{Ai}(-z) + \beta(z)\mathrm{Bi}(-z) = 1; \quad \alpha(z)\mathrm{Ai}'(0) + \beta(z)\mathrm{Bi}'(0) = 0.$$

Solving this system and simplifying using classical formulae for Ai(0), Ai′(0), Bi(0), Bi′(0) *and Euler's reflection formula, we obtain the final result:*

$$f(z) = f^1(z,0) = \frac{4}{\pi} \cdot \frac{1}{\mathrm{Bi}'(0)\mathrm{Ai}(-z) - \mathrm{Ai}'(0)\mathrm{Bi}(-z)}.$$

Example 19. *The matrices of* $\mathcal{A}_2$ *are* 1×1*-dimensional since it has only one state:*

$$\mathbf{I} = (\{\epsilon\})\,;\ \mathbb{T} = (\{\langle a, x \le 1, \{y\}\rangle, \langle b, y \le 1, \{x\}\rangle\})\,;\ \mathbf{F} = (\{\epsilon\})\,.$$

The matrices of CVF are:

$$\mathbb{V} = (\min(x',1) \dot{-} x\mathbf{1}_{y'\ge 0}\ + \min(y',1) \dot{-} y\mathbf{1}_{x'\ge 0})\,;\ \mathbf{V}_F = (1)\,.$$

The equation on generating function is

$$f^1(z,x,y) = 1 + z\int_x^1 f^1(z,x',0)dx' + z\int_y^1 f^1(z,0,y')dy'.$$

This automaton is not $1\frac{3}{4}$*-clocks, but language equivalent to its split form* $\mathcal{A}_3$ *treated in the chapter.*

7.4 Conclusion and perspectives

In this chapter, we introduced generating functions of timed languages, explored their properties and characterized them by integral equations. For subclasses of regular timed languages we have presented closed-form expressions or a simpler characterization of generating functions. Generating functions describe with high accuracy the quantitative behaviour of timed languages.

At the current stage of research, the computation of generating functions is a semi-manual task and restrictions are imposed to automata. We are planning to explore theoretical and practical algorithmics of timed generating functions, and to implement the algorithms On the other hand, we want to see whether closed form solutions are possible beyond the class of $1\frac{3}{4}$-clocks languages.

We hope that this approach will lead to better quantitative characterization of timed languages with applications to information theory and verification of real-time systems. Also, the approach can be extended to timed formal series, non-regular timed languages, or to richer models such as hybrid automata.

Generating functions of timed languages already have an application to enumerative combinatorics of permutations... this is the subject of the next chapter.

Chapter 8

Counting and generating permutations using timed languages

Abstract of the chapter

The signature of a permutation σ is a word $\mathtt{sg}(\sigma) \subseteq \{\mathbf{a}, \mathbf{d}\}^*$ whose i^{th} letter is $\mathbf{a}$ when σ has a descent (i.e. $\sigma(i) > \sigma(i+1)$) and is $\mathbf{d}$ when σ has an ascent (i.e. $\sigma(i) < \sigma(i+1)$). Combinatorics of permutations with a prescribed signature is quite well explored. Languages of signatures permit to express a broad number of classes of permutations (e.g. the permutations with an even number of descents). Here we state and address the two problems of counting and randomly generating in the set $\mathtt{sg}^{-1}(L)$ of permutations with signature in a given regular language $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$. First we give an algorithm that computes a closed form formula for the exponential generating function of $\mathtt{sg}^{-1}(L)$. Then we give an algorithm that generates randomly the n-length permutations of $\mathtt{sg}^{-1}(L)$ in a uniform manner, i.e. all the permutations of a given length with a signature in L are equally probable to be returned. Both contributions are based on a geometric interpretation of a subclass of regular timed languages.

Chapter structure

In section 8.1 we expose the problem statements. In section 8.2 we establish the link between the classes of permutations associated with languages of signatures and timed languages of a particular form. We address the two problems in section 8.3, treat several examples and discuss our results and perspectives in the last section.

8.1 Two problem statements

All along the chapter we use the two letter alphabet $\{\mathbf{a}, \mathbf{d}\}$ whose elements must be read as "ascent" and "descent". Words of $\{\mathbf{a}, \mathbf{d}\}^*$ are called signatures. For $n \in \mathbb{N}$ we denote $[n] = \{1, \ldots, n\}$ and by $\mathfrak{S}_n$ the set of permutation of $[n]$. We also use the one line notation of permutations e.g. $\sigma = 231$ means that $\sigma(1) = 2$, $\sigma(2) = 3$, $\sigma(3) = 1$.

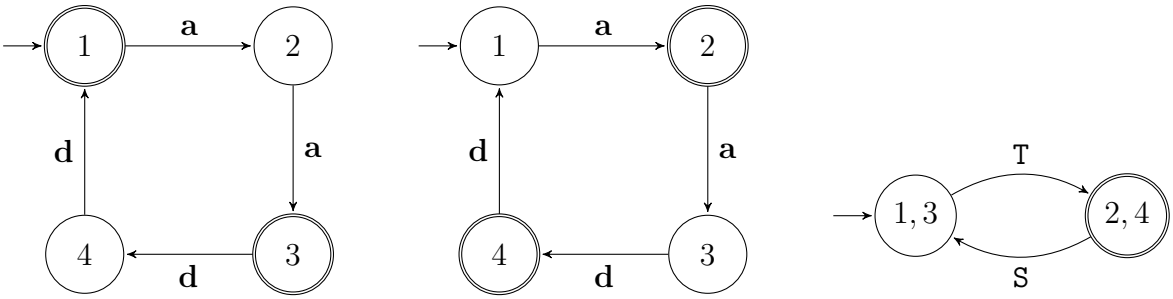

Figure 8.1: From left to right: automata for L^{ex}, $L^{ex'}$ and $\mathtt{st}_{\mathbf{d}}(L^{ex'})$

Let n be a positive integer. The *signature* of a permutation $\sigma = \sigma_1 \cdots \sigma_n$ is the word $u = u_1 \cdots u_{n-1} \in \{\mathbf{a}, \mathbf{d}\}^{n-1}$ denoted by $\mathtt{sg}(\sigma)$ such that for $i \in [n]$, $\sigma_i < \sigma_{i+1}$ iff $u_i = \mathbf{a}$ (we speak of an "ascent", also known as an ascent) and $\sigma_i > \sigma_{i+1}$ iff $u_i = \mathbf{d}$ (we speak of a "descent", also known as a descent) e.g. $\mathtt{sg}(21354) = \mathtt{sg}(32451) = \mathbf{daad}$.

This notion appears in the literature under several different names and forms such as descent word, descent set, ribbon diagram, etc. The usual definition of signature of a permutation is an n-tuple of $+1$ ("ascent") and -1 ("descent"). Here we use words to express in a very convenient way constraints on permutations in terms of languages. More precisely we are interested in $\mathtt{sg}^{-1}(L) = \{\sigma \mid \mathtt{sg}(\sigma) \in L\}$: the class of permutations with a signature in $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$. Given a language L we denote by L_n the sub-language of L restricted to words of length n. The exponential generating function of $\mathtt{sg}^{-1}(L)$ is

$$EGF[\mathtt{sg}^{-1}(L)](z) = \sum_{\sigma \in \mathtt{sg}^{-1}(L)} \frac{z^{|\sigma|}}{|\sigma|!} = \sum_{n \geq 1} |\mathtt{sg}^{-1}(L_{n-1})| \frac{z^n}{n!} = \sum_{u \in L} |\mathtt{sg}^{-1}(u)| \frac{z^{|u|+1}}{(|u|+1)!}.$$

Example 20. *Consider as a running example of the chapter the class of "up-up-down-down" permutations with signature in the language*[1] $L^{ex} = (\mathbf{aadd})^*(\mathbf{aa} + \varepsilon)$ *recognized by the automaton depicted in the left of Figure 8.1. The theory developed in the chapter permits to find the exponential generating function of* $\mathtt{sg}^{-1}(L^{ex})$.

$$EGF[\mathtt{sg}^{-1}(L^{ex})](z) = \frac{\sinh(z) - \sin(z) + \sin(z)\cosh(z) + \sinh(z)\cos(z)}{1 + \cos(z)\cosh(z)}.$$

Its Taylor expansion is

$$z + \frac{z^3}{3!} + 6\frac{z^5}{5!} + 71\frac{z^7}{7!} + 1456\frac{z^9}{9!} + 45541\frac{z^{11}}{11!} + 2020656\frac{z^{13}}{13!} + \ldots.$$

For instance, there are 1456 *up-up-down-down permutations of length* 9.

Now we state the two problems solved in this chapter.

Problem 1. *Design an algorithm which takes as input a regular language* $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$ *and returns a closed form formula for* $EGF(\mathtt{sg}^{-1}(L))$

[1] We confuse regular expressions with the regular languages they express.

Problem 2. *Design an algorithm which takes as input a regular language $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$ and a positive integer n and returns a random permutation σ uniformly in $\mathtt{sg}^{-1}(L_{n-1})$ i.e. such that the probability for each $\sigma \in \mathtt{sg}^{-1}(L_{n-1})$ to be returned is $1/|\mathtt{sg}^{-1}(L_{n-1})|$.*

8.2 A timed and geometric approach

In section 8.2.1 we introduce a sequence of sets $\mathcal{O}_n(L) \subseteq [0,1]^n$ and see how the two problems posed can be reformulated as computing the volume generating function of the sequence $(\mathcal{O}_n(L))_{n\geq 1}$ and generating points uniformly in $\mathcal{O}_n(L)$. Then we define a timed language $\mathbb{L}'$ associated to L as well as its volume (section 8.2.2) and describe a volume preserving transformation between $\mathcal{O}_n(L)$ and $\mathbb{L}'_n$.

8.2.1 Order sets of a language of signatures $(\mathcal{O}_n(L))_{n\geq 1}$

We say that a collection of polytopes $(S_1, \cdots, S_n)$ is an almost disjoint partition of a set A if A is the union of S_i and they have pairwise a null volume intersection. In this case we write $S = \bigsqcup_{i=1}^n S_i$.

The set $\{(\nu_1, \ldots, \nu_n) \in [0,1]^n \mid 0 \leq \nu_{\sigma_1^{-1}} \leq \ldots \leq \nu_{\sigma_n^{-1}} \leq 1\}$ is called the order simplex[2] of σ and denoted by $\mathcal{O}(\sigma)$ e.g. $\vec{\nu} = (0.3, 0.2, 0.4, 0.5, 0.1)$ belongs to $\mathcal{O}(32451)$ since $\nu_5 \leq \nu_2 \leq \nu_1 \leq \nu_3 \leq \nu_4$ and $(32451)^{-1} = 52134$. The set $O(\sigma)$ for $\sigma \in \mathfrak{S}_n$ forms an almost disjoint partition of $[0,1]^n$. By symmetry all the order simplices of permutations have the same volume which is $1/n!$.

If $\vec{\nu}$ is uniformly sampled in $[0,1]^n$ then it falls in any $O(\sigma)$ with probability $1/n!$. To retrieve σ from $\vec{\nu}$ it suffices to use a sorting algorithm. We denote by $\Pi(\vec{\nu})$ the permutation σ returned by the sorting algorithm on $\vec{\nu}$ i.e. such that $0 \leq \nu_{\sigma_1^{-1}} \leq \ldots \leq \nu_{\sigma_n^{-1}} \leq 1$.

The *signature* of a vector $\vec{\nu} \in [0,1]^n$ is the word $\mathtt{sg}(\vec{\nu}) = \mathtt{sg}(\Pi(\nu))$ i.e. such that $\nu_i < \nu_{i+1}$ iff $u_i = \mathbf{a}$ and $\nu_i > \nu_{i+1}$ iff $u_i = \mathbf{d}$. e.g. $\mathtt{sg}(0.3, 0.2, 0.4, 0.5, 0.1) = \mathbf{daad}$. The *order polytope* [Sta86] of a signature $u \in \{\mathbf{a}, \mathbf{d}\}^{n-1}$ is the polytope $\mathcal{O}(u) = \{\vec{\nu} \in [0,1]^n \mid \mathtt{sg}(\vec{\nu}) = u\}$. It is clear that the collection of order simplices $\mathcal{O}(\sigma)$ with all σ having the same signature u form an almost disjoint partition of the order polytope $\mathcal{O}(u)$: $\mathcal{O}(u) = \bigsqcup_{\sigma \in \mathtt{sg}^{-1}(u)} \mathcal{O}(\sigma)$ (e.g. $\mathcal{O}(\mathbf{daa}) = \mathcal{O}(2134) \sqcup \mathcal{O}(3124) \sqcup \mathcal{O}(4123)$). Passing to volume we get:

$$\mathtt{Vol}(\mathcal{O}(u)) = \sum_{\sigma \in \mathtt{sg}^{-1}(u)} \mathtt{Vol}(\mathcal{O}(\sigma)) = \frac{|\mathtt{sg}^{-1}(u)|}{n!} \tag{8.1}$$

Let L be a language of signatures and $n \geq 1$, then the family $(\mathcal{O}(u))_{u \in L_{n-1}}$ forms an almost disjoint partition of a subset of $[0,1]^n$ called the n^{th} order set of L and denoted by $\mathcal{O}_n(L)$:

$$\mathcal{O}_n(L) = \bigsqcup_{u \in L_{n-1}} \mathcal{O}(u) = \bigsqcup_{\sigma \in \mathtt{sg}^{-1}(L_{n-1})} \mathcal{O}(\sigma) = \{\vec{\nu} \in [0,1]^n \mid \mathtt{sg}(\vec{\nu}) \in L_{n-1}\}. \tag{8.2}$$

[2]Order simplices, order and chain polytopes of signatures defined here are particular cases of Stanley's order and chain polytopes [Sta86].

For volumes we get:

$$\mathtt{Vol}(\mathcal{O}_n(L)) = \sum_{u\in L_{n-1}} \mathtt{Vol}(\mathcal{O}(u)) = \sum_{\sigma\in \mathtt{sg}^{-1}(L_{n-1})} \mathtt{Vol}(\mathcal{O}(\sigma)) = \frac{|\mathtt{sg}^{-1}(L_{n-1})|}{n!} \tag{8.3}$$

Reformulating the two problems with the geometric approach

As a consequence of (8.3), Problem 1 can be reformulated as computing the *volume generating function* (VGF) of the sequence $\mathcal{O}(L) =_{\text{def}} (\mathcal{O}_n(L))_{n\geq 1}$:

$$VGF(\mathcal{O}(L))(z) =_{\text{def}} \sum_{n\geq 1} \mathtt{Vol}(\mathcal{O}_n(L))z^n = EGF(\mathtt{sg}^{-1}(L))(z) \tag{8.4}$$

Problem 2 can also be treated using order polytopes $\mathcal{O}_n(L)$. Indeed it suffices to generate uniformly a vector $\nu \in \mathcal{O}_n(L)$ and then sort it to get a permutation $\sigma = \Pi(\nu)$. As the simplices $\mathcal{O}(\sigma)$ for $\sigma \in \mathtt{sg}^{-1}(L_n)$ form an almost disjoint partition of $\mathcal{O}_n(L)$ and all these simplices have the same volume $1/n!$, they are equally probable to receive the random vector ν, and thus all $\sigma \in \mathtt{sg}^{-1}(L_n)$ have the same probability to be chosen.

We have seen with (8.2) that permutations of a fixed length n fit well with the n^{th} order set. However, it is not clear how to fit the sequence of order sets (when n varies) with the dynamics of the language L. It is easier to handle a timed language $\mathbb{L}$ since its sequence of volumes $(\mathtt{Vol}(\mathbb{L}_n))_{n\in\mathbb{N}}$ satisfies a recursive equation (see [AD09a] and (2.4) in the present thesis). We will find a volume preserving transformation between order sets $\mathcal{O}_n(L)$ and timed languages $(\mathbb{L}_n)_{n\in\mathbb{N}}$ and hence reduce Problem 1 to the computation of the ordinary generating function of $(\mathtt{Vol}(\mathbb{L}_n))_{n\in\mathbb{N}}$. For the second problem, by generating uniformly a timed word in $\mathbb{L}_n$ and applying the volume preserving transformation we will get a uniform random point in $\mathcal{O}_n(L)$.

8.2.2 Timed semantics of a language of signatures $(\mathbb{L}'_n)_{n\in\mathbb{N}}$

In this section, we expose the material on timed languages needed to solve our problem. We adopt as in the previous chapter an approach based on the notion of clock languages introduced by [BP02]. The material of the present chapter is simpler than that of the previous one, that is why we choose to give a self-contained exposition. In particular, the next paragraph is essential for a reader that is only interested by the present chapter of the thesis but can be skipped by a reader that has already encountered the definition of volume sequences of a timed language and its generating function.

Timed languages, their volumes and their generating functions

An alphabet of *timed events* is the product $\mathbb{R}^+ \times \Sigma$ where Σ is a finite alphabet. The meaning of a timed event (t_i, w_i) is that t_i is the time delay before the event w_i. A *timed word* is just a word of timed events and a *timed language* a set of timed words. Adopting a geometric point of view, a timed word is a vector of delays $(t_1, \ldots, t_n) \in \mathbb{R}^n$ together with a

word of events $w = w_1 \cdots w_n \in \Sigma^n$. We adopt the following convention, we write $(\vec{t}, w)$ for the timed word $(t_1, w_1) \cdots (t_n, w_n)$ with $\vec{t} = (t_1, \cdots, t_n)$ and $w \in \Sigma^n$ $(n \geq 1)$. Continuing with the same convention, given a timed language $\mathbb{L}' \subseteq (\mathbb{R}^+ \times \Sigma)^*$, then the timed language restricted to words of length n, $\mathbb{L}'_n$ can be seen as a formal union of sets $\biguplus_{w \in \Sigma^n} \mathbb{L}'_w \times \{w\}$ where $\mathbb{L}'_w = \{\vec{t} \in \mathbb{R}^n \mid (\vec{t}, w) \in \mathbb{L}'\}$ is the set of delay vectors that together with w form a timed word of $\mathbb{L}'$. In the sequel we will only consider languages $\mathbb{L}'$ for which every $\mathbb{L}'_w$ is volume measurable. To such a $\mathbb{L}'_n$ one can associate a sequence of volumes and a VGF as follows:

$$\texttt{Vol}(\mathbb{L}'_n) = \sum_{w \in \Sigma^n} \texttt{Vol}(\mathbb{L}'_w);$$

$$VGF(\mathbb{L})(z) = \sum_{w \in \Sigma^*} \texttt{Vol}(\mathbb{L}'_w) z^{|w|} = \sum_{n \in \mathbb{N}} \texttt{Vol}(\mathbb{L}'_n) z^n.$$

The clock semantics of a signature.

A clock is a non-negative real variable. Here we only consider two *clocks* bounded by 1 and denoted by $x^{\mathbf{a}}$ and $x^{\mathbf{d}}$. A clock word is a tuple whose component are a starting clock vector $(x_0^{\mathbf{a}}, x_0^{\mathbf{d}}) \in [0,1]^2$, a timed word $(t_1, a_1) \cdots (t_n, a_n) \in ([0,1] \times \{\mathbf{a}, \mathbf{d}\})^*$ and an ending clock vector $(x_n^{\mathbf{a}}, x_n^{\mathbf{d}}) \in [0,1]^2$, it is denoted by $(x_0^{\mathbf{a}}, x_0^{\mathbf{d}}) \xrightarrow{(t_1,a_1)\cdots(t_n,a_n)} (x_n^{\mathbf{a}}, x_n^{\mathbf{d}})$.

Two clock words $\vec{x}_0 \xrightarrow{\mathbf{w}} \vec{x}_1$ and $\vec{x}_2 \xrightarrow{\mathbf{w}'} \vec{x}_3$ are said to be compatible if $\vec{x}_2 = \vec{x}_1$, in this case their product is $(\vec{x}_0 \xrightarrow{\mathbf{w}} \vec{x}_1) \cdot (\vec{x}_2 \xrightarrow{\mathbf{w}'} \vec{x}_3) = \vec{x}_0 \xrightarrow{\mathbf{w}\mathbf{w}'} \vec{x}_3$. A *clock language* is a set of clock words. The product of two clock languages $\mathcal{L}$ and $\mathcal{L}'$ is

$$\mathcal{L} \cdot \mathcal{L}' = \{c \cdot c' \mid c \in \mathcal{L},\ c' \in \mathcal{L}',\ c \text{ and } c' \text{ compatible}\}. \tag{8.5}$$

The clock language[3] $\mathcal{L}(\mathbf{a})$ (resp. $\mathcal{L}(\mathbf{d})$) associated to an ascent (resp. a descent) is the set of clock words of the form $(x^{\mathbf{a}}, x^{\mathbf{d}}) \xrightarrow{(t,\mathbf{a})} (x^{\mathbf{a}}+t, 0)$ (resp. $(x^{\mathbf{a}}, x^{\mathbf{d}}) \xrightarrow{(t,\mathbf{d})} (0, x^{\mathbf{d}}+t)$) and such that $x^{\mathbf{a}}+t \in [0,1]$ and $x^{\mathbf{d}}+t \in [0,1]$ (and by definition of clocks and delays $x^{\mathbf{a}} \geq 0, x^{\mathbf{d}} \geq 0, t \geq 0$). These definitions extend inductively to all signatures $\mathcal{L}(u_1 \cdots u_n) = \mathcal{L}(u_1) \cdots \mathcal{L}(u_n)$ (using the product of clock languages as defined in (8.5)).

Example 21. $(0,0) \xrightarrow{(0.7,\mathbf{d})(0.2,\mathbf{a})(0.2,\mathbf{a})(0.5,\mathbf{d})} (0,0.5) \in \mathcal{L}(\mathbf{daad})$ *since*

$(0,0) \xrightarrow{(0.7,\mathbf{d})} (0,0.7) \in \mathcal{L}(\mathbf{d})$; $\quad (0,0.7) \xrightarrow{(0.2,\mathbf{a})} (0.2,0) \in \mathcal{L}(\mathbf{a})$;

$(0.2,0) \xrightarrow{(0.2,\mathbf{a})} (0.4,0) \in \mathcal{L}(\mathbf{a})$; $\quad (0.4,0) \xrightarrow{(0.5,\mathbf{a})} (0,0.5) \in \mathcal{L}(\mathbf{d})$.

The timed semantics of a language of signatures.

The *timed polytope* associated to a signature $w \in \{\mathbf{a}, \mathbf{d}\}^*$ is $P_w =_{\text{def}} \{\vec{t} \mid (0,0) \xrightarrow{(\vec{t},w)} \vec{y} \in \mathcal{L}(w)$ for some $\vec{y} \in [0,1]^2\}$ e.g. $(0.7, 0.2, 0.2, 0.5, 0.1) \in P_{\mathbf{daada}}$. The definition of such a

[3] A reader acquainted with timed automata would have noticed that the clock language $\mathcal{L}(\mathbf{a})$ (resp. $\mathcal{L}(\mathbf{d})$) corresponds to a transition of a timed automaton where the guards $x^{\mathbf{a}} \leq 1$ and $x^{\mathbf{d}} \leq 1$ are satisfied and where $x^{\mathbf{d}}$ (resp. $x^{\mathbf{a}}$) is reset. See also the definition given in section 2.1.2 of this thesis.

timed polytope will be clarified in Proposition 51 and its following example. The timed semantics of a language of signatures L' is

$$\mathbb{L} = \{(\vec{t}, w) \mid \vec{t} \in P_w \text{ and } w \in L'\} = \cup_{w \in L'} P_w \times \{w\}.$$

This language restricted to words of length n is $\mathbb{L}'_n = \cup_{w \in L'_n} P_w \times \{w\}$, its volume is $\texttt{Vol}(\mathbb{L}'_n) = \sum_{w \in L'} \texttt{Vol}(P_w)$.

The *chain polytope* [Sta86] of a signature u is the set $\mathcal{C}(u)$ of vectors $\vec{t} \in [0,1]^n$ such that for all $i < j \leq n$ and $l \in \{\mathbf{a}, \mathbf{d}\}$, $w_i \cdots w_{j-1} = l^{j-i} \Rightarrow t_i + \ldots + t_j \leq 1$.

Proposition 51. *Given a word $u \in \{\mathbf{a}, \mathbf{d}\}^*$ and $l \in \{\mathbf{a}, \mathbf{d}\}$, the timed polytope of ul is the chain polytope of u: $P_{ul} = \mathcal{C}(u)$.*

Proof. Let $w = ul$ i.e. for all $i \in [n-1]$ $w_i = u_i$ and $w_n = l$. $P_{ul} \subseteq \mathcal{C}(u)$) Let $(t_1, \ldots, t_n) \in P_w$ i.e. there exist value of clocks x^a_k ($a \in \{\mathbf{a}, \mathbf{d}\}, k \in [n]$) such that $x^{\mathbf{a}}_0 = x^{\mathbf{d}}_0 = 0$ and $(x^{\mathbf{a}}_{k-1}, x^{\mathbf{d}}_{k-1}) \xrightarrow{(t_k, w_k)} (x^{\mathbf{a}}_k, x^{\mathbf{d}}_k) \in \mathcal{L}(w_k)$. Let $i < j \leq n$ and $a \in \{\mathbf{a}, \mathbf{d}\}$ such that $w_i \cdots w_{j-1} = a^{j-i}$, then for $k \in \{i, \ldots, j-1\}$, $x^a_k = x^a_{k-1} + t_k$ by definition of $\mathcal{L}(a)$. Then $x^a_{j-1} = x^a_{i-1} + t_i + \ldots + t_{j-1}$. Moreover $x^a_{j-1} + t_j \leq 1$ by definition of $\mathcal{L}(w_j)$ and thus $t_i + \ldots + t_{j-1} + t_j \leq x^a_{i-1} + t_j \leq 1$ which is the wanted inequality.
$\mathcal{C}(u) \subseteq P_{ul}$) Let $(t_1, \ldots, t_n) \in \mathcal{C}(u)$. We show inductively that for every $a \in \{\mathbf{a}, \mathbf{d}\}$, the condition $x^a_{j-1} + t_j \leq 1$ is satisfied and thus that x^a_j can be defined ($x^a_j = x^a_{j-1} + t_j$ if $w_j = a$ and $x^a_j = 0$ otherwise). For this we suppose that clock values $x^a_0, \ldots, x^a_{j-1}$ are well defined. Let $lr(x^a, j)$ be the maximal index before transition j such that $w_{lr(x^a,j)} \neq a$. Necessarily $w_{lr(x^a,j)+1} \ldots w_j = a^{j-i}$ and thus $t_{lr(x^j)+1} + \ldots + t_j \leq 1$ by definition of $\mathcal{C}(u)$. This latter sum is equal to $x^a_{j-1} + t_j \leq 1$ and thus the condition on x^a imposed by $\mathcal{L}(u_j)$ is satisfied. □

Example 22. *A vector $(t_1, t_2, t_3, t_4, t_5) \in [0,1]^5$ belongs to $P_{\mathbf{daada}} = \mathcal{C}(\mathbf{daad})$ iff $t_1 + t_2 \leq 1, t_2 + t_3 + t_4 \leq 1, t_4 + t_5 \leq 1$ iff $1 - t_1 \geq t_2 \leq t_2 + t_3 \leq 1 - t_4 \geq t_5$ iff $(1 - t_1, t_2, t_2 + t_3, 1 - t_4, t_5) \in \mathcal{O}(\mathbf{daad})$. One can check this fact on examples given before: $(0.7, 0.2, 0.2, 0.5, 0.1) \in P_{\mathbf{daada}}$ corresponds to the vector $(0.3, 0.2, 0.4, 0.5, 0.1) \in \mathcal{O}(\mathbf{daad})$.*

The purpose of the following section is to give the general formula for the correspondence between timed polytopes and order polytopes we have foreseen in the previous example.

8.2.3 Volume preserving transformation between $\mathbb{L}'_n$ and $\mathcal{O}_n(L)$.

Let n be a positive integer. We define for $w = ul$ with $u \in \{\mathbf{a}, \mathbf{d}\}^{n-1}$ and $l \in \{\mathbf{a}, \mathbf{d}\}$ a volume preserving function $(t_1, \cdots, t_n) \mapsto (\nu_1, \cdots, \nu_n)$ from the chain polytope $\mathcal{C}(u) = P_{ul}$ to the order polytope $\mathcal{O}(u)$. This is a simple case of Theorem 2.1 of [HL12].

Let $w \in \{\mathbf{a}, \mathbf{d}\}^n$ and $n = |w|$. Let $j \in [n]$ and i be the index such that $w_i \cdots w_{j-1}$ is a maximal ascending or descending block i.e. i is minimal such that $w_i \cdots w_{j-1} = l^{j-i}$ with $l \in \{\mathbf{a}, \mathbf{d}\}^*$. If $w_j = \mathbf{d}$ we define $\nu_j = 1 - \sum_{k=i}^{j} t_k$ and $\nu_j = \sum_{k=i}^{j} t_k$ otherwise.

Proposition 52. *The mapping $\phi_{ul} : (t_1, \cdots, t_n) \mapsto (\nu_1, \cdots, \nu_n)$ is a volume preserving transformation from $\mathcal{C}(u) = P_{ul}$ to $\mathcal{O}(u)$. It can be computed in linear time using the following recursive characterization:*

$$\left|\begin{array}{llll} \nu_1 = & t_1 & \text{if} & w_1 = \mathbf{a} \\ \nu_1 = & 1 - t_1 & \text{if} & w_1 = \mathbf{d} \end{array}\right. \quad \text{and for } i \geq 2\text{:} \quad \left|\begin{array}{llll} \nu_i = & \nu_{i-1} + t_i & \text{if} & w_{i-1}w_i = \mathbf{aa}; \\ \nu_i = & t_i & \text{if} & w_{i-1}w_i = \mathbf{da}; \\ \nu_i = & 1 - t_i & \text{if} & w_{i-1}w_i = \mathbf{ad}; \\ \nu_i = & \nu_{i-1} - t_i & \text{if} & w_{i-1}w_i = \mathbf{dd}. \end{array}\right.$$

Proof. The function ϕ_{ul} is a volume preserving transformation since it is a linear function given by a unimodular (i.e. an integer matrix having determinant $+1$ or -1) matrix. Indeed $\phi_{ul}(\vec{t}) = \vec{\nu}$ iff $\vec{\nu}^\top = M_{ul}\vec{t}^\top + \vec{b}$ with for all $j \in [n]$: if $w_j = \mathbf{a}$ (resp. $w_j = \mathbf{d}$) then the j^{th} row of the matrix M_{ul} has only 1s (resp. -1s) between coordinates i and j included and the j^{th} row of $\vec{b}$ is 0 (resp. -1). One can see that M_{ul} is upper triangular and has only 1 and -1 on its diagonal and thus is unimodular. Now it remains to prove that $\vec{\nu}$ $(= \phi_{ul}(\vec{t}))$ belongs to $\mathcal{O}(u)$ for $\vec{t} \in \mathcal{C}(u)$. For this we show that the two conditions (C-1) and (C-2) below are equivalent; the former is the definition of $(t_1, \cdots, t_n) \in \mathcal{C}(u)$ while the latter is equivalent to $\nu_1, \ldots, \nu_n \in \mathcal{O}(u)$:

(C-1) for all $i < j \leq n$ and $l \in \{\mathbf{a}, \mathbf{d}\}$, $u_i \cdots u_{j-1} = l^{j-i} \Rightarrow t_i + \ldots + t_j \leq 1$;

(C-2) for all $i < j \leq n$, $u_i \cdots u_{j-1} = \mathbf{a}^{j-i} \Rightarrow \nu_i \leq \ldots \leq \nu_j \leq 1$ and $u_i \cdots u_{j-1} = \mathbf{d}^{j-i} \Rightarrow \nu_j \leq \ldots \leq \nu_i \leq 1$.

Let $i < j \leq n$ and $u_i \cdots u_{j-1} = \mathbf{a}^{j-i}$ then the following chain of inequalities $[0 \leq \nu_i = t_i \leq \ldots \leq \nu_{j-1} = t_i + \ldots + t_{j-1} \leq \nu_j = (1 - t_j \text{ or } t_i + \ldots + t_j) \leq 1]$ is equivalent to $t_i + \ldots + t_j \leq 1$. The case of descents can be proved in a similar way by applying $x \mapsto 1 - x$ to the preceding inequalities. □

Proposition 52 links the timed polytope of a signature of length $n + 1$ and the order polytopes of a signature of length n. We correct this mismatch of length using prolongation of languages. We say that a language L' is a *prolongation* of a language L whenever the truncation of the last letter $w_1 \ldots w_n \mapsto w_1 \ldots w_{n-1}$ is a bijection between L' and L. Every language L has prolongations e.g. $L' = Ll$ for $l \in \{\mathbf{a}, \mathbf{d}\}$ are prolongations of L. A prolongation of L^{ex} is $L^{ex'} = (\mathbf{aadd})^*(\mathbf{aad} + \mathbf{a})$ recognized by the automaton depicted in the middle of Figure 8.1.

Now we can extend Proposition 52 to a language of signatures.

Theorem 35. *Let $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$ and $\mathbb{L}'$ be the timed semantics of a prolongation of L then for all $n \in \mathbb{N}$, the following function is a volume preserving transformation between $\mathbb{L}'_n$ and $\mathcal{O}_n(L)$. Moreover it is computable in linear time.*

$$\begin{array}{rccc} \phi & : & \mathbb{L}'_n & \to & \mathcal{O}_n(L) \\ & & (\vec{t}, w) & \mapsto & \phi_w(\vec{t}) \end{array} \tag{8.6}$$

As a consequence, the two problems can be solved if we know how to compute the VGF of a timed language $\mathbb{L}'$ and how to generate timed vector uniformly in $\mathbb{L}'_n$. A characterization of the VGF of a timed language as a solution of a system of differential equations is done in our previous work [ABDP12]. Nevertheless the equations of this article were quite uneasy to handle and did not give a closed form formula for the VGF. To get simpler equations than in [ABDP12] we work with a novel class of timed languages involving two kinds of transitions $\mathtt{S}$ and $\mathtt{T}$.

8.2.4 The S-T (timed) language encoding.

The S-T-encoding

We consider the finite alphabet $\{\mathtt{S},\mathtt{T}\}$ whose elements must be respectively read as *straight* and *turn*. The S-T-encoding of type $l \in \{\mathbf{a},\mathbf{d}\}$ of a word $w \in \{\mathbf{a},\mathbf{d}\}^*$ is a word $w' \in \{\mathtt{S},\mathtt{T}\}^*$ denoted by $\mathtt{st}_l(w)$ and defined recursively as follows: for every $i \in [n]$, $w'_i = \mathtt{S}$ if $w_i = w_{i-1}$ and $w'_i = \mathtt{T}$ otherwise, with the convention that $w_0 = l$. The mapping $\mathtt{st}_l$ is invertible: $w = \mathtt{st}_l^{-1}(w')$ is defined recursively as follows: for every $i \in [n]$, $w_i = w_{i-1}$ if $w'_i = \mathtt{S}$ and $w_i \neq w_{i-1}$ otherwise, with convention that $w_0 = l$. Notion of S-T-encoding can be extended naturally to languages e.g. for the running example: $\mathtt{st}_{\mathbf{d}}(L^{ex'}) = (\mathtt{TS})^*\mathtt{T}$. We call an S-T-automaton, a deterministic finite state automaton with transition alphabet $\{\mathtt{S},\mathtt{T}\}$ (see Figure 8.1 for an S-T-automaton recognizing $\mathtt{st}_{\mathbf{d}}(L^{ex'})$).

Timed semantics and S-T-encoding

In the following we define clock and timed languages similarly to what we have done in section 8.2.2. Here we need only one clock x that remains bounded by 1. We define the clock language associated to $\mathtt{S}$ by $\mathcal{L}(\mathtt{S}) = \{x \xrightarrow{(t,\mathtt{S})} x+t \mid x \in [0,1], t \in [0,1-x]\}$ and the clock language associated to $\mathtt{T}$ by $\mathcal{L}(\mathtt{T}) = \{x \xrightarrow{(t,\mathtt{T})} t \mid x \in [0,1], t \in [0,1-x]\}$. Let $L'' \subseteq \{\mathtt{S},\mathtt{T}\}^*$ we denote by $L''(x)$ the timed language starting from x: $L''(x) = \{(\vec{t},w) \mid \exists y \in [0,1], x \xrightarrow{(\vec{t},w)} y \in \mathcal{L}(w), w \in L''\}$. The *timed semantics* of $L'' \subseteq \{\mathtt{S},\mathtt{T}\}^*$ is $L''(0)$.

The S-T-encodings yields a natural volume preserving transformation between timed languages:

Proposition 53. *Let $L' \subseteq \{\mathbf{a},\mathbf{d}\}^*$, $l \in \{\mathbf{a},\mathbf{d}\}$, $\mathbb{L}'$ be the timed semantics of L' and $\mathbb{L}''$ be the timed semantics of $\mathtt{st}_l(L')$ then the function $(\vec{t},w) \mapsto (\vec{t},\mathtt{st}_l^{-1}(w))$ is a volume preserving transformation from $\mathbb{L}''_n$ to $\mathbb{L}'_n$.*

Using notation and results of Theorem 35 and Proposition 53 we get a volume preserving transformation from $\mathbb{L}''_n$ to $\mathcal{O}_n(L)$.

Theorem 36. *The function $(\vec{t},w) \mapsto \phi_{\mathtt{st}_l^{-1}(w)}(\vec{t})$ is a volume preserving transformation from $\mathbb{L}''_n$ to $\mathcal{O}_n(L)$ computable in linear time. In particular*

$$\mathtt{Vol}(\mathbb{L}''_n) = \frac{|\mathtt{sg}^{-1}(L_{n-1})|}{n!} \text{ for } n \geq 1, \text{ and } VGF(\mathbb{L}'')(z) = EGF(\mathtt{sg}^{-1}(L))(z)$$

Thus to solve Problem 1 it suffices to characterize the VGF of an S-T-automaton.

8.3 Solving the two problems

8.3.1 Characterization of the VGF of an S-T-automaton.

In this section we characterize precisely the VGF of the timed language recognized by an S-T-automaton. This solves Problem 1.

We have defined just above timed language $L''(x)$ parametrized by an initial clock vector x. Given an S-T-automaton, we can also consider the intial state p as a parameter and write Kleene like systems of equations on parametric language $L_p(x)$ (similarly to [ABDP12]). More precisely, let $\mathcal{A} = \{\{\mathtt{S},\mathtt{T}\}, Q, i, F, \delta\}$ be an S-T-automaton. To every state $p \in Q$ we denote by $L_p \subseteq \{\mathtt{S},\mathtt{T}\}^*$ the language starting from p i.e. recognized by $\mathcal{A}_p =_{\text{def}} \{\{\mathtt{S},\mathtt{T}\}, Q, p, F, \delta\}$. We adopt the convention that $L_{\delta(p,l)}$ is empty when $\delta(p,l)$ is undefined and the corresponding generating function is null. Then for every $p \in Q$, we have a parametric language equation:

$$L_p(x) = \left[\cup_{t\leq 1-x}(t,\mathtt{S})L_{\delta(p,\mathtt{S})}(x+t)\right] \cup \left[\cup_{t\leq 1-x}(t,\mathtt{T})L_{\delta(p,\mathtt{T})}(t)\right] \cup (\epsilon \text{ if } p \in F) \tag{8.7}$$

Passing to volume generating functions $f_p(x,z) =_{\text{def}} VGF(L_p(x))(z)$ (as in [ABDP12]) we get:

$$f_p(x,z) = z\int_x^1 f_{\delta(p,\mathtt{S})}(s,z)ds + z\int_0^{1-x} f_{\delta(p,\mathtt{T})}(t,z)dt + (1 \text{ if } p \in F) \tag{8.8}$$

In matrix notation:

$$\vec{f}(x,z) = zM_{\mathtt{S}}\int_x^1 \vec{f}(s,z)ds + zM_{\mathtt{T}}\int_0^{1-x} \vec{f}(t,z)dt + \vec{F} \tag{8.9}$$

where $\vec{f}(x,z), \int_x^1 \vec{f}(s,z)ds$ and $\int_0^{1-x} \vec{f}(t,z)dt$ are the column vectors whose coordinates are respectively the $f_p(x,z), \int_x^1 \vec{f_p}(s,z)ds$ and $\int_0^{1-x} \vec{f_p}(t,z)dt$ for $p \in Q$. The p^{th} coordinate of the column vector $\vec{F}$ is 1 if $p \in F$ and 0 otherwise. The $Q \times Q$-matrices $M_{\mathtt{S}}$ and $M_{\mathtt{T}}$ are the adjacency matrices corresponding to letter S and T i.e. for $l \in \{\mathtt{S},\mathtt{T}\}$, $M_l(p,q) = 1$ if $\delta(p,l) = q$ and 0 otherwise.

The equation (8.9) is equivalent to the differential equation:

$$\frac{\partial}{\partial x}\vec{f}(x,z) = -zM_{\mathtt{S}}\vec{f}(x,z) - zM_{\mathtt{T}}\vec{f}(1-x,z) \tag{8.10}$$

with boundary condition

$$\vec{f}(1,z) = \vec{F}. \tag{8.11}$$

The equation (8.10) is equivalent to the following linear homogeneous system of ordinary differential equations with constant coefficients:

$$\frac{\partial}{\partial x}\begin{pmatrix} \vec{f}(x,z) \\ \vec{f}(1-x,z) \end{pmatrix} = z\begin{pmatrix} -M_{\mathtt{S}} & -M_{\mathtt{T}} \\ M_{\mathtt{T}} & M_{\mathtt{S}} \end{pmatrix}\begin{pmatrix} \vec{f}(x,z) \\ \vec{f}(1-x,z) \end{pmatrix}. \tag{8.12}$$

Algorithm 1 Computation of the generating function

1: Compute an S-T-automaton $\mathcal{A}$ for an extension of L and its corresponding adjacency matrices $M_{\mathtt{T}}$ and $M_{\mathtt{S}}$;
2: Compute $\begin{pmatrix} A_1(z) & A_2(z) \\ A_3(z) & A_4(z) \end{pmatrix} =_{\text{def}} \exp\left[z \begin{pmatrix} -M_{\mathtt{S}} & -M_{\mathtt{T}} \\ M_{\mathtt{T}} & M_{\mathtt{S}} \end{pmatrix}\right]$;
3: Compute $\vec{f}(0,z) = [A_1(z)]^{-1}[I - A_2(z)]\vec{F}$ (or $\vec{f}(0,z) = [I - A_3(z)]^{-1}A_4(z)\vec{F}$);
4: **return** the component of $\vec{f}(0,z)$ corresponding to the initial state of $\mathcal{A}$.

whose solution is of the form

$$\begin{pmatrix} \vec{f}(x,z) \\ \vec{f}(1-x,z) \end{pmatrix} = \exp\left[xz \begin{pmatrix} -M_{\mathtt{S}} & -M_{\mathtt{T}} \\ M_{\mathtt{T}} & M_{\mathtt{S}} \end{pmatrix}\right] \begin{pmatrix} \vec{f}(0,z) \\ \vec{f}(1,z) \end{pmatrix} \tag{8.13}$$

Taking $x = 1$ in (8.13) and using the boundary condition (8.11) we obtain:

$$\begin{array}{rcl} \vec{F} & = & A_1(z)\vec{f}(0,z) + A_2(z)\vec{F} \\ \vec{f}(0,z) & = & A_3(z)\vec{f}(0,z) + A_4(z)\vec{F} \end{array} \tag{8.14}$$

where $\begin{pmatrix} A_1(z) & A_2(z) \\ A_3(z) & A_4(z) \end{pmatrix} = \exp\left[z \begin{pmatrix} -M_{\mathtt{S}} & -M_{\mathtt{T}} \\ M_{\mathtt{T}} & M_{\mathtt{S}} \end{pmatrix}\right]$. In particular when $z = 0$, $A_1(0) = I - A_3(0) = I$ and thus the two continuous functions $z \mapsto \det A_1(z)$ and $z \mapsto \det(I - A_3(z))$ are positive in a neighbourhood of 0. We deduce that the inverses of the matrices $A_1(z)$ and $I - A_3(z)$ are well defined in a neighbourhood of 0 and thus both rows of (8.14) permit to express $\vec{f}(0,z)$ with respect to $\vec{F}$:

$$\begin{array}{rcl} \vec{f}(0,z) = & [A_1(z)]^{-1}[I - A_2(z)]\vec{F} \\ \vec{f}(0,z) = & [I - A_3(z)]^{-1}A_4(z)\vec{F} \end{array} \tag{8.15}$$

Finally the coordinate of the column vector $\vec{f}(0,z)$ associated to the initial state gives the expected VGF. To sum up we have:

Theorem 37. *Given a regular language $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$, one can compute the exponential generating function $EGF(\mathtt{sg}^{-1}(L))(z)$ using Algorithm 1.*

Some comments about the algorithm. In line 1, several choices are left to the user: the prolongation L' of the language L, the type of the S-T-encoding and the automaton that realizes the S-T-encoding. These choices should be made such that the output automaton has a minimal number of states or more generally such that the matrices $M_{\mathtt{T}}$ and $M_{\mathtt{S}}$ are the simplest possibles. Exponentiation of matrices is implemented in most of computer algebra systems.

8.3.2 An algorithm for Problem 2

Now we can solve Problem 2 using a uniform sampler of timed words (Algorithm 2), the volume preserving transformation of Theorem 36 and a sorting algorithm.

Theorem 38. *Let $L \subseteq \{\mathbf{a}, \mathbf{d}\}^*$ and $\mathbb{L}''$ be the timed semantics of a* S-T*-encoding of type l (for some $l \in \{\mathbf{a}, \mathbf{d}\}$) of a prolongation of L. The following algorithm permits to achieve a uniform sampling of permutation in $\mathtt{sg}^{-1}(L_{n-1})$. i.e. For $\sigma \in \mathtt{sg}_n^{-1}(L)$, the probability that the permutation σ is returned is $1/|\mathtt{sg}^{-1}(L_{n-1})|$.*

1. *Choose uniformly an n-length timed word $(\vec{t}, w) \in \mathbb{L}''_n$ using Algorithm 2;*
2. *Return $\Pi(\phi_{\mathtt{st}_l^{-1}(w)}(\vec{t}))$.*

Proof. For all $\sigma \in \mathtt{sg}_n^{-1}(L)$, the probability $p(\sigma)$ that the output is σ is the probability to choose a timed word $(\vec{t}, w)$ such that $\Pi[\phi_{\mathtt{st}^{-1}(w)}(\vec{t})] = \sigma$. Since the timed words are uniformly sampled this probability is equal to $\mathtt{Vol}(\{(\vec{t}, w) \mid \Pi[\phi_w(\vec{t})] = \sigma\})/\mathtt{Vol}(\mathbb{L}''_n)$ which is equal to $\mathtt{Vol}(\{\vec{\nu} \mid \Pi(\vec{\nu}) = \sigma\})/\mathtt{Vol}(\mathbb{L}''_n)$ since the mapping $(\vec{t}, w) \mapsto \phi_{\mathtt{st}^{-1}(w)}(\vec{t})$ is a volume preserving transformation. The numerator is the volume of the order simplex associated to σ which is $\mathtt{Vol}(\mathcal{O}(\sigma)) = 1/n!$; the denominator $\mathtt{Vol}(\mathbb{L}''_n)$ is $|\mathtt{sg}^{-1}(L_{n-1})|/n!$ by virtue of Theorem 36. We get the expected result $p(\sigma) = (1/n!)/(|\mathtt{sg}^{-1}(L_{n-1})|/n!) = 1/|\mathtt{sg}^{-1}(L_{n-1})|$. □

Uniform sampling of timed words.

Recursive formulae (8.16) and (8.17) below are freely inspired by those of [AD09a] (and by those of the previous chapter of the thesis). They are the key tools to design a uniform sampler of timed word. This algorithm is a lifting from the discrete case of the so-called recursive method (see [BG12, FZVC94]). For all $q \in Q$, $n \in \mathbb{N}$ and $x \in [0, 1]$ we denote by $L_{q,n}(x)$ the language $L_q(x)$ restricted to n-length timed words. The languages $L_{q,n}(x)$ can be recursively defined as follows: $L_{q,0}(x) = \epsilon$ if $q \in F$ and $L_{q,0} = \emptyset$ otherwise;

$$L_{q,n+1}(x) = \left[\cup_{t \leq 1-x}(t, \mathtt{S}) L_{\delta(q,\mathtt{S}),n}(x+t)\right] \cup \left[\cup_{t \leq 1-x}(t, \mathtt{T}) L_{\delta(q,\mathtt{T}),n}(t)\right]. \tag{8.16}$$

For $q \in Q$ and $n \geq 0$, we denote by $v_{q,n}$ the function $x \mapsto \mathtt{Vol}[L_{q,n}(x)]$ from $[0, 1]$ to $\mathbb{R}^+$. Each function $v_{q,n}$ is a polynomial of a degree less or equal to n that can be computed recursively using the recurrent formula: $v_{q,0}(x) = 1_{q \in F}$ and

$$v_{q,n+1}(x) = \int_x^1 v_{\delta(q,\mathtt{S}),n}(y)dy + \int_0^{1-x} v_{\delta(q,\mathtt{T}),n}(y)dy. \tag{8.17}$$

The polynomials $v_{q,n}(x)$ play a key role for the uniform sampler, they permit also to retrieve directly the terms of the wanted VGF: $\mathtt{Vol}(\mathbb{L}''_n) = v_{q_0,n}(0)$ where q_0 is the initial state of the S-T automaton.

Theorem 39. *Algorithm 2 is a uniform sampler of timed words of $\mathbb{L}''_n$ i.e. for every volume measurable subset $A \subseteq \mathbb{L}''_n$, the probability that the returned timed word belongs to A is $\mathtt{Vol}(A)/\mathtt{Vol}(\mathbb{L}''_n)$.*

Proof. One can first check that for all $k \in [n]$, $(q_{k-1}, x_{k-1}) \xrightarrow{(t_k, w_k)} (q_k, x_k) \in \mathcal{L}(w_k)$ and that $w_1 \cdots w_n \in L''$.

Algorithm 2 Recursive uniform sampler of timed words

$x_0 \leftarrow 0$; $q_0 \leftarrow$ initial state;
for $k = 1$ to n **do**
  Compute $m_k = v_{q_{k-1},n-(k-1)}(x_{k-1})$ and $p_{\mathtt{S}} = \int_{x_{k-1}}^{1} v_{\delta(q_{k-1},\mathtt{S}),n-k}(y)dy/m_k$;
  $b \leftarrow \mathtt{BERNOULLI}(p_{\mathtt{S}})$; (return 1 with probability $p_{\mathtt{S}}$ and 0 otherwise)
  if $b = 1$ **then**
    $w_k \leftarrow \mathtt{S}$; $q_k \leftarrow \delta(q_{k-1}, \mathtt{S})$;
    $r \leftarrow \mathtt{RAND}([0,1])$; (return a number uniformly sampled in $[0,1]$)
    $t_k \leftarrow$ the unique solution in $[0, 1 - x_{k-1}]$ of $\frac{1}{m_k p_{\mathtt{S}}} \int_{x_{k-1}}^{x_{k-1}+t_k} v_{q_k,n-k}(y)dy - r = 0$;
    $x_k \leftarrow x_{k-1} + t_k$;
  else
    $w_k \leftarrow \mathtt{T}$; $q_k \leftarrow \delta(q_{k-1}, \mathtt{T})$;
    $r \leftarrow \mathtt{RAND}([0,1])$; (return a number uniformly sampled in $[0,1]$)
    $t_k \leftarrow$ the unique solution in $[0, 1 - x_{k-1}]$ of $\frac{1}{m_k(1-p_{\mathtt{S}})} \int_{0}^{t_k} v_{q_k,n-k}(y)dy - r = 0$;
    $x_k \leftarrow t_k$;
  end if
end for
return $(t_1, w_1)(t_2, w_2)\dots(t_n, w_n)$

We denote by $p[(t_1, w_1)\cdots(t_n, w_n)]$ the density of probability of the timed word $(t_1, w_1)\cdots(t_n, w_n) \in \mathbb{L}''$ to be returned. The algorithm is a uniform sampler if it assign the same density of probability to every timed word of $\mathbb{L}''$ i.e. $p[(t_1, w_1)\cdots(t_n, w_n)] = 1/\mathtt{Vol}(\mathbb{L}'')$.

During the k^{th} loop, w_k and t_k are chosen, knowing q_{k-1}, x_{k-1} and the index k, according to a density of probability (implicitly defined by the algorithm) denoted by $p_k[(t_k, w_k) \mid q_{k-1}, x_{k-1}]$. The new general state (q_k, x_k) is (deterministically) defined using q_{k-1}, x_{k-1}, t_k, w_k. The following chain rule is satisfied

$$p[(t_1, w_1)\cdots(t_n, w_n)] = \prod_{k=1}^{n} p_k[(t_k, w_k) \mid q_{k-1}, x_{k-1}] \tag{8.18}$$

No it suffices to plug (8.19) proven in Lemma 48 just below in (8.18) to get the expected result:

$$p[(t_1, w_1)\cdots(t_n, w_n)] = \frac{\prod_{k=1}^{n} m_{k+1}}{\prod_{k=1}^{n} m_k} = \frac{m_{n+1}}{m_1} = \frac{v_{q_n,0}(x_n)}{v_{q_0,n}(0)} = \frac{1}{\mathtt{Vol}(\mathbb{L}''_n)}.$$

□

Lemma 48. *In Algorithm 2 during the k^{th} loop* **for** *the timed transition (t_k, w_k) is chosen knowing the current state (q_{k-1}, x_{k-1}) according to the following probability distribution function (variables of the following equation such as m_k are defined in the algorithm):*

$$p_k[(t_k, w_k) \mid q_{k-1}, x_{k-1}] = \frac{m_{k+1}}{m_k} = \frac{v_{q_k,n-k}(x_k)}{v_{q_{k-1},n-(k-1)}(x_{k-1})}. \tag{8.19}$$

Proof. The choice of (t_k, w_k) is done in two steps: first w_k is chosen (and thus $q_k = \delta(q_{k-1}, w_k)$) and then t_k. We write this

$$p_k[(t_k, w_k) \mid q_{k-1}, x_{k-1}] = p_k[w_k \mid q_{k-1}, x_{k-1}] p_k[t_k \mid q_k, x_{k-1}] \tag{8.20}$$

Remark that $b = 1$ iff $w_k = \mathtt{S}$ and thus $p_k[\mathtt{S} \mid q_{k-1}, x_{k-1}] = p_{\mathtt{S}}$ (the probability that 1 is returned in line 4) and $p_k[\mathtt{T} \mid q_{k-1}, x_{k-1}] = 1 - p_{\mathtt{S}}$ otherwise.

In both cases ($b = 0$ or 1) the delay t_k is sampled using the so-called inverse transform sampling. This method states that to sample a random variable according to a probability density function (PDF) $p(t)$ (here $p(t) = p_k[t \mid q_k, x_{k-1}]$) it suffices to uniformly sample a random number in $[0, 1]$ and define t such that $\int_0^t p(t')dt' = r$. The latter integral is known as the cumulative density function[4] (CDF) associated to p.

- When $b = 1$ (and thus $w_k = \mathtt{S}$), the CDF used in the algorithm is

  $$t \mapsto \frac{1}{m_k p_{\mathtt{S}}} \int_0^t v_{q_k, n-k}(x_{k-1} + t')dt'.$$

  Its corresponding PDF is

  $$p_k[t_k \mid q_k, x_{k-1}] = \frac{1}{m_k p_{\mathtt{S}}} v_{q_k, n-k}(x_{k-1} + t_k) = \frac{m_{k+1}}{m_k p_{\mathtt{S}}}.$$

  Plugging this in (8.20) we get the expected result (8.19).

- When $b = 0$ (and thus $w_k = \mathtt{T}$), a similar reasoning permits to prove (8.19) which is then true in both cases.

□

Remark 4. *One can remark that the probability depends on the index k of the loop which is different[5] from stochastic processes over runs of Chapter 4.*

Some comments about the algorithm. Algorithm 2 requires a precomputation of all functions $v_{q,k}$ for $q \in Q$ and $k \leq n$ done by Algorithm 3 below (see also Proposition 54 for the complexity). The expressions in lines 8 and 13 are polynomials increasing in $[x, 1]$ (the derivative is the integrand which is positive on $(x, 1)$). Finding the root of such a polynomial can be done numerically and efficiently with a controlled error using a numerical scheme such as the Newton's method.

Proposition 54. *Algorithm 3 has space and time complexity $O(|Q|n^2)$. Its bit space complexity is $O(|Q|n^3)$.*

Proof. The polynomial $v_{q,m}$ is of degree m, it has $O(m)$ coefficients. Therefore the time and space complexity are $O(\sum_{m=1}^{n} |Q|m) = O(|Q|n^2)$.

Magnitudes of coefficients of $v_{q,m}$ behave like $2^{m\mathcal{H}}$ where $\mathcal{H}$ is the entropy of the timed language (see [AD09a]) and thus one needs $O(m)$ bits to store them. This explains why an extra factor n appears when dealing with bit space complexity. □

[4] Its inverse (t function of r) is known as the quantile function.

[5] Such a property is sometimes called time-inhomogeneous while the SPOR of Chapter 4 are time-homogeneous.

Algorithm 3 Preprocessing for Algorithm 2

```
for p ∈ Q do
    define v_{p,0}(x) = 1_{p∈F}.
    for k = 1 to n do
        compute v_{p,k}(x) using (8.17).
    end for
end for
```

8.4 Examples

In section 8.4.1 we show how Algorithm 1 applies to the classical example of alternating permutations. In section 8.4.2 we apply this algorithm to what we call up-up-down-down permutations. In section 8.4.3 we treat the running example given in section 8.1.

8.4.1 The alternating permutations

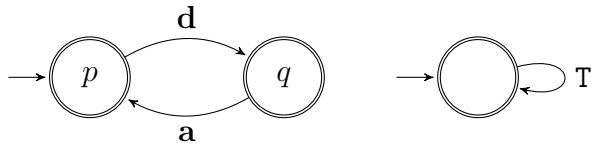

Figure 8.2: An automaton for $(\mathbf{da})^*(\varepsilon + \mathbf{d})$ and its S-T encoding of type $\mathbf{d}$

The class of alternating permutation is[6] $\mathtt{Alt} = \mathfrak{S}_0 \cup \mathtt{sg}^{-1}[(\mathbf{da})^*(\varepsilon + \mathbf{d})]$. It is well known since the 19^{th} century and the work of Désiré André that

$$EGF(\mathtt{Alt})(z) = \tan(z) + \sec(z) \quad (\text{where } \sec(z) = 1/\cos(z)).$$

Several different proofs of this results can be found in [Sta10]. Here we give a novel proof based on the application of Algorithm 1 on $(\mathbf{da})^*(\varepsilon + \mathbf{d})$.

A prolongation of $(\mathbf{da})^*(\varepsilon + \mathbf{d})$ is $(\mathbf{da})^*(\mathbf{d} + \mathbf{da})$. We add ε to the language to add 1 to its VGF, indeed

$$EGF(\mathtt{Alt})(z) = 1 + VGF[(\mathbf{da})^*(\mathbf{d} + \mathbf{da})](z) = VGF[(\mathbf{da})^*(\varepsilon + \mathbf{d})](z)$$

The S-T encoding of type $\mathbf{a}$ of $(\mathbf{da})^*(\varepsilon + \mathbf{d})$ is just $\mathtt{S}^*$ which is recognized by the one loop automaton depicted in the right of Figure 8.2. Thus $M_{\mathtt{S}} = (1)$, $M_{\mathtt{T}} = (0)$ and we must compute $\exp(zM) = \sum_{n\in\mathbb{N}} z^n M^n/n!$ with $M = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$.

Computation of $\exp(zM)$ is easy since M is unipotent and thus its sequence of power M^k is periodic: $M^0 = I_2$, $M^1 = M$, $M^2 = -I_2$, $M^3 = -M$, $M^3 = I_2$, $M^4 = M$, ...

[6]The unique permutation on the empty set has no signature and thus $\mathfrak{S}_0 \not\subseteq \mathtt{sg}^{-1}(L)$ for any language L of signature.

Then for all $k \geq 0$:

$$M^{2k} = \begin{pmatrix} (-1)^k & 0 \\ 0 & (-1)^k \end{pmatrix}; \quad M^{2k+1} = \begin{pmatrix} 0 & (-1)^{2k} \\ (-1)^{2k+1} & 0 \end{pmatrix}$$

Hence $\exp(zM) = \sum_{n\in\mathbb{N}} z^n M^n/n! = \begin{pmatrix} \cos(z) & -\sin(z) \\ \sin(z) & \cos(z) \end{pmatrix}$.

By definition $A_1(z) = \cos(z)$, $A_2(z) = -\sin(z)$. We can conclude:

$$EGF(\texttt{Alt})(z) = A_1(z)^{-1}(1 - A_2(z)) = \frac{1}{\cos(z)} + \tan(z).$$

8.4.2 The up-up-down-down permutations

Here we compute the exponential generating function of the class of up-up-down-down permutations given as running example of the chapter. Recall that the corresponding regular language is $L^{ex} = (\mathbf{aadd})^*(\mathbf{aa} + \varepsilon)$, one of its extension is $L^{ex'} = (\mathbf{aadd})^*(\mathbf{aad} + \mathbf{a})$ and the $\texttt{S}$-$\texttt{T}$-encoding of type $\mathbf{d}$ of this latter language is $\texttt{st}_{\mathbf{d}}(L') = (\texttt{TS})^*\texttt{T}$. These languages are recognized by automata depicted in Figure 8.1. The adjacency matrices of the third automaton are $M_{\texttt{S}} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$, $M_{\texttt{T}} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and the row vector of final state is $\vec{F} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. Let $M = \begin{pmatrix} -M_{\texttt{S}} & -M_{\texttt{T}} \\ M_{\texttt{T}} & M_{\texttt{S}} \end{pmatrix}$. Again the computation of $\exp(zM)$ is easy since M is unipotent[7]:

$$M = \begin{pmatrix} 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}; M^2 = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}; M^3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \end{pmatrix}$$

and $M^4 = I_4$.

Hence if we denote by $f_i(z) = \sum_{n=0}^{+\infty} z^{4n+i}/(4n+i)!$ for $i \in \{0, 1, 2, 3\}$ we have:

$$\exp zM = f_0(z)I + f_1(z)M + f_2(z)M^2 + f_3(z)M^3 \text{ and}$$

$$A_1(z) = \begin{pmatrix} f_0(z) & -f_3(z) \\ -f_1(z) & f_0(z) \end{pmatrix}; A_2(z) = \begin{pmatrix} f_2(z) & -f_1(z) \\ f_3(z) & f_2(z) \end{pmatrix}.$$

The function f_i can be expressed with trigonometric and hyperbolic functions:

$$\begin{aligned} f_0(z) &= [\cosh(z) + \cos(z)]/2; \\ f_1(z) &= [\sinh(z) + \sin(z)]/2; \\ f_2(z) &= [\cosh(z) - \cos(z)]/2; \\ f_3(z) &= [\sinh(z) - \sin(z)]/2. \end{aligned}$$

[7]This is in fact the case for all cyclic automata.

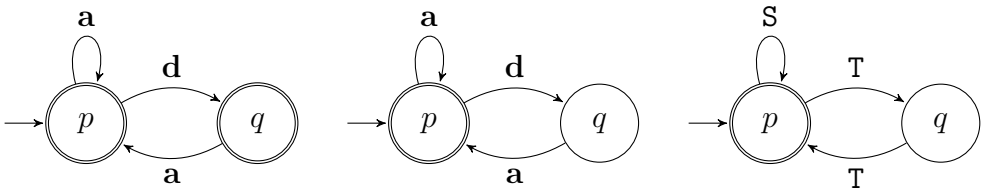

Figure 8.3: From left to right automata for L^{ex_3}, $L^{ex'_3} = \{\varepsilon\} \cup L^{ex_3}.\{\mathbf{a}\}$ and $\mathtt{st}_{\mathbf{a}}(L^{ex'_3})$

It holds that

$$[I_2 - A_2(z)]\vec{F} = \begin{pmatrix} f_1(z) \\ 1 - f_2(z) \end{pmatrix}$$

and thus

$$\begin{pmatrix} f_p(z) \\ f_q(z) \end{pmatrix} = \begin{pmatrix} f_0(z) & -f_3(z) \\ -f_1(z) & f_0(z) \end{pmatrix}^{-1} \begin{pmatrix} f_1(z) \\ 1 - f_2(z) \end{pmatrix}.$$

Using Cramer formula we get $f_p(z) = [f_1(z)f_0(z) + f_3(z)(1 - f_2(z))]/[f_0^2(z) + f_1(z)f_3(z)]$. After straightforward simplifications we obtain the wanted result:

$$f(z) = f_p(z) = \frac{\sinh(z) - \sin(z) + \sin(z)\cosh(z) + \sinh(z)\cos(z)}{1 + \cos(z)\cosh(z)}.$$

8.4.3 Permutations without two consecutive descents

Consider the class C^{ex_3} of permutations without two consecutive descents. This class has already been studied and its EGF computed. References and many details can be found in the On-Line Encyclopedia of Integer Sequences (OEIS), sequence A049774. In particular the following EGF is given:

$$EGF(C^{ex_3})(z) = \frac{\sqrt{3}e^{z/2}}{\sqrt{3}\cos\left(\frac{\sqrt{3}}{2}z\right) - \sin\left(\frac{\sqrt{3}}{2}z\right)}.$$

We give an alternative proof of this result based on the method developed in this chapter. The class C^{ex_3} can be described in terms of regular languages:

$$C^{ex_3} = \mathfrak{S}_0 \cup \mathtt{sg}^{-1}[(\mathbf{a} + \mathbf{da})^*(\varepsilon + \mathbf{d})].$$

A prolongation of $(\mathbf{a}+\mathbf{da})^*(\varepsilon+\mathbf{d})$ is $(\mathbf{a}+\mathbf{da})^*\mathbf{a}$. As for alternating permutations we add the word ε to this language to add 1 to the final generating function, thus we get the language $(\mathbf{a}+\mathbf{da})^*$ recognized by the automaton depicted in the middle of Figure 8.3. Its S-T encoding of type $\mathbf{a}$ is $(\mathtt{S}+\mathtt{TT})^*$ which is recognized by the automaton depicted in the right of Figure 8.3. Its adjacency matrices are $M_{\mathtt{S}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, $M_{\mathtt{T}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ and the row vector of final state

is $\vec{F} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Let $M = \begin{pmatrix} -M_{\mathtt{S}} & -M_{\mathtt{T}} \\ M_{\mathtt{T}} & M_{\mathtt{S}} \end{pmatrix}$. We will solve directly the differential equation (8.10) with boundary condition (8.11), i.e. the system

$$\frac{\partial f_p}{\partial x}(x,z) = -zf_p(x,z)dy - zf_q(1-x,z)dy; \tag{8.21}$$

$$\frac{\partial f_q}{\partial x}(x,z) = -zf_p(1-x,z). \tag{8.22}$$

with boundary conditions $f_p(1,z) = 1; f_q(1,z) = 0$ Equation (8.21) taken at $x = 1$ ensures that $\frac{\partial f_p}{\partial x}(1,z) = -zf_p(0,z) - zf_q(1,z) = -zf_p(0,z)$. Thus we have the boundary conditions

$$f_p(1,z) = 1; \tag{8.23}$$

$$\frac{\partial f_p}{\partial x}(1,z) = -zf_p(0,z). \tag{8.24}$$

Differentiating (8.21) and replacing $\frac{\partial f_q}{\partial x}(1-x,z)$ using (8.22) we get:

$$\frac{\partial^2 f_p}{\partial x^2}(x,z) = -z\frac{\partial f_p}{\partial x} - z^2 f_p(x,z); \tag{8.25}$$

Solutions are of the form: $f_p(x,z) = e^{-zx/2}\left[a(z)\cos\left(\frac{\sqrt{3}}{2}zx\right) + b(z)\sin\left(\frac{\sqrt{3}}{2}zx\right)\right]$ with $a(z)$ and $b(z)$ to be determined using boundary conditions (8.23) and (8.24) i.e. $a(z)$ and $b(z)$ should satisfy:

$$\begin{array}{cccc} \cos\left(\frac{\sqrt{3}}{2}z\right) & a(z)+ & \sin\left(\frac{\sqrt{3}}{2}z\right) & b(z) = e^{z/2}; \\ & a(z)+ & \sqrt{3} & b(z) = 0. \end{array}$$

Solving this system we obtained the expected EGF:

$$EGF(C^{ex_3})(z) = f_p(0,z) = a(z) = \frac{\sqrt{3}e^{z/2}}{\sqrt{3}\cos\left(\frac{\sqrt{3}}{2}z\right) - \sin\left(\frac{\sqrt{3}}{2}z\right)}.$$

8.5 Conclusion and perspectives

We have stated and solved the problems of counting and uniform sampling of permutations with signature in a given regular language of signatures. The timed semantics of such a language is a particular case of regular timed languages (i.e. recognized by timed automata [AD94]). However, with the approach used, timed languages can be defined from any kind of languages of signatures. A challenging task for us is to treat the case of context free languages of signatures. For this we should use as in Chapter 7 and in [ABD13] (see also the kernels of Chapter 4), volume of languages parametrized both by a starting and an ending state.

Volumes and languages parametrized both by a starting and an ending states would also be useful to gain a linear factor for the time and space complexity of the preprocessing stage

(Algorithm 3). Indeed such parametrized volume are needed to adapt the divide and conquer algorithm of [BG12].

Our work can also benefit timed automata research. Indeed, we have proposed a uniform sampler for a particular class of timed languages. An ongoing work is to adapt this algorithm to all deterministic timed automata with bounded clocks using recursive equations of [AD09a]. A uniform sampler of timed word would be useful to solve the proportional model checking problem introduced in Chapter 4

It would be interesting to see whether the sequence of probability density functions $(p_k)_{k\in\mathbb{N}}$ defined in Lemma 48 converges (in a sense to be clarified) toward the conditional PDF of p^* of the maximal entropy SPOR studied in Chapter 4.

A toy implementation of the algorithms is available on-line:
`http://www.liafa.univ-paris-diderot.fr/~nbasset/sage/sage.htm`.

Chapter 9

Conclusion and perspectives

In this thesis we applied entropy-based studies of timed languages in several research areas and these methods gave interesting results. Indeed they allow

- to distinguish between essentially Zeno and essentially non-Zeno TA;
- to develop enumerative combinatorics of timed languages;
- to initiate constrained channel information transmission theory;
- to contribute to simulation of timed automata and uniform generation of timed words;
- to define a natural stochastic process for a timed automaton;
- to formulate timed automata theory in terms of symbolic dynamics;
- and even to have a general method for uniform random generation of permutations.

This confirms that an information-based approach to timed automata is very useful.

It would be also interesting to extend the methods presented here to timed automata with weights, in the sense that each transition could have both a time duration and a cost. Generating functions would again be a helpful tool to handle this more general setting.

We believe that these methods can/should be extended to other models from computer science such as Hybrid automata and their languages, data languages, programs.

In a shorter perspective we would like to explore the two followings topics:

Uniform generation of timed words and permutations.

Let us compare the two kinds of random generations of timed words that appeared in the thesis:

- In Chapter 4, the maximal entropy SPOR permits to perform a **quasi**-uniform random generation of runs in **linear** time. The timed region graphs considered are quite **general**. This permits to generate timed words of a quite general timed automaton in

linear time. However the question of how to compute the maximal entropy SPOR has not been addressed yet.

- In Chapter 8, there is an **exact** uniform sampling of timed word in **quadratic** time (and even **cubic** time for the bit complexity) for a very **restrictive** subclass of timed automata. However, this sub-class is general enough to solve the problem of random generation of permutations with signature in a given regular language.

It remains some work to marry the strength of both approaches while eliminating their weakness. In fact the method of Chapter 8 can easily be adapted to the whole class of bounded deterministic timed automata using the recursive equation on volume functions of [AD09a] recalled in (2.3),(2.4). Nevertheless it remains slower than the method of Chapter 4. Unfortunately, this latter method does not yield a truly uniform random sampling. In [Mar12] the author uses a stochastic process very similar to ours for the particular problem of generating alternating permutations. He corrects the non uniformity of the process by an ad hoc method. Some ideas of this latter work may be helpful to improve uniformity of our process Y^*.

For the particular case of timed automata considered in Chapter 8, we are convinced that symbolic computations of objects $\rho,\ v,\ w$ defining the maximal entropy SPOR Y^* can easily be done and thus that quasi-uniform random generation of permutations can be performed in linear time.

Empirical entropy

In [Sha48] and [Sha51], Shannon estimated the entropy of the printed English by assuming (and motivating) that English texts were produced as with an irreducible Markov chain (see also [Lot05]).

In a verification context, a language can represents the observable behaviours of a system. The entropy measures the complexity of a language. The higher the entropy is, the more complex the language is.

It is a challenging task for us to estimate empirically the entropy from samples of a timed language. An empirical estimation of the entropy can be useful for monitoring a system: A fall of the entropy can traduce a breakdown of a part of the system observed. In a different context, if we model a spoken language by a timed language, then we could estimate for instance the entropy of spoken English as Shannon did for printed English.

Bibliography

[ABB+12] Eugene Asarin, Nicolas Basset, Marie-Pierre Béal, Aldric Degorre, and Dominique Perrin. Toward a timed theory of channel coding. In Jurdzinski and Nickovic [JN12], pages 27–42.

[ABD13] Eugene Asarin, Nicolas Basset, and Aldric Degorre. Spectral gap in timed automata. In Víctor A. Braberman and Laurent Fribourg, editors, *FORMATS*, volume 8053 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 2013.

[ABDP12] Eugene Asarin, Nicolas Basset, Aldric Degorre, and Dominique Perrin. Generating functions of timed languages. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *MFCS*, volume 7464 of *Lecture Notes in Computer Science*, pages 124–135. Springer, 2012.

[AC88] Paul H. Algoet and Thomas M. Cover. A sandwich proof of the Shannon-McMillan-Breiman theorem. *The annals of probability*, 16(2):899–909, 1988.

[ACD91] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Model-checking for probabilistic real-time systems (extended abstract). In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *ICALP*, volume 510 of *Lecture Notes in Computer Science*, pages 115–126. Springer, 1991.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[AD09a] Eugene Asarin and Aldric Degorre. Volume and entropy of regular timed languages: Analytic approach. In Joël Ouaknine and Frits W. Vaandrager, editors, *FORMATS*, volume 5813 of *Lecture Notes in Computer Science*, pages 13–27. Springer, 2009.

[AD09b] Eugene Asarin and Aldric Degorre. Volume and entropy of regular timed languages: Discretization approach. In Mario Bravetti and Gianluigi Zavattaro, editors, *CONCUR*, volume 5710 of *Lecture Notes in Computer Science*, pages 69–83. Springer, 2009.

[AD10] Eugene Asarin and Aldric Degorre. Two size measures for timed languages. In Kamal Lodaya and Meena Mahajan, editors, *FSTTCS*, volume 8 of *LIPIcs*, pages 376–387. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.

[AKY10] Parosh Aziz Abdulla, Pavel Krcál, and Wang Yi. Sampled semantics of timed automata. *Logical Methods in Computer Science*, 6(3), 2010.

[AMP98] Eugene Asarin, Oded Maler, and Amir Pnueli. On discretization of delays in timed automata and digital circuits. In Davide Sangiorgi and Robert de Simone, editors, *CONCUR*, volume 1466 of *Lecture Notes in Computer Science*, pages 470–484. Springer, 1998.

[BA07] Mikhail Bernadsky and Rajeev Alur. Symbolic analysis for gsmp models with one stateful clock. In Alberto Bemporad, Antonio Bicchi, and Giorgio C. Buttazzo, editors, *HSCC*, volume 4416 of *Lecture Notes in Computer Science*, pages 90–103. Springer, 2007.

[BA11] Nicolas Basset and Eugene Asarin. Thin and thick timed regular languages. In Fahrenberg and Tripakis [FT11], pages 113–128.

[BAM06] Sebastian Burckhardt, Rajeev Alur, and Milo M. K. Martin. Bounded model checking of concurrent data types on relaxed memory models: A case study. In Thomas Ball and Robert B. Jones, editors, *CAV*, volume 4144 of *Lecture Notes in Computer Science*, pages 489–502. Springer, 2006.

[Bas13a] Nicolas Basset. Counting and generating permutations using timed languages (long version). 2013.

[Bas13b] Nicolas Basset. A maximal entropy stochastic process for a timed automaton. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *ICALP (2)*, volume 7966 of *Lecture Notes in Computer Science*, pages 61–73. Springer, 2013.

[Basar] Nicolas Basset. Counting and generating permutations using timed languages. volume 8392 of *Lecture Notes in Computer Science*. Springer, 2014 (to appear).

[BBB+07] Christel Baier, Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Marcus Größer. Probabilistic and topological semantics for timed automata. In Vikraman Arvind and Sanjiva Prasad, editors, *FSTTCS*, volume 4855 of *Lecture Notes in Computer Science*, pages 179–191. Springer, 2007.

[BBBM08] Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Quantitative model-checking of one-clock timed automata under probabilistic semantics. In *QEST*, pages 55–64. IEEE Computer Society, 2008.

[BBEP10] Marie-Pierre Béal, Jean Berstel, S. Eilers, and Dominique Perrin. Symbolic dynamics. *CoRR*, abs/1006.1265, 2010.

[BBJM12] Patricia Bouyer, Thomas Brihaye, Marcin Jurdzinski, and Quentin Menet. Almost-sure model-checking of reactive timed automata. In *QEST*, pages 138–147. IEEE Computer Society, 2012.

[BBM+10] Marie-Pierre Béal, Jean Berstel, Brian Marcus, Dominique Perrin, Christophe Reutenauer, and Paul H. Siegel. Variable-length codes and finite automata. In Subhas Chandra Misra Issac Woungang, Sudip Misra, editor, *Selected topics in information and coding theory*, chapter 14, pages 505–584. World Scientific Publishing Company, 2010.

[Bea98] Danièle Beauquier. Pumping lemmas for timed automata. In Maurice Nivat, editor, *FoSSaCS*, volume 1378 of *Lecture Notes in Computer Science*, pages 81–94. Springer, 1998.

[BG12] Olivier Bernardi and Omer Giménez. A linear algorithm for the random sampling from regular languages. *Algorithmica*, 62(1-2):130–145, 2012.

[Bil12] Patrick Billingsley. *Probability and measure*, volume 939. Wiley, 2012.

[BL12] Mikołaj Bojańczyk and Sławomir Lasota. A machine-independent characterization of timed languages. In *Automata, Languages, and Programming*, pages 92–103. Springer, 2012.

[Bla90] Richard E. Blahut. *Digital Transmission of Information*. Addison Wesley, 1990.

[BLD+05] M Beck, JA De Loera, M Develin, J Pfeifle, and RP Stanley. Coefficients and roots of ehrhart polynomials. *Contemporary Mathematics*, 374:15–36, 2005.

[Bow71] Rufus Bowen. Entropy for group endomorphisms and homogeneous spaces. *Transactions of the American Mathematical Society*, 153:401–414, 1971.

[BP02] Patricia Bouyer and Antoine Petit. A Kleene/Büchi-like theorem for clock languages. *Journal of Automata, Languages and Combinatorics*, 7(2):167–186, 2002.

[BPR09] Jean Berstel, Dominique Perrin, and Christophe Reutenauer. *Codes and Automata*, volume 129 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2009. 634 pages.

[BR07] M. Beck and S. Robins. *Computing the continuous discretely: Integer-point enumeration in polyhedra*. Springer, 2007.

[BR11] Jean Berstel and Christophe Reutenauer. *Noncommutative rational series with applications*, volume 137 of *Enc. of Math. and Appl.* Cambridge University Press, 2011.

[BS02] Michael Brin and Garrett Stuck. *Introduction to Dynamical Systems*. Cambridge University Press, New York, NY, USA, 2002.

[CJ99] Hubert Comon and Yan Jurski. Timed automata and the theory of real numbers. In Jos C. M. Baeten and Sjouke Mauw, editors, *CONCUR*, volume 1664 of *Lecture Notes in Computer Science*, pages 242–257. Springer, 1999.

[CT06] Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.

[Dim01] Catalin Dima. Real-time automata. *Journal of Automata, Languages and Combinatorics*, 6(1):3–24, 2001.

[Dim02] Catalin Dima. Computing reachability relations in timed automata. In *LICS*, pages 177–. IEEE Computer Society, 2002.

[DLL+11] Alexandre David, Kim G. Larsen, Axel Legay, Marius Mikucionis, Danny Bøgsted Poulsen, Jonas van Vliet, and Zheng Wang. Statistical model checking for networks of priced timed automata. In Fahrenberg and Tripakis [FT11], pages 80–96.

[Edg93] Gerald A. Edgar, editor. *Classics on Fractals*. Addison-Wesley, 1993.

[EJ12] Richard Ehrenborg and JiYoon Jung. Descent pattern avoidance. *Advances in Applied Mathematics*, 2012.

[Fek23] M. Fekete. Uber die verteilung der wurzeln bei gewissen algebraischen gleichungen mit ganzzahligen koeffizienten. *Mathematische Zeitschrift*, 17:228–249, 1923.

[FS09] Philippe Flajolet and Robert Sedgewick. *Analytic combinatorics*. Cambridge University Press, 2009.

[FT11] Uli Fahrenberg and Stavros Tripakis, editors. *Formal Modeling and Analysis of Timed Systems - 9th International Conference, FORMATS 2011, Aalborg, Denmark, September 21-23, 2011. Proceedings*, volume 6919 of *Lecture Notes in Computer Science*. Springer, 2011.

[FZVC94] Philippe Flajolet, Paul Zimmerman, and Bernard Van Cutsem. A calculus for the random generation of labelled combinatorial structures. *Theoretical Computer Science*, 132(1):1–35, 1994.

[GB07] Rodolfo Gómez and Howard Bowman. Efficient detection of zeno runs in timed automata. In Jean-François Raskin and P. S. Thiagarajan, editors, *FORMATS*, volume 4763 of *Lecture Notes in Computer Science*, pages 195–210. Springer, 2007.

[GHJ97] Vineet Gupta, Thomas A. Henzinger, and Radha Jagadeesan. Robust timed automata. In Oded Maler, editor, *HART*, volume 1201 of *Lecture Notes in Computer Science*, pages 331–345. Springer, 1997.

[HL12] T. Hibi and N. Li. Unimodular equivalence of order and chain polytopes. *arXiv preprint arXiv:1208.4029*, 2012.

[HMP92] Thomas A. Henzinger, Zohar Manna, and Amir Pnueli. What good are digital clocks? In Werner Kuich, editor, *ICALP*, volume 623 of *Lecture Notes in Computer Science*, pages 545–558. Springer, 1992.

[HR00] Thomas A. Henzinger and Jean-François Raskin. Robust undecidability of timed and hybrid systems. In Nancy A. Lynch and Bruce H. Krogh, editors, *HSCC*, volume 1790 of *Lecture Notes in Computer Science*, pages 145–159. Springer, 2000.

[Imm95] K.A.S. Immink. EFMPlus: The coding format of the multimedia compact disc. *IEEE Transactions on Consumer Electronics*, 41(3):491–497, 1995.

[JN12] Marcin Jurdzinski and Dejan Nickovic, editors. *Formal Modeling and Analysis of Timed Systems - 10th International Conference, FORMATS 2012, London, UK, September 18-20, 2012. Proceedings*, volume 7595 of *Lecture Notes in Computer Science*. Springer, 2012.

[KBM13] Jean-Francois Kempf, Marius Bozga, and Oded Maler. As soon as probable: Optimal scheduling under stochastic uncertainty. In Nir Piterman and Scott A. Smolka, editors, *TACAS*, volume 7795 of *Lecture Notes in Computer Science 7795*, pages 385–400. Springer, 2013.

[Kit11] Sergey Kitaev. *Patterns in permutations and words*. Springer, 2011.

[KLS89] M.A. Krasnosel'skij, E.A. Lifshits, and A.V. Sobolev. *Positive Linear Systems: The method of positive operators*. Number 5 in Sigma Series in Applied Mathematics. Heldermann Verlag, Berlin, 1989.

[Krc09] Pavel Krcál. *Infinite Structures in Timed Systems*. PhD thesis, University of Uppsala, Dept. of Information Technology, May 2009.

[KT59] A.N. Kolmogorov and V.M. Tikhomirov. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Mat. Nauk*, 14(2):3–86, 1959. Russian, partial English translation in [Edg93].

[LM95] Douglas Lind and Brian Marcus. *An introduction to symbolic dynamics and coding*. Cambridge University Press, 1995.

[Lot05] M. Lothaire. *Applied Combinatorics on Words (Encyclopedia of Mathematics and its Applications)*. Cambridge University Press, New York, NY, USA, 2005.

[LW00] Elon Lindenstrauss and Benjamin Weiss. Mean topological dimension. *Israel J. of Math.*, 115:1–24, 2000.

[Mar12] Philippe Marchal. Generating random alternating permutations in time n log n. 2012.

[MP04] Oded Maler and Amir Pnueli. On recognizable timed languages. In Igor Walukiewicz, editor, *FoSSaCS*, volume 2987 of *Lecture Notes in Computer Science*, pages 348–362. Springer, 2004.

[MRS98] Brian Marcus, Ron M. Roth, and Paul H. Siegel. Constrained systems and coding for recording channels. In *Handbook of Coding Theory*, pages 1635–1764. North-Holland, 1998.

[NW78] Albert Nijenhuis and Herbert S Wilf. Combinatorial algorithms for computers and calculators. *Computer Science and Applied Mathematics, New York: Academic Press, 1978, 2nd ed.*, 1, 1978.

[Par64] W. Parry. Intrinsic Markov chains. *Transactions of the American Mathematical Society*, pages 55–66, 1964.

[Pin64] S. Pinsker. *Information and information stability of random variables and processes*. Holden-Day series in time series analysis. Holden-Day, 1964.

[Pur00] Anuj Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.

[SBMR13] Ocan Sankur, Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust controller synthesis in timed automata. In Pedro R. D'Argenio and Hernán C. Melgratti, editors, *CONCUR*, volume 8052 of *Lecture Notes in Computer Science*, pages 546–560. Springer, 2013.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J.*, 27:379–423, 623–656, 1948.

[Sha51] Claude E Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

[Sim90] Imre Simon. Factorization forests of finite height. *Theor. Comput. Sci.*, 72(1):65–94, 1990.

[SS78] A. Salomaa and M. Soittola. *Automata-theoretic aspects of formal power series*. Springer Verlag, 1978.

[Sta86] R.P. Stanley. Two poset polytopes. *Discrete & Computational Geometry*, 1(1):9–23, 1986.

[Sta10] Richard P. Stanley. A survey of alternating permutations. In *Combinatorics and graphs*, volume 531 of *Contemp. Math.*, pages 165–196. Amer. Math. Soc., Providence, RI, 2010.

[Sta12] Amélie Stainer. Frequencies in forgetful timed automata. In Jurdzinski and Nickovic [JN12], pages 236–251.

[SV07] Luigi Sassoli and Enrico Vicario. Close form derivation of state-density functions over dbm domains in the analysis of non-markovian models. In *QEST*, pages 59–68. IEEE Computer Society, 2007.

[SW99] H.H. Schaefer and M.P.H. Wolff. *Topological vector spaces*, volume 3. Springer Verlag, 1999.

[Szp01] George G Szpiro. The number of permutations with a given signature, and the expectations of their elements. *Discrete Mathematics*, 226(1):423–430, 2001.

[WDMR08] Martin De Wulf, Laurent Doyen, Nicolas Markey, and Jean-François Raskin. Robust safety of timed automata. *Formal Methods in System Design*, 33(1-3):45–84, 2008.

[WDR05] Martin De Wulf, Laurent Doyen, and Jean-François Raskin. Almost asap semantics: from timed models to timed implementations. *Formal Asp. Comput.*, 17(3):319–341, 2005.

List of Figures

1 Un automate temporisé $\mathcal{A}^{ex}$ 11

1.1 A timed automaton $\mathcal{A}^{ex}$ 21

2.1 A bounded deterministic timed automaton $\mathcal{A}$ 35
2.2 The TRG $\mathcal{G}^{\mathtt{ex1}}$ and its state space 36
2.3 Fleshy region-split forms of automata $\mathcal{A}$ 42
2.4 The three polytopes associated to a path. 45

3.1 Examples of thin and thick TRGs 48
3.2 Two non strongly connected orbit graphs 53

5.1 A right resolving LCTRG 97
5.2 LCTRGs for Examples 5, 6, 7 and 8 98

6.1 A sofic automaton of a channel C (left) and a $(\{0,1\}^*, C)$-encoder (right) . . 117
6.2 An abstract representation of a LCTRG 119
6.3 $\mathcal{A}_\varepsilon^-$: an automaton recognizing C_ε^- with $\varepsilon = 1$. 125
6.4 The split form of $\mathcal{A}_\varepsilon^-$. 125
6.5 Top: transducers $\tau_1 : S \to C$ and $\tau_2 : C \to S$. Bottom: languages S_2 and C_2. 128

7.1 Timed automata. First line: $\mathcal{A}_1$, $\mathcal{A}_2$; second line: $\mathcal{A}_3$, $\mathcal{A}_4$. 131
7.2 A regenerating automaton 140
7.3 A real-time automaton 142

8.1 From left to right: automata for L^{ex}, $L^{ex'}$ and $\mathtt{st}_{\mathbf{d}}(L^{ex'})$ 148
8.2 An automaton for $(\mathbf{da})^*(\varepsilon + \mathbf{d})$ and its S-T encoding of type $\mathbf{d}$ 160
8.3 From left to right automata for L^{ex_3}, $L^{ex'_3} = \{\varepsilon\} \cup L^{ex_3}.\{\mathbf{a}\}$ and $\mathtt{st}_{\mathbf{a}}(L^{ex'_3})$. 162

Abstract

Since early 90s, timed automata and timed languages are extensively used for modelling and verification of real-time systems, and thoroughly explored from a theoretical standpoint. Recently Asarin and Degorre introduced the notions of volume and entropy of timed languages to quantify the size of these languages and the information content of their elements.

In this thesis we build new developments of this theory (called by us volumetry of timed languages) and apply it to several problems occurring in various domains of theoretical computer science such as verification, enumerative combinatorics or information theory.

Among other we (i) develop a theory of timed symbolic dynamics; (ii) characterize a dichotomy between bad behaving and well behaving timed automata; (iii) define a least biased stochastic process for a timed automaton; (iv) develop a timed theory of constrained channel coding; (v) count and generate randomly and uniformly permutations in certain classes.

Résumé

Depuis le début des années 90, les automates temporisés et les langages temporisés ont été largement utilisés pour modéliser et vérifier les systèmes temps réels. Ces langages ont aussi été largement étudié d'un point de vue théorique. Plus récemment, Asarin et Degorre ont introduit les notions de volume et d'entropie des langages temporisés pour quantifier la taille de ces langages et l'information que ses éléments contiennent.

Dans cette thèse nous construisons de nouveaux développements à cette théorie (que nous appelons volumétrie des langages temporisés) et l'appliquons a plusieurs problèmes apparaissant dans divers domaines de recherche tels que la théorie de l'information, la vérification, la combinatoire énumérative.

Entre autre, nous (i) développons une théorie de la dynamique symbolique temporisée ; (ii) caractérisons une dichotomie entre les automates temporisés se comportant bien ou mal ; (iii) définissons pour un automate temporisé donné, un processus stochastique d'entropie maximale le moins biaisé possible ; (iv) développons une version temporisé de la théorie des codes sur canal contraint (v) énumérons et générons aléatoirement des permutations dans une certaine classe.