Une approche générique à la vérification quantitative

Uli Fahrenberg, Aline Fahrenberg

Une approche générique à la vérification quantitative

Uli Fahrenberg Aline Fahrenberg (trad.)

10 septembre 2023

1 Introduction

Ceci est une traduction française de l'introduction du mémoire d'habilitation à diriger des recherches (HDR) rédigé par Uli Fahrenberg [Fah22]. Elle a été produite par *Aline Fahrenberg* en réaction à plusieurs demandes d'une version française de ce mémoire. Le mémoire HDR en version complète n'existe toujours qu'en anglais, mais nous espérons que cette introduction suffira à aiguiser l'intérêt du lecteur et le convaincre de lire les autres chapitres en anglais.

Rennes, septembre 2023
Aline & Uli Fahrenberg

1.1 Préambule

Le mémoire [Fah22] a pour thème la vérification quantitative, c'est-à-dire la vérification des propriétés quantitatives de systèmes quantitatifs. Ces systèmes sont présents dans de nombreuses applications, et leur vérification quantitative est cruciale mais relève aussi du défi. En particulier, étant donné que la plupart des systèmes se trouvant dans les applications sont assez lourds, il est alors essentiel que les méthodes utilisées soient compositionnelles et incrémentielles. En effet, la vérification quantitative devrait autant que possible être appliquée à des sous-systèmes et ceci au niveau le plus abstrait possible, ensuite les spécifications partielles vérifiées devraient être composées et affinées pour enfin construire l'implémentation.

Le domaine de la conception compositionnelle et incrémentielle est un domaine déjà bien développé, cependant il n'existe pas d'extensions quantitatives satisfaisantes. Ce mémoire présente le travail de publication de 2009 à 2020 de l'auteur et de différents co-auteurs dans cette perspective. Beaucoup reste à faire dans cette thématique, en particulier au niveau des applications dédiées aux systèmes hybrides et temps-réel, mais nous pensons que les fondements développés ici seront utiles dans cette entreprise.

1.1.1 Vérification quantitative

Motivée par les applications dédiées aux systèmes temps-réel, systèmes hybrides et systèmes embarqués, entre autres, la vérification formelle s'est

orientée vers la modélisation et l'analyse des systèmes qui contiennent des informations quantitatives. Les informations quantitatives peuvent donc être très diverses : probabilités, temps, pression du réservoir, consommation d'énergie, *etc.*

Un certain nombre de modèles quantitatifs ont été développés : automates probabilistes [SL94] ; algèbres stochastiques de processus [Hil96] ; automates temporisés [AD94] ; automates hybrides [ACH+95] ; variants temporisés de réseaux de Petri [MF76, Han93] ; processus de Markov à temps continu [Ste94] ; *etc.* De manière similaire, il existe un certain nombre de formalisme de spécification pour exprimer les propriétés quantitatives : logique arborescente temporisée [HNSY94] ; logique arborescente probabiliste [HJ94] ; logique temporelle métrique [Koy90] ; logique stochastique continue [ASSB00] ; *etc.* La vérification quantitative de modèles, c'est-à-dire la vérification des propriétés quantitatives des systèmes quantitatifs, connaît un développement rapide : en ce qui concerne les systèmes probabilistes avec PRISM [KNP02] et PEPA [GH94] ; pour les systèmes temps-réel avec Uppaal [LPY97], RED [WME93], TAPAAL [BJS09] et Romeo [GLMR05] ; et pour les systèmes hybrides avec HyTech [HHWT97], SpaceEx [FGD+11] et HySAT [FH07], pour en nommer quelques uns.

Cependant, la vérification quantitative de modèles se trouve confrontée à un problème de *robustesse*. Lorsque les réponses à des problèmes de vérification de modèles sont des booléennes—ou bien le système vérifie les spécifications ou bien ce n'est pas le cas—alors de petites perturbations dans les paramètres du système peuvent invalider le résultat. Ce qui veut dire que, du point de vue de la vérification de modèles, de petites voire insignifiantes altérations de quantités ne peuvent pas être distinguées d'altérations plus importantes qui pourraient être critiques.

Par exemple, la figure 1.1 présente trois modèles d'automates temporisés simples d'un passage à niveau, chacun modélise qu'une fois les barrières fermées, un certain temps doit s'écouler avant l'arrivée du train. Supposons à présent que la spécification du système soit

Les barrières doivent être fermées 60 secondes avant l'arrivée du train

Le modèle A vérifie cette propriété, et donc respecte la spécification. Le modèle B assure seulement que les barrières seront fermées 58 secondes avant l'arrivée du train, et avec le modèle C une seconde uniquement s'écoulera entre la fermeture des barrières et l'arrivée du train.

Ni le modèle B ni le modèle C ne respectent la spécification, ce résultat est celui qui serait produit par un vérificateur de modèle comme Uppaal par exemple. Ce que cela ne nous dit pas, cependant, c'est que le modèle C s'éloigne dangereusement de la spécification, alors que le modèle B ne le transgresse qu'à peine, et il se peut que ce dernier soit toutefois acceptable lorsque l'on prend en compte d'autres contraintes d'ingénierie, ou encore qu'il soit davantage susceptible d'être adapté pour satisfaire les spécifications que le modèle C.

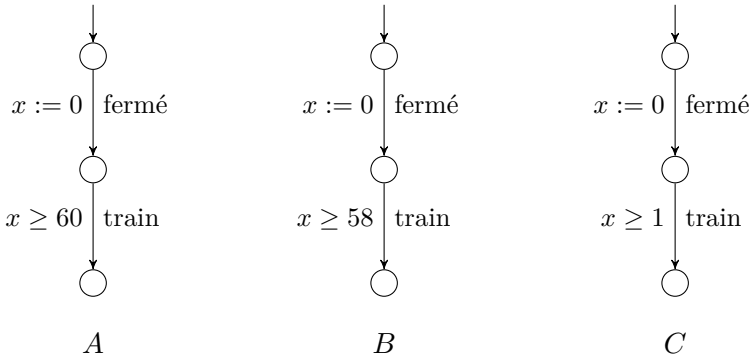

Figure 1.1 – Trois automates temporisés modélisant un passage à niveau.

Afin de prendre en compte le problème de la robustesse, notre approche est de remplacer la réponse booléenne oui/non de vérification classique par la notion de *distances*. Par conséquent, le codomaine booléen de la vérification du modèle est remplacé par un nombre réel positif. Dans ce réajustement, le booléen `vrai` correspond à une distance nulle et celui de `faux` correspond à tout nombre non nul, et de ce fait, la vérification quantitative du modèle à présent nous éclaire non seulement sur la spécification qui est enfreinte, mais aussi *dans quelle mesure* elle est enfreinte et à quel point le système s'éloigne de ses spécifications.

Dans l'exemple de la figure 1.1 et pour une définiton simple des distances systèmes, la distance de A à notre spécification serait de 0, alors que les distances de B et C à la spécification seraient respectivement de 2 et 59. L'interprétation précise des valeurs des distances se fera en relation avec l'application ; mais quoi qu'il en soit, il apparaît clairement que C est bien plus éloigné que B de la spécification.

1.1.2 Théories de spécification

A l'heure actuelle, l'un des défis majeurs dans la conception rigoureuse de logiciels systèmes réside dans le fait que ces systèmes sont d'une complexité croissante et qu'il devient alors laborieux de raisonner à leurs propos [Sif11]. A titre d'exemple, un système de communication intégré à bord d'un avion moderne peut présenter jusqu'à 10^{900} états différents [BBB+10], et les outils de pointe n'offrent pas la possibilité de l'examiner si ce n'est à travers la vérification du modèle dans son ensemble comme un tout. Une approche prometteuse pour surmonter ce type de problèmes est celle de la *conception compositionnelle et incrémentielle*. Ici l'analyse se porte autant que possible sur des niveaux élevés de *specification* plutôt que sur les *implementations* ; les spécifications partielles avérées correctes sont alors composées et affinées jusqu'à l'obtention d'un modèle d'implémentation. L'expérience pratique montre

la pertinence d'une telle approche [Str, SPE].

Les spécifications des exigences du système sont des abstractions finies de haut niveau d'ensembles, potentiellement infinis, d'implémentations. Un modèle d'un système est considéré comme étant une implémentation d'une spécification donnée si le comportement de l'implémentation est impliqué par la description prescrite par la spécification.

Tout formalisme pratique de spécifications est muni d'un certain nombre d'opérations qui permettent le raisonnement compositionnel et incrémentiel. La première d'entre elles est une relation d'*affinement* qui permet successivement plus de finesse et de détail dans les spécifications et ensuite une implémentation. Dans l'implémentation, tout comportement optionel défini dans la spécification l'a été en conformité avec la spécification elle-même. Une opération également nécessaire est celle de *conjonction logique* qui permet de combiner les spécifications afin que les systèmes qui affinent la conjonction de deux spécifications soient précisément ceux qui vérifient les deux. Ensemble, l'affinement et la conjonction permettent le raisonnement incrémentiel où les spécifications sont successivement affinées et conjointes.

Pour le raisonnement compositionnel, une autre opération de *composition parallèle* est nécessaire pour pouvoir déduire les spécifications de sous spécifications d'exigences indépendantes, imitant au niveau de l'implémentation par exemple les intéractions de composants d'un système distribué. L'opération *quotient* en est l'opération inverse partielle, celle-ci permet de synthétiser une spécification de composants manquants en partant d'une spécification globale et d'une implémentation qui réalise une partie de cette spécification.

Au cours de ces dernières années, les théories de spécification ont connu bon nombre d'avancées [dAH05, CdAHM02, DLL+10, Del10, LT89, Nym08, Thr11]. Les approches prédominantes sont basées sur la logique modale et les algèbres de processus, cependant elles ont le désavantage qu'elles n'intègrent pas de manière naturelle à la fois la conjonction logique et à la fois la composition parallèle dans le même formalisme [Lar89]. En conséquence de tels formalismes ne permettent pas le raisonnement incrémentiel lors de l'affinement.

Pour traiter ces problèmes, le concept de *système de transition modal* a été introduit [Lar89]. Succintement, les systèmes de transition modaux sont des systèmes de transition où les transitions peuvent être de deux types : les transitions *must* qui sont nécessaires pour toute implémentation, et les transitions *may* optionnelles dans l'implémentation. Il est aujourd'hui bien établi que les systèmes de transition modaux sont à la mesure de satisfaire aux exigences d'une théorie correcte de spécification, et ce domaine est en pleine évolution, voir par exemple [Nym08, GLS08, GHJ01, GLLS05] ou [AHL+08] pour un aperçu. D'ailleurs, en pratique, l'expérience montre que ce formalisme est suffisamment expressif pour manipuler des problèmes industriels complexes [Str, SPE].

Par exemple, considérons le système de transition modal de la figure 1.2 modélisant les exigences d'un système simple de courrier électronique où les

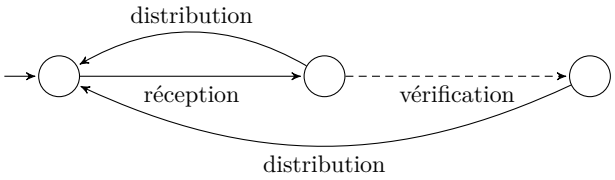

Figure 1.2 – Système de transition modal modélisant un système simple de courrier électronique, avec un comportement optionnel : lorsqu'un courriel est reçu, celui-ci peut par exemple être scanné pour la détection de virus, ou être automatiquement déchiffré, avant qu'il ne soit distribué au récepteur.

courriels sont d'abord reçus et ensuite distribués. Avant de délivrer le courriel, le système peut le vérifier et le traiter par exemple pour un chiffrement ou déchiffrement, filtrer les spams, ou encore pour générer des réponses automatiques (voir aussi [Hal00]). Les transitions *must* qui représentent les comportements indispensables sont indiqués par des flèches pleines, alors que les transitions *may* modélisant les comportements optionnels sont représentées par des flèches en tirets : par conséquent, toute implémentation de cette spécification *doit* être capable de recevoir et de distribuer le courriel, et il *peut* aussi être capable de vérifier le courriel reçu avant de le délivrer. Aucune autre action n'est autorisée.

Les implémentations peuvent aussi être représentées à l'aide du formalisme du système de transition modal, comme des spécifications sans transitions optionnelles. Dans ce cas, tout choix dans l'implémentation a été déterminé, de manière à ce que les implémentations soient (isomorphes à) des systèmes de transition. Formellement, pour qu'un système de transition soit une implémentation d'une spécification donnée, il est requis que les états de deux objets soient reliés par une relation d'affinement ayant la propriété que tout comportement requis par la spécification ait été implémenté, et que toute implémentation de comportement soit permise dans cette spécification. La figure 1.3 montre une implémentation de notre spécification de courrier électronique avec deux vérifications différentes, entraînant des états distincts.

1.1.3 Theories quantitatives de spécification

Par des travaux récents [JLS12, BJL+12a, BJL+12b, BKL+12], les systèmes de transition modaux ont été étendus par l'ajout d'informations plus précises à l'ensemble d'étiquettes discrètes des systèmes de transition habituels, permettant de raisonner sur les aspects *quantitatifs* des modèles et des spécifications. Ces étiquettes quantitatives peuvent être utilisées par exemple pour modéliser et analyser les aspects temporels [HMP05, DLL+10], l'usage de ressources [RLS06, BJL+12b], ou la consommation d'énergie [BFLM11, FJLS11].

En particulier, [JLS12] étend les étiquettes des systèmes de transition mo-

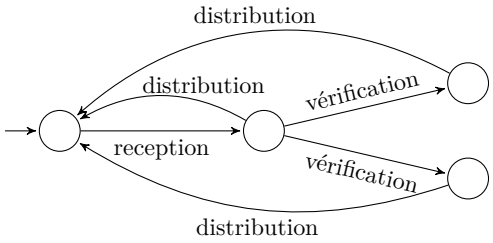

FIGURE 1.3 – Une implémentation du système de courrier électronique de la figure 1.2 dans laquelle deux types distincts de vérification de courriel sont modélisés.

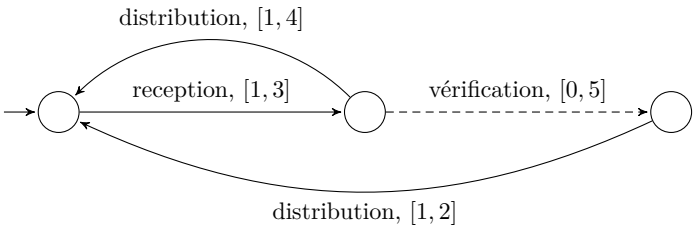

FIGURE 1.4 – Spécification d'un système de courrier électronique où les contraintes de temps pour effectuer certaines actions sont modélisées par des intervales de nombres entiers.

daux aux intervalles de nombres entiers et introduit des extensions aux opérations citées plus haut qui respectent l'information quantitative ajoutée, et [BJL+12a] les généralise en théorie d'*étiquettes structurées*. Ces deux théories cependant sont *fragiles* du fait qu'elles reposent sur des notions booléennes de satisfaction et d'affinement : l'affinement est satisfait ou ne l'est pas, mais elles ne donnent pas la possibilité de *quantifier* l'impact de petites variations de quantités.

Un exemple de spécification quantitative est présenté dans la figure 1.4. Intuitivement, toute implémentation concrète *doit* être capable de recevoir et de distribuer le courrier électronique dans un intervalle de temps de une à trois unités et une à quatre unités, respectivement ; mais elle *peut* aussi être capable de vérifier le courrier entrant, par exemple pour détecter des virus, avant de le distribuer. Aucune autre action ne peut être effectuée.

La figure 1.5 présente quatre implémentations candidates différentes de la spécification de la figure 1.4. La première, dans la figure 1.5(a), comporte une erreur dans la structure discrète : après la réception d'un courrier électronique, la vérification peut avoir lieu indéfiniment. Par conséquent, la spécification n'est pas respectée. La seconde, dans la figure 1.5(b), est aussi problématique : il est permis par la spécification de ne pas implémenter la partie vérification, cependant la réception du courrier prend trop de temps. Ainsi,

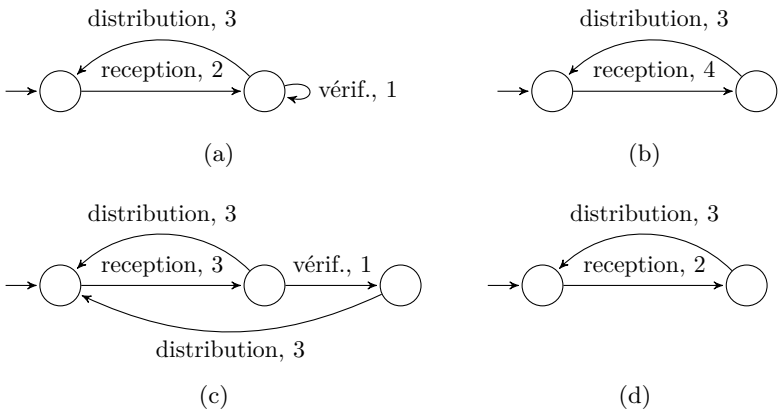

Figure 1.5 – Quatre implémentations du système de courrier électronique de la figure 1.4.

ce serait une implémentation tout à fait correcte si l'on pouvait faire abstraction des contraintes de temps. L'implémentation de la figure 1.5(c) présente le même type de défaut puisqu'après la vérification, la distribution est trop lente. Le système de transition de la figure 1.5(d) est finalement le seul des quatre qui satisfait la spécification.

Une observation importante à présent est que même si les systèmes des figures 1.5(b) et 1.5(c) ne sont pas à strictement parler des implémentations de la spécification du système de courriel, elles s'y conforment pourtant davantage que le système de la figure 1.5(a). Intuitivement, elles vérifient "presque" la spécification ; considérant d'autres contraintes d'ingénierie, elles pourraient être considérées comme "acceptables" dans cette spécification. C'est précisément ce "presque" et "acceptable" que ce mémoire aspire à formaliser.

De manière plus générale, le point de vue adopté ici est que *tout* formalisme quantitatif de spécification avec une notion booléenne de satisfaction et d'affinement n'est pas adapté de manière empirique. Si le formalisme de spécification a pour objectif de modéliser les propriétés quantitatives, alors il n'est pas très utile de savoir qu'une implémentation proposée ne vérifie pas une spécification ; il est beaucoup plus utile de savoir *dans quelle mesure* cette dernière est enfreinte et *à quel point* le système s'éloigne de ses spécifications. La réponse à la question *à quel point* peut bien entendu être ∞, ceci étant dû à des erreurs discrètes comme dans la figure 1.5(a) ; mais dans le cas où la réponse serait finie, des informations importantes pourraient être obtenues, à savoir par exemple quel effort supplémentaire il faudra fournir au niveau de l'implémentation, ou bien si l'on peut se contenter de cette implémentation légèrement imparfaite. Notre approche par conséquent sera de remplacer les relations de satisfaction et d'affinement par des *distances*.

1.1.4 Travaux connexes

L'approche de la vérification quantitative basée sur la notion de distance a été le plus développée dans les systèmes probabilistes et stochastiques. Panangaden et Desharnais *et.al.* ont travaillé avec des distances sur les processus de Markov dans [DGJP04, FPP05, DGJP99, DLT08, DJGP02, Pan09, LMP12, BBLM13] entre autres articles, et van Breugel et Worrell *et.al.* ont développé la notion de distances pour les systèmes de transition probabilistes dans [vBW05, vBW01, vBW06]. De Alfaro et Stoelinga *et.al.* ont travaillé sur les distances entre les systèmes probabilistes et les spécifications dans [dAFH$^+$05, dAHM03, dAMRS07, CdAF$^+$06, CdAMR10, dAMRS08, dAFS04] entre autres.

Concernant les systèmes hybrides et temps-réel, des travaux explicites sur les distances sont disponibles dans [HMP05, CHP11, QFD11]. Par ailleurs, des distances ont été utilisées dans des approches à la vérification robuste [LLTW11, BLM$^+$11], et Girard *et.al.* ont développé une théorie de bisimulation approchée pour le contrôle robuste [ZG09, GP07].

De plus, de larges travaux sur les distances des systèmes quantitatifs où la signification précise des quantités ne sont pas spécifiées ont été menés. Van Breugel a développé une théorie générale de pesudo-métriques comportementales [vB01, BvBR98, vB96, vB05]. Henzinger *et.al.* ont employé des distances dans le contexte du génie logiciel dans [ČHR12, ČHR10] et pour les affinements d'abstractions et la synthèse dans [ČHR13, ČH11, ČCH$^+$11, ČCHR14, CdAF$^+$06].

Toutes les approches basées sur les distances ont en commun d'introduire des distances entre systèmes, ou entre systèmes et spécifications, et d'employer ensuite celles-ci pour la vérification approchée ou quantitative. Cependant, suivant le contexte de l'application, il existe pléthore de différentes distances utilisées, ce qui motive la nécessité d'une théorie générale. Ce point de vue est aussi argumenté dans [ČHR13, CdAF$^+$06].

Pour être plus précis, la plupart des approches citées peuvent être classifiées suivant la manière dont elles mesurent les distances entre les *executions*, ou les traces des systèmes. La manière la plus commode de le faire est d'utiliser la distance *point par point* qui mesure la plus grande distance individuelle entre des éléments correspondants dans les traces des systèmes. La théorie autour de cette distance spécifique a été développée dans [dAFS09, dAFH$^+$05, dAFS04, BLM$^+$11] entre autres. Parfois un *escompte* est appliqué pour diminuer l'influence des distances individuelles dans un futur plus éloigné comme dans par exemple [dAFS09, dAFH$^+$05, dAFS04].

Une autre distance utilisée est la *distance accumulative*, elle somme les distances individuelles au cours des exécutions. Deux principaux types sont alors considérés : la distance accumulative avec *escompte* dans par exemple [ČHR10, dAHM03, AT11, dAFH$^+$05] et la distance accumulative avec *gain moyen* dans par exemple [ČHR10, AT11]. Ces deux dernières sont bien documentées dans la théorie des jeux avec escompte et gain moyen [EM79, ZP96].

Concernant les systèmes temps-réel, une distance intéressante est la distance d'*avance maximale* de [HMP05], elle mesure la différence maximale entre les délais accumulés dans des traces. Pour les systèmes hybrides, les choses sont plus délicates puisque les distances entre les traces hybrides doivent également prendre en compte les différences spaciales et temporelles, comme dans par exemple [QFD11, Gir10, ZG09, GP07].

1.1.5 Theorie Générale de la Vérification Quantitative

Suivant le contexte de l'application, de nombreux types de distances sont utilisés dans la vérification quantitative. Par conséquent, une théorie générale de distances systèmes qui puisse faire abstraction des distances concrètes tout en développant la vérification quantitative indépendamment de la distance, devient nécessaire. Notre point de vue à propos de la théorie de la vérification quantitative, est que les aspects quantitatifs doivent être traités comme des entrées, tout autant que les aspects qualitatifs.

L'objet du travail présenté est de développer une telle théorie générale de vérification quantitative. Nous considérons alors comme entrée une distance entre traces, ou exécutions, et nous utilisons ensuite la théorie des jeux aux objectifs quantitatifs pour définir des distances entre des systèmes quantitatifs. Les différentes versions du jeu de bisimulation (quantitatif) entraînent différents types de distances, par exemple distance de bisimulation, distance de simulation, distance d'équivalence de trace, *etc.*, ceci nous donnant la possibilité de construire une généralisation quantitative du spectre temps-linéaire–temps-branchant.

Nous étendrons aussi notre théorie générale de vérification quantitative à une théorie de spécifications quantitatives. Nous utiliserons pour cela les systèmes modaux de transition, et nous développerons les propriétés quantitatives des opérateurs usuels au niveau des théories de spécification comportementales. Tout cela sera fait indépendamment de la distance concrète entre les traces qui sera utilisée.

1.2 Contributions

Dans les chapitres qui suivent, nous présenterons un travail basé sur huit articles, publiés entre 2009 et 2020 par l'auteur de ce mémoire avec différents co-auteurs, sur les théories de la vérification et de la spécification quantitatives. Les trois premiers, chapitres 2 à 4, portent sur les propriétés de trois distances systèmes spécifiques : la distance point par point, la distance accumulative avec escompte, et la distance d'avance maximale. Dans le chapitre 5 suivant, nous développerons notre théorie générale de vérification quantitative et nous montrerons certaines propriétés basiques. Les chapitres 6 et 7 étendent alors cette théorie aux théories de spécification, d'abord pour la distance accumulative avec escompte dans le chapitre 6 puis dans un cadre plus général

dans le chapitre 7. Dans le chapitre 8 nous mettrons en pause le cadre quantitatif afin d'introduire une extension des systèmes modaux de transition en montrant que la théorie de spécification ainsi obtenue est en relation étroite avec d'autres formalismes de spécification populaires. Le dernier chapitre 9 étend ces résultats à des théories générales de spéficication quantitative et développe leurs propriétés.

En comparaison des articles originaux, chaque chapitre a été soigneusement rédigé, de manière à corriger certaines erreurs, unifier la notation, et lisser la présentation. L'auteur de ce mémoire engage sa responsabilité concernant toute erreur résiduelle.

1.2.1 Preliminaires Géométriques

Avant de pouvoir donner un aperçu de notre contribution, nous rappellerons quelques notions classiques de géométrie et de topologie que nous utiliserons. Soit $\mathbb{R}_{\geq 0} \cup \{\infty\}$ l'ensemble des réels non négatifs étendu.

Un *hémimétrique* sur un ensemble X est une application $d : X \times X \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ qui satisfait $d(x,x) = 0$ et $d(x,y) + d(y,z) \geq d(x,z)$ (*inégalité triangulaire*) pour tout $x, y, z \in X$. L'hémimétrique est dit *symétrique* si $d(x,y) = d(y,x)$ pour tout $x, y \in X$; il vérifie la propriété de *séparation* si $d(x,y) = 0$ implique $x = y$.

Un hémimétrique symétrique est en général appelé *pseudométrique*, et un hémimétrique qui est à la fois symétrique et séparé est simplement un *métrique*. Le couple (X, d) est appelé un *espace* (hémi/pseudo)métrique.

On peut noter que les hémimétriques sont *étendus* et peuvent donc prendre la valeur ∞. Ceci se révèle pratique pour différentes raisons, *cf.* [Law73], l'une étant que cela permet l'union disjointe ou coproduit d'espaces hémimétriques : l'union disjointe des espaces (X_1, d_1) et (X_2, d_2) est l'espace hémimetrique $(X_1, d_1) \uplus (X_1, d_2) = (X_1 \uplus X_2, d)$ où les points des différents composants sont infiniment distants les uns des autres, *i.e.*, avec d définie par

$$d(x,y) = \begin{cases} d_1(x,y) & \text{si } x, y \in X_1, \\ d_2(x,y) & \text{si } x, y \in X_2, \\ \infty & \text{sinon.} \end{cases}$$

Le *produit* de deux espaces hémimétriques (X_1, d_1) et (X_2, d_2) est l'espace hémimétrique $(X_1, d_1) \times (X_2, d_2) = (X_1 \times X_2, d)$ avec d donnée par $d((x_1, x_2), (y_1, y_2)) = \max(d_1(x_1, y_1), d_2(x_2, y_2))$.

La *symétrisation* d'un hémimétrique d sur X est l'hémimétrique symétrique $\bar{d} : X \times X \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ défini par $\bar{d}(x,y) = \max(d(x,y), d(y,x))$; c'est le plus petit parmi tous les pseudométriques d' sur X pour lequel $d \leq d'$. La *topologie* générée par un hémimétrique d sur X est définie de manière à être la même que celle engendrée par sa symétrisation $\bar{d}$; elle admet pour ensembles ouverts toutes les unions de boules ouvertes $B(x; r) = \{y \in X \mid \bar{d}(x,y) < r\}$, pour $x \in X$ et $r > 0$.

Une fonction continue $f : X \to X$ sur un espace pseudométrique (X, d) est appelée *contractante* si il existe $0 \leq \alpha < 1$ (sa *constante de Lipschitz*) tel que $d(f(x), f(y)) \leq \alpha d(x, y)$ pour tous $x, y \in X$.

Deux pseudométriques d_1, d_2 sur X sont dits être

— *topologiquement équivalents* si pour tout $x \in X$ et tout $\varepsilon \in \mathbb{R}_+$, il existe $\delta \in \mathbb{R}_+$ tel que $d_1(x, y) < \delta$ implique $d_2(x, y) < \varepsilon$ et $d_2(x, y) < \delta$ implique $d_1(x, y) < \varepsilon$ pour tout $y \in X$,

— *Lipschitz équivalents* s'il existe $m, M \in \mathbb{R}_+$ tel que $md_1(x, y) \leq d_2(x, y) \leq Md_1(x, y)$ pour tous $x, y \in X$.

Des hémimétriques sont topologiquement ou Lipschitz équivalents si leurs symétrisations le sont.

L'équivalence topologique revient à demander à la fonction identité $\mathsf{id} : (X, d_1) \to (X, d_2)$ d'être un homéomorphisme, aussi l'équivalence de Lipschitz implique l'équivalence topologique.

L'équivalence topologique de d_1 et d_2 est aussi la même chose que d'exiger des topologies engendrées par d_1 et d_2 de coïncider. Cependant, l'équivalence topologique conserve les notions topologiques comme la convergence des séquences : Si une séquence (x_j) de points dans X converge dans l'une des pseudométriques, alors elle converge aussi dans l'autre. Par conséquent, l'équivalence topologique des hémimétriques d_1 et d_2 implique que pour tous $x, y \in X$, $d_1(x, y) = 0$, si et seulement si $d_2(x, y) = 0$.

L'équivalence topologique est la plus faible des notions communes d'équivalence de métriques ; elle ne conserve pas les propriétés géométriques comme la distance ou les angles. Nous sommes cependant principalement intéressés par le fait d'utiliser l'équivalence topologique comme un outil pour montrer des propriétés *négatives* ; nous démontrerons plus tard un certain nombre de résultats sur l'*inéquivalence* topologique des hémimétriques qui impliquera que tout autre équivalence de métrique raisonnable, comme l'équivalence de Lipschitz, échoue également pour ces cas précis.

L'*hémimétrique de Hausdorff* associé à un hémimétrique $d : X \times X \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ est la fonction $d^{\mathrm{H}} : 2^X \times 2^X \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ donnée pour les sous-ensembles $A, B \subseteq X$ par

$$d^{\mathrm{H}}(A, B) = \sup_{x \in A} \inf_{y \in B} d(x, y).$$

C'est une construction bien connue pour les espaces métriques, *cf.* [Mun00, AB07] ; dans ce cas il est habituellement symétrisé et défini seulement pour des sous-ensembles *fermés*, auquel cas il devient un métrique. La formulation alternative suivante découle directement de la définition :

1.1 Proposition. *Pour un hémimétrique d sur X, $A, B \subseteq X$ et $\varepsilon \in \mathbb{R}_+$, on a $d(A, B) \leq \varepsilon$ si et seulement si, pour tout $x \in A$, il existe $y \in B$ pour lequel $d(x, y) \leq \varepsilon$.*

Une séquence (x_j) dans un espace métrique X est une *séquence de Cauchy* si on peut vérifier que pour tout $\varepsilon > 0$, il existe $N \in \mathbb{N}$ tel que $d(x_m, x_n) < \varepsilon$ pour tous $n, m \geq N$. X est dit *complet* si toute séquence de Cauchy dans X converge dans X.

Enfin, nous rappellerons le théorème du point fixe de Banach : Toute fonction contractante sur un espace métrique complet admet précisément un point fixe.

1.2.2 Chapitre 2, "Analyse quantitative de systèmes de transition pondérés"

Dans le chapitre 2 nous introduirons les distances de traces point par point, accumulative et d'avance maximale, toutes trois dans une version avec escompte. Avec une notation plus simple que celle utilisée dans le chapitre 2 et adaptée aux chapitres suivants, ces distances seront exprimées ainsi : Soit $\mathbb{K}$ un ensemble de symboles et une métrique étendue $d_{\mathbb{K}} : \mathbb{K} \times \mathbb{K} \to \mathbb{R}_{\geq 0} \cup \{\infty\}$. Une *trace* $\sigma = (\sigma_0, \sigma_1, \dots)$ est une séquence infinie de symboles dans $\mathbb{K}$. Soit $\lambda \in \mathbb{R}_{\geq 0}$ tel que $\lambda \leq 1$ un *facteur d'escompte* et σ et τ des traces.

— La *distance de trace point par point* entre σ et τ est

$$d_{\bullet}^{\mathsf{T}}(\sigma, \tau) = \sup_{i \geq 0} \lambda^i d_{\mathbb{K}}(\sigma_i, \tau_i)\,.$$

— La *distance de trace accumulative* entre σ et τ est

$$d_{+}^{\mathsf{T}}(\sigma, \tau) = \sum_{i \geq 0} \lambda^i d_{\mathbb{K}}(\sigma_i, \tau_i)\,.$$

— La *distance de trace avec avance maximale* entre σ et τ est

$$d_{\pm}^{\mathsf{T}}(\sigma, \tau) = \sup_{i \geq 0} \lambda^i \Big| \sum_{0 \leq j \leq i} (\sigma_i - \tau_i) \Big|\,.$$

On peut remarquer que la définition de la dernière distance requiert des opérations internes d'addition et de soustraction sur $\mathbb{K}$; en général celle-ci n'est employée que sur $\mathbb{K} = \mathbb{Z}$ ou $\mathbb{K} = \mathbb{R}$.

Nous utilisons alors ces distances de trace pour définir les distances *linéaires* point par point, accumulative et avec avance maximale entre des états dans des systèmes de transition pondérés. Si d^{T} est l'une de ces distances de trace citée plus haut, alors la distance linéaire entre deux états s et t d'un système de transition est défini par :

$$d^{\mathsf{L}}(s, t) = \sup_{\sigma \in \mathsf{Tr}(s)} \inf_{\tau \in \mathsf{Tr}(t)} d^{\mathsf{T}}(\sigma, \tau)\,,$$

où $\mathsf{Tr}(s)$ indique l'ensemble des traces émanant de s, même chose pour $\mathsf{Tr}(t)$. Nous avons ainsi la distace de Hausdorff de $\mathsf{Tr}(s)$ à $\mathsf{Tr}(t)$; nous pouvons remarquer que la définition est indépendante de la distance de trace qui sera utilisée.

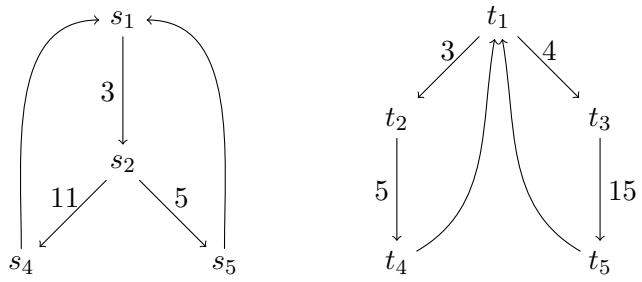

Figure 1.6 – Deux exemples de systèmes de transition

1.2 Example. Nous montrons un exemple de calcul des différentes distances entre les états s_1 et t_1 dans le système de transition dans la figure 1.6. Les sommets qui n'ont pas de poids spécifié ont pour poids 0, et pour facteur d'escompte on utilise $\lambda = 0{,}90$.

Il est aisé de voir que la distance de trace maximale est obtenue pour le chemin qui de s_1 tourne toujours à gauche en s_2, *i.e.*, qui prend la transition $s_2 \xrightarrow{11} s_4$, puis pour les distances de trace point par point et accumulative, que la trace correspondante issue de t_1 donnant la distance de trace minimale est obtenue avec le chemin qui prend toujours la transition $t_1 \xrightarrow{4} t_3$. Ainsi nous pouvons calculer :

$$d_\bullet^{\mathsf{L}}(s_1, t_1) = \sup_i \left\{ \max(1, 4 \cdot 0{,}90) \cdot 0{,}90^{3i} \right\} = 3{,}60\,,$$
$$d_+^{\mathsf{L}}(s_1, t_1) = \sum_i (1 + 4 \cdot 0{,}90) \cdot 0{,}90^{3i} \approx 17{,}0\,.$$

Pour la distance de trace avec avance maximale, la situation est plus complexe. Il peut être montré que pour cette distance, une trace minimale τ à partir de t_1 prend le chemin $t_1 \xrightarrow{4} t_3$, suivi de $t_1 \xrightarrow{3} t_2$ trois fois, et ensuite répète $t_1 \xrightarrow{4} t_3$ indéfiniment. En utilisant cette trace, on obtient :

$$d_\pm^{\mathsf{L}}(s_1, t_1) = 13 \cdot 0{,}90^{10} \approx 4{,}53\,. \qquad \square$$

La définition de la distance de branchement *n'est pas* indépendante de la distance de trace utilisée. Nous donnerons les définitions seulement pour les deux premières de nos exemples de distances de trace. Elles sont définies comme les plus petits points fixes des équations suivantes :

$$d_\bullet^{\mathsf{B}}(s, t) = \sup_{s \xrightarrow{x} s'} \inf_{t \xrightarrow{y} t'} \max\left(d_{\mathbb{K}}(x, y), \lambda\, d_\bullet^{\mathsf{B}}(s', t')\right)$$
$$d_+^{\mathsf{B}}(s, t) = \sup_{s \xrightarrow{x} s'} \inf_{t \xrightarrow{y} t'} d_{\mathbb{K}}(x, y) + \lambda\, d_\bullet^{\mathsf{B}}(s', t')$$

1.3 Example. Pour faire suite à l'exemple précédent, l'application répétée de la définition entraîne l'équation de point fixe suivante pour $d_\bullet^{\mathsf{B}}(s_1, t_1)$ (notons qu'il n'y a qu'une seule transition pour respectivement s_1, t_2 et t_3) :

$$\begin{aligned}
d_\bullet^{\mathsf{B}}(s_1,t_1) &= \inf \begin{cases} \max\left(|3-3|, 0{,}90\, d_\bullet^{\mathsf{B}}(s_2,t_2)\right) \\ \max\left(|3-4|, 0{,}90\, d_\bullet^{\mathsf{B}}(s_2,t_3)\right) \end{cases} \\
&= \inf \begin{cases} \max\left(0, 0{,}90\,|11-5|, 0{,}90^2 d_\bullet^{\mathsf{B}}(s_4,t_4), 0{,}90\,|5-5|, 0{,}90^2 d_\bullet^{\mathsf{B}}(s_5,t_4)\right) \\ \max\left(1, 0{,}90\,|11-15|, 0{,}90^2 d_\bullet^{\mathsf{B}}(s_4,t_5), 0{,}90\,|5-15|, 0{,}90^2 d_\bullet^{\mathsf{B}}(s_5,t_5)\right) \end{cases} \\
&= \inf \begin{cases} \max\left(5{,}4, 0{,}90^3 d_\bullet^{\mathsf{B}}(s_1,t_1)\right) \\ \max\left(9, 0{,}90^3 d_\bullet^{\mathsf{B}}(s_1,t_1)\right) \end{cases}
\end{aligned}$$

qui a le plus petit point fixe $d_\bullet^{\mathsf{B}}(s_1, t_1) = 5{,}4$. Pour la distance accumulative, on calcule :

$$\begin{aligned}
d_+^{\mathsf{B}}(s_1,t_1) &= \inf \begin{cases} |3-3| + 0{,}90\, d_+^{\mathsf{B}}(s_2,t_2) \\ |3-4| + 0{,}90\, d_+^{\mathsf{B}}(s_2,t_3) \end{cases} \\
&= \inf \begin{cases} 0{,}90 \sup \begin{cases} |11-5| + 0{,}90\, d_+^{\mathsf{B}}(s_4,t_4) \\ |5-5| + 0{,}90\, d_+^{\mathsf{B}}(s_5,t_4) \end{cases} \\ 1 + 0{,}90 \sup \begin{cases} |11-15| + 0{,}90\, d_+^{\mathsf{B}}(s_4,t_5) \\ |5-15| + 0{,}90\, d_+^{\mathsf{B}}(s_5,t_5) \end{cases} \end{cases} \\
&= \inf \begin{cases} 0{,}90 \sup \begin{cases} 6 + 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1) \\ 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1) \end{cases} \\ 1 + 0{,}90 \sup \begin{cases} 4 + 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1) \\ 10 + 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1) \end{cases} \end{cases} \\
&= \inf\left(0{,}90(6 + 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1)), 1 + 0{,}90(10 + 0{,}90^2 d_+^{\mathsf{B}}(s_1,t_1))\right) \\
&= 5{,}4 + 0{,}90^3 d_+^{\mathsf{B}}(s_1,t_1)
\end{aligned}$$

Ainsi $d_+^{\mathsf{B}}(s_1, t_1) \approx 19{,}9$. □

Les distances linéaires généralisent *l'inclusion de traces* pour les systèmes de transition, alors que les distances de branchement généralisent *la simulation*. Nous montrons que la distance linéaire entre deux états admet toujours pour borne supérieure la distance de branchement (théorème 2.13), c'est une généralisation du fait que la simulation implique l'inclusion de traces.

Nous montrons également que la distance linéaire point par point et la distance de branchement point par point sont *topologiquement inéquivalentes*, c'est-à-dire que l'une peut être égale à zéro pendant que l'autre peut être infinie. C'est une généralisation quantitative du fait que l'inclusion de traces et la simulation ne sont pas équivalentes. Nous montrons la même inéquivalence topologique pour les distances accumulatives et d'avance maximale (théorème 2.14).

Lorsque un escompte est appliqué, alors les distances point par point, accumulative et d'avance maximale sont *Lipschitz équivalentes* (Theorem 2.17) ; de manière similaire, les trois distances de branchement sont Lipschitz équivalentes (Theorem 2.18). Sans escompte, les distances sont topologiquement inéquivalentes. L'équivalence de Lipschitz signifie qu'une distance admet pour borne supérieure l'autre distance, multipliée par un facteur d'échelle ; ainsi les propriétés de l'une des distances peuvent être héritées par l'autre distance.

Le chapitre 2 est basé sur les travaux du doctorant de l'auteur Claus Thrane, de Kim G. Larsen et de l'auteur. Ils ont été présentés lors du 20ème Nordic Workshop on Programming Theory (NWPT) [TFL08] et publiés par la suite dans le Journal of Logic and Algebraic Programming (maintenant le Journal of Logical and Algebraic Methods in Programming) en 2010 [TFL10].

1.2.3 Chapitre 3, "Une caractérisation quantitative des structures pondérées de Kripke en logique temporelle"

Dans le chapitre 3 nous examinons les distances avec escompte point par point et accumulative et nous introduisons la sémantique correspondante pour la logique *CTL pondérée* (WCTL). Avec cette sémantique, l'évaluation d'une formule dans un état n'est pas un Booléen `vrai` ou `faux`, mais plutôt un nombre réel non négatif (ou l'infini) qui, de manière intuitive, caractérise à quel point un état satisfait une formule. Notre syntaxe pour WCTL est une extension de celle de la CTL [CE81] exprimée de cette manière :

$$\Phi ::= p \mid \neg p \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \mathsf{E}\Psi \mid \mathsf{A}\Psi$$
$$\Psi ::= \mathsf{X}_c\Phi \mid \mathsf{G}_c\Phi \mid \mathsf{F}_c\Phi \mid [\Phi_1 \mathsf{U}_c \Phi_2]$$

Comme d'habitude, ici Φ engendre des formules d'état alors que Ψ engendre des formules de chemin, p est une proposition atomique, et $c \in \mathbb{R}_{\geq 0}$ est un nombre réel non négatif. Sémantiquement, les formules sont interpétées dans des états d'une structure de Kripke avec étiquettes dans $\mathbb{K}$, et le résultat d'une telle interprétation est un nombre réel non négatif. Tout d'abord, la sémantique des formules d'états est donnée ainsi :

$$[\![p]\!](s) = \begin{cases} 0 & \text{si } p \in L(s) \\ \infty & \text{sinon} \end{cases} \qquad [\![\neg p]\!](s) = \begin{cases} 0 & \text{si } p \in AP \setminus L(s) \\ \infty & \text{sinon} \end{cases}$$
$$[\![\varphi_1 \vee \varphi_2]\!](s) = \inf\{[\![\varphi_1]\!](s), [\![\varphi_2]\!](s)\} \qquad [\![\varphi_1 \wedge \varphi_2]\!](s) = \sup\{[\![\varphi_1]\!](s), [\![\varphi_2]\!](s)\}$$
$$[\![\mathsf{E}\psi]\!](s) = \inf\{[\![\psi]\!](\sigma) \mid \sigma \in \mathsf{Tr}(s)\} \qquad [\![\mathsf{A}\psi]\!](s) = \sup\{[\![\psi]\!](\sigma) \mid \sigma \in \mathsf{Tr}(s)\}$$

Dans les deux dernières formules, $[\![\psi]\!](\sigma)$ est la sémantique de la trace σ par rapport à ψ, qui dépend de la distance utilisée (point par point ou accumulative). Par exemple, la sémantique point par point du chemin est

exprimée de la sorte :

$$\begin{aligned}
[\![\varphi]\!](\sigma) &= [\![\varphi]\!](\sigma_0) \\
[\![\mathsf{X}_c\varphi]\!](\sigma) &= \max\left(d_{\mathbb{K}}(\sigma_0, c), \lambda[\![\varphi]\!](\sigma^1)\right\} \\
[\![\mathsf{F}_c\varphi]\!](\sigma) &= \inf_{k\geq 0}\Big(\max\big(\max_{0\leq j<k}\left(\lambda^j d_{\mathbb{K}}(\sigma_j, c)\right), \lambda^k[\![\varphi]\!](\sigma^k)\big)\Big) \\
[\![\mathsf{G}_c\varphi]\!](\sigma) &= \sup_{k\geq 0}\Big(\max\big(\max_{0\leq j<k}\left(\lambda^j d_{\mathbb{K}}(\sigma_j, c)\right), \lambda^k[\![\varphi]\!](\sigma^k)\big)\Big) \\
[\![\varphi_1\mathsf{U}_c\varphi_2]\!](\sigma) &= \inf_{k\geq 0}\Big(\max\big(\max_{0\leq j<k}\left(\lambda^j d_{\mathbb{K}}([\![\varphi_1]\!](\sigma^j), c)\right), \lambda^k[\![\varphi_2]\!](\sigma^k)\big)\Big)
\end{aligned}$$

Ici, $\sigma^k = (\sigma_k, \sigma_{k+1}, \dots)$ dénote le k-décalage de la trace $\sigma = (\sigma_0, \sigma_1, \dots)$.

Ensuite nous montrons dans les théorèmes 3.11 et 3.13 qu'avec ces sémantiques, WCTL est *adéquate* pour les distances de bisimulation correspondantes. C'est-à-dire que la distance de bisimulation entre deux états est précisément la borne supérieure, parmi toutes les formules WCTL, de la valeur absolue de la différence dans l'évaluation de la formule dans ces deux états.

Nous montrons également dans les théorèmes 3.17 et 3.18, qu'avec la sémantique correspondante, WCTL est *expressive* pour les distances point par point et accumulative. Ce qui veut dire qu'étant donné un état dans une structure de Kripke, il existe une formule WCTL qui caractérise cet état dans le sens que la distance de bisimulation à n'importe quel autre état est précisément l'évaluation de la formule dans cet état.

Ces notions d'adéquation et d'expressivité sont des généralisations quantitatives des définitions de Hennessy et Milner dans [HM85].

Le chapitre 3 repose sur les travaux du doctorant de l'auteur Claus Thrane, de Kim G. Larsen et de l'auteur. Ils ont été présentés lors de la 5ème Conference on Mathematical and Engineering Methods in Computer Science (MEMICS ; best paper award) [FLT09] et publiés par la suite dans le Journal of Computing and Informatics [FLT10].

1.2.4 Chapitre 4, "Métriques pour les systèmes de transition pondérés : Axiomatisation"

Dans le chapitre 4, nous développons des axiomatisations valides et complètes de la distance point par point et de la distance accumulative avec escompte pour des processus pondérés finis et réguliers. Dans ce contexte, un processus pondéré fini est exprimé en utilisant la grammaire :

$$E ::= \mathbf{0} \mid n.E \mid E + E \mid \qquad n \in \mathbb{K}\,,$$

où $\mathbb{K}$ est un ensemble fini de poids, $d_{\mathbb{K}}$ une métrique, et où $\mathbf{0}$ désigne le processus vide. Un processus pondéré régulier est exprimé en utilisant la grammaire :

$$E ::= \mathbf{U} \mid X \mid n.E \mid E + E \mid \mu X.E \qquad n \in \mathbb{K}, X \in V\,,$$

$$\text{(A1)}\ \frac{}{\vdash [\mathbf{0}, E] \bowtie r} \quad 0 \bowtie r \qquad \text{(A2)}\ \frac{}{\vdash [n.E, \mathbf{0}] \bowtie r} \quad \infty \bowtie r$$

$$\text{(R1}^{\bullet}\text{)}\ \frac{\vdash [E, F] \bowtie r_1}{\vdash [n.E, m.F] \bowtie r} \quad \max(d_{\mathbb{K}}(n, m), \lambda r_1) \bowtie r$$

$$\text{(R1}^{+}\text{)}\ \frac{\vdash [E, F] \bowtie r_1}{\vdash [n.E, m.F] \bowtie r} \quad d_{\mathbb{K}}(n, m) + \lambda r_1 \bowtie r$$

$$\text{(R2)}\ \frac{\vdash [E_1, F] \bowtie r_1 \qquad \vdash [E_2, F] \bowtie r_2}{\vdash [E_1 + E_2, F] \bowtie r} \quad \max(r_1, r_2) \bowtie r$$

$$\text{(R3)}\ \frac{\vdash [n.E, F_1] \bowtie r_1 \qquad \vdash [n.E, F_2] \bowtie r_2}{\vdash [n.E, F_1 + F_2] \bowtie r} \quad \min(r_1, r_2) \bowtie r$$

Figure 1.7 – Axiomatisation des distances point par point et accumulative avec escompte pour des processus pondérés finis

où V est un ensemble de variables, $\mathbf{U}$ le processus universel, et $\mu X.E$ désigne le point fixe minimal.

Nous établissons ensuite des axiomatisations de la distance point par point et de la distance accumulative avec escompte pour des processus pondérés finis, comme illustré dans la figure 1.7. Celles-ci diffèrent au niveau d'une règle d'inférence seulement : pour la distance point par point, la règle (R1$^{\bullet}$) s'applique, alors que pour la distance accumulative avec escompte, c'est la règle (R1^{+}) qui s'applique. Nous montrons que ces axiomatisations sont valides et complètes dans les théorèmes 4.8, 4.9 et 4.10.

Pour les processus pondérés reguliers, nous développons des axiomatisations similaires que nous montrons être valides et *ε-complètes* dans les théorèmes 4.12, 4.16 et 4.17. Ici, l'ε-complétude signifie qu'une distance de d peut être démontrée dans un intervalle $[d - \varepsilon, d + \varepsilon]$, pour tout réel positif ε.

Le chapitre 4 repose sur les travaux du doctorant de l'auteur Claus Thrane, de Kim G. Larsen et de l'auteur et publiés dans Theoretical Computer Science [LFT11].

1.2.5 Chapitre 5, "Le spectre temps-linéaire–temps-branchant quantitatif"

Le chapitre 5 présente une généralisation du travail fait dans le chapitre 2 dans plusieurs directions. Au lieu de développer une théorie spécifique pour chaque distance de trace, nous considérons la distance de trace comme une entrée et nous développons une théorie générale de distances linéaires et de branchement se rapportant à une distance de trace donnée, mais non spécifiée.

Soit $\mathbb{K}$ un ensemble de symboles, on note $\mathbb{K}^\infty = \mathbb{K}^* \cup \mathbb{K}^\omega$ l'ensemble de traces finies et infinies sur $\mathbb{K}$. Une *distance de trace* est alors une fonction $d : \mathbb{K}^\infty \times \mathbb{K}^\infty \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ qui vérifie $d(\sigma, \sigma) = 0$ et $d(\sigma, \tau) + d(\tau, \chi) \geq d(\sigma, \chi)$ pour tout $\sigma, \tau, \chi \in \mathbb{K}^\infty$, ainsi que $d(\sigma, \tau) = \infty$ si σ et τ ont des longueurs différentes.

Une telle distance de trace générale d étant donnée, on peut définir la distance linéaire entre deux états s et t d'un système de transition par :

$$d^{\text{1-trace}}(s,t) = \sup_{\sigma \in \mathsf{Tr}(s)} \inf_{\tau \in \mathsf{Tr}(t)} d(\sigma, \tau)$$

comme fait auparavant. Puisque ceci généralise le préordre d'inclusion de trace, nous l'appellerons *distance d'inclusion de trace* (1-imbriquée).

En utilisant un *jeu quantitatif Ehrenfeucht-Fraïssé*, on peut alors définir une *distance de simulation* (1-imbriquée) $d^{\text{1-sim}}$ correspondante, et montrer que $d^{\text{1-trace}}(s,t) \leq d^{\text{1-sim}}(s,t)$ pour tous états s, t. De manière similaire, on peut définir la *distance d'équivalence de trace* (1-imbriquée) entre s et t par :

$$d^{\text{1-trace-eq}}(s,t) = \max\Big(\sup_{\sigma \in \mathsf{Tr}(s)} \inf_{\tau \in \mathsf{Tr}(t)} d(\sigma, \tau), \sup_{\tau \in \mathsf{Tr}(t)} \inf_{\sigma \in \mathsf{Tr}(s)} d(\sigma, \tau) \Big)$$

et utiliser un jeu quantitatif Ehrenfeucht-Fraïssé différent pour définir la *distance de bisimulation* d^{bisim}, vérifiant la propriété $d^{\text{1-trace-eq}}(s,t) \leq d^{\text{bisim}}(s,t)$ pour tous s, t.

Dans le chapitre 5, nous généralisons ces considérations pour définir les distances linéaires et de branchement pour la plupart des préordres et équivalences dans le spectre temps-linéaire–temps-branchant de van Glabbeek [vG01].

Ainsi nous pouvons définir des distances imbriquées de simulation, des distances de simulation promptes à portée, des distances futures possibles, des distances à portée, entre autres, toutes paramétrées par la distance de trace donnée mais non spécifiée. Le *spectre quantitatif temps-linéaire–temps-branchant* qui en résulte est illustré dans la figure 1.8.

Nous montrons également que si la distance de trace a une caractérisation récursive dans un treillis L au dessus de $\mathbb{R}_{\geq 0} \cup \{\infty\}$, alors toutes les distances dans le spectre temps-linéaire–temps-branchant quantitatif possèdent une *charactérisation de point fixe* sur L. Par exemple, si $d : \mathbb{K}^\infty \times \mathbb{K}^\infty \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ est la distance point par point, alors :

$$d(\sigma, \tau) = F\big(\sigma_0, \tau_0, d(\sigma^1, \tau^1)\big)$$

pour tous $\sigma, \tau \in \mathbb{K}^\infty$ (rappelons que σ_0 indique la tête de σ et σ^1 sa queue), où $F : \mathbb{K} \times \mathbb{K} \times (\mathbb{R}_{\geq 0} \cup \{\infty\}) \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ est donné par $F(x, y, \alpha) = \max\big(d_{\mathbb{K}}(x, y), \lambda\alpha\big)$. (Dans ce cas le treillis est $L = \mathbb{R}_{\geq 0} \cup \{\infty\}$.)

La distance de simulation est alors le plus petit point fixe des équations :

$$d^{\text{1-sim}}(s,t) = \sup_{s \xrightarrow{x} s'} \inf_{t \xrightarrow{y} t'} F\big(x, y, d^{\text{1-sim}}(s', t')\big)\,.$$

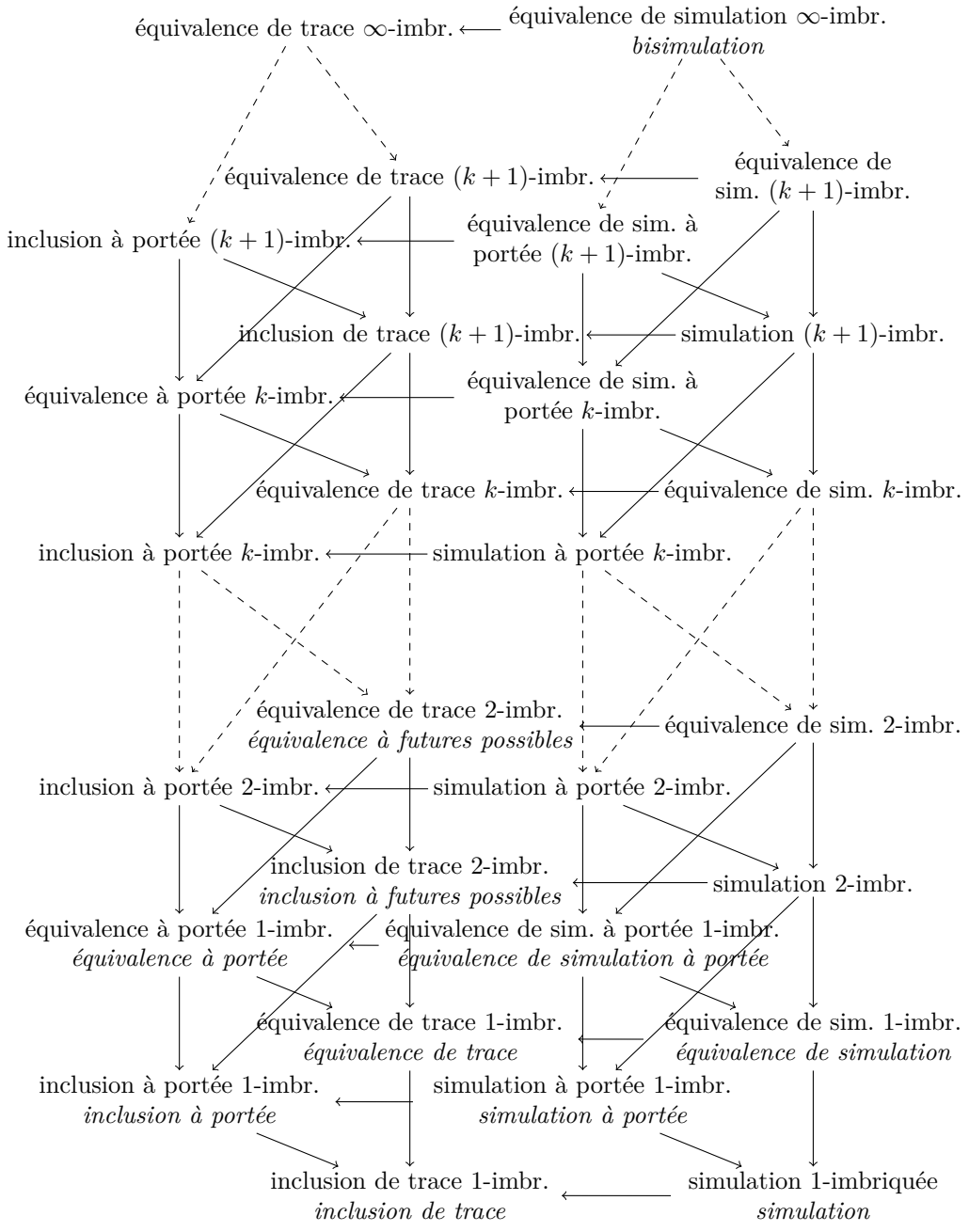

Figure 1.8 – Le spectre temps-linéaire–temps-branchant quantitatif. Les sommets sont des distances système différentes, et un arc $d_1 \longrightarrow d_2$ ou $d_1 \dashrightarrow d_2$ indique que $d_1(s,t) \geq d_2(s,t)$ pour tous états s, t et que d_1 et d_2 sont en général topologiquement inéquivalents.

Si au lieu de cela, d est la distance accumulative avec escompte, alors les équations précédentes sont valables pour F remplacé par $F(x, y, \alpha) = d_{\mathbb{K}}(x, y) + \lambda\alpha$.

Le chapitre 5 repose sur les travaux du doctorant de l'auteur Claus Thrane, de Kim G. Larsen, d'Axel Legay et de l'auteur, qui ont été présentés lors du 9ème Workshop on Quantitative Aspects of Programming Languages (QAPL) [FTL11] et à la 31ème IARCS Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS) [FLT11] et publiés par la suite dans Theoretical Computer Science [FL14b].

1.2.6 Chapitre 6, "Systèmes de transition modaux pondérés"

Le chapitre 6 présente une adaptation de notre travail concernant la vérification quantitative aux *théories de specification quantitative*. L'aspect fondamental des théories de spécification est la *relation d'affinement* qui permet d'affiner les spécifications jusqu'à atteindre une implémentation. Ici les implémentations sont les modèles sur lesquels les chapitres précédents portaient, *i.e.*, systèmes de transition. Dans le contexte de la vérification quantitative, dans les chapitres précédents, nous avons remplacé des relations d'équivalence et de préordre entre les modèles par des distances linéaires et de branchement. Dans le même esprit, nous remplaçons dans ce chapitre la relation d'affinement par une *distance d'affinement*, afin de pouvoir raisonner quantitativement sur des spécifications quantitatives.

Dans le chapitre 6 nous traiterons un cas particulier de la théorie de spécification quantitative, en utilisant des modèles qui sont des systèmes de transition dont les transitions sont étiquetées avec des symboles d'un alphabet discret Σ et avec des nombres entiers. Nous utiliserons aussi une distance particulière, la distance accumulative avec escompte. Dans le chapitre suivant, le chapitre 7, nous généraliserons ce paramétrage à des modèles et des spécifications arbitraires et à des distances arbitraires.

Dans les spécifications du chapitre 6, des nombres entiers sont remplacés par des *intervalles d'entiers* et, comme communément avec les spécifications modales, les transition peuvent être du type *must* (nécessaire) ou du type *may* (possible). Ainsi nous définissons un *système de transition modal pondéré* (WMTS) comme étant une structure $\mathcal{S} = (S, s^0, \dashrightarrow, \longrightarrow)$ qui consiste en un ensemble d'états S avec un état initial $s^0 \in S$ et des relations de transition $\longrightarrow, \dashrightarrow \subseteq S \times \mathsf{Spec} \times S$ tel que pour tout $(s, k, s') \in \longrightarrow$ il existe $(s, \ell, s') \in \dashrightarrow$ où $k \preccurlyeq \ell$. Ici :

$$\mathsf{Spec} = \Sigma \times \{[x, y] \mid x \in \mathbb{Z} \cup \{-\infty\}, y \in \mathbb{Z} \cup \{\infty\}, x \leq y\}$$

est l'ensemble d'*étiquettes de specification* (pondérées), et l'ordre partiel $\preccurlyeq$ sur Spec est défini par $(a, I) \preccurlyeq (a', I')$ si $a = a'$ et $I \subseteq I'$.

Un WMTS $\mathcal{S}$ comme celui au dessus est une *implémentation* si $\longrightarrow = \dashrightarrow \subseteq S \times \mathsf{Imp} \times S$, où

$$\mathsf{Imp} = \Sigma \times \{[x, x] \mid x \in \mathbb{Z}\} \approx \Sigma \times \mathbb{Z}$$

est l'ensemble d'*étiquettes d'implementation* (pondérées) dans Spec : les éléments minimaux de Spec par rapport à $\preccurlyeq$.

À présent, dans un *affinement modal* usuel $\mathcal{S}_1 \leq_{\mathsf{m}} \mathcal{S}_2$, les transitions *must* dans $\mathcal{S}_2$ doivent être préservées dans $\mathcal{S}_1$, alors que les transitions *may* dans $\mathcal{S}_1$ doivent correspondre aux transitions *may* dans $\mathcal{S}_2$. En utilisant la distance accumulative avec l'escompte $\lambda < 1$ et le travail fait dans le chapitre 5, nous étendons cela à une *distance d'affinement modal* qui est définie comme suit. D'abord une distance au niveau des étiquettes de spécification est introduite par : $d_{\mathsf{Spec}}((a, I), (a', I')) = \infty$ si $a \neq a'$ et

$$\begin{aligned} d_{\mathsf{Spec}}((a, [x, y]), (a, [x', y'])) &= \sup_{z \in [x,y]} \inf_{z' \in [x', y']} |z - z'| \\ &= \max(x' - x, y - y', 0)\,. \end{aligned}$$

La distance d'affinement modal entre les états des systèmes de transition modaux pondérés $\mathcal{S}_1 = (S_1, s_1^0, \dashrightarrow_1, \longrightarrow_1)$ et $\mathcal{S}_2 = (S_2, s_2^0, \dashrightarrow_2, \longrightarrow_2)$ est alors définie comme étant le plus petit point fixe des équations :

$$d_{\mathsf{m}}(s_1, s_2) = \max \begin{cases} \sup\limits_{s_1 \xrightarrow{k_1}_1 t_1} \inf\limits_{s_2 \xrightarrow{k_2}_2 t_2} d_{\mathsf{Spec}}(k_1, k_2) + \lambda d_{\mathsf{m}}(t_1, t_2)\,, \\ \sup\limits_{s_2 \xrightarrow{k_2}_2 t_2} \inf\limits_{s_1 \xrightarrow{k_1}_1 t_1} d_{\mathsf{Spec}}(k_1, k_2) + \lambda d_{\mathsf{m}}(t_1, t_2)\,, \end{cases}$$

enfin, $d_{\mathsf{m}}(\mathcal{S}_1, \mathcal{S}_2) = d_{\mathsf{m}}(s_1^0, s_2^0)$.

Dans le théorème 6.14, nous montrons que la distance d'affinement modal est une borne pour la *distance d'affinement par implémentations* : pour toute implémentation $\mathcal{I}_1 \leq_{\mathsf{m}} \mathcal{S}_1$, il existe une implémentation $\mathcal{I}_2 \leq_{\mathsf{m}} \mathcal{S}_2$ telle que $d(\mathcal{I}_1, \mathcal{I}_2) \leq d_{\mathsf{m}}(\mathcal{S}_1, \mathcal{S}_2)$. Ainsi la distance d'affinement modal entre deux spécifications peut servir de majorant de l'éloignement potentiel des implémentations.

Les spécifications modales sont munies d'une opération logique de *conjonction* et des opérations structurelles de *composition* et de *quotient*. La conjonction est la borne inférieure dans le préordre d'affinement modal. Nous montrons que cette conjonction existe pour notre formalisme, mais qu'elle *ne vérifie pas* une généralisation quantitative de la propriété de la borne inférieure ; en fait, le théorème 6.24 montre qu'*il n'y a pas* d'opération $\wedge$ sur WMTS qui vérifie la propriété suivante : pour tout $\varepsilon \geq 0$, il existe $\varepsilon_1, \varepsilon_2 \geq 0$ tel que dès que $d_{\mathsf{m}}(S, S_1) \leq \varepsilon_1$ et $d_{\mathsf{m}}(S, S_2) \leq \varepsilon_2$ pour un WMTS S, S_1, S_2, alors $d_{\mathsf{m}}(S, S_1 \wedge S_2) \leq \varepsilon$. La conjonction est donc, dans ce sens, *discontinue* ; nous verrons dans le chapitre 7 que ceci est un problème fondamental dans *toute* théorie de spécification quantitative.

Pour la composition structurelle, nous utilisons une synchronisation comme dans le CSP pour les étiquettes et l'addition d'intervalles. C'est-à-dire que la synchronisation $(a, I) \oplus (a', I')$ sur Spec n'est pas définie si $a \neq a'$, et que sinon,

$$(a, [x, y]) \oplus (a, [x', y']) = (a, [x + x', y + y']) .$$

En utilisant cette opération d'étiquetage, nous montrons dans le théorème 6.27 qu'il existe un opérateur de composition structurelle $\|$ pour WMTS qui vérifie $d_{\mathsf{m}}(\mathcal{S}_1 \| \mathcal{S}_3, \mathcal{S}_2 \| \mathcal{S}_4) \leq d_{\mathsf{m}}(\mathcal{S}_1, \mathcal{S}_2) + d_{\mathsf{m}}(\mathcal{S}_3, \mathcal{S}_4)$ pour tous WMTS $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$. Cette propriété d'*implémentabilité indépendante* assure que la composition conserve les distances. Nous montrons aussi dans le théorème 6.27 que la composition structurelle admet un inverse partiel, une opération quotient $/$ telle que : $d_{\mathsf{m}}(\mathcal{S}_3, \mathcal{S}_1 / \mathcal{S}_2) = d_{\mathsf{m}}(\mathcal{S}_2 \| \mathcal{S}_3, \mathcal{S}_1)$ pour tous WMTS $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ si $\mathcal{S}_2$ est *déterministe*. L'opération quotient peut ainsi être utilisée pour synthétiser des spécifications partielles également dans le contexte quantitatif.

Le chapitre 6 est basé sur le travail de Sebastian S. Bauer, Line Juhl, Claus Thrane, Kim G. Larsen, Axel Legay et de l'auteur, travail qui a été présenté au 36ème International Symposium on Mathematical Foundations of Computer Science (MFCS) [BFJ+11] et publié ensuite dans Formal Methods in System Design [BFJ+13].

1.2.7 Chapitre 7, "Théories générales de spécification quantitative avec des systèmes de transition modaux"

Dans le chapitre 7, nous développons un paramétrage général pour les théories de spécification quantitative. En assemblant le travail des chapitres 5 et 6, nous produisons un paramétrage des systèmes de transition modaux où ceux-ci sont étiquetés avec des éléments d'un ensemble partiellement ordonné Spec d'étiquettes de spécification. L'ensemble des étiquettes d'implémentation est alors $\mathsf{Imp} = \{k \in \mathsf{Spec} \mid k' \preccurlyeq k \Rightarrow k' = k\}$, et les implémentations sont des systèmes de transition Imp-étiquetés.

Nous admettons comme entrée une distance de trace abstraite $d : \mathsf{Spec}^\infty \times \mathsf{Spec}^\infty \to \mathbb{L}$, où $\mathbb{L} = (\mathbb{R}_{\geq 0} \cup \{\infty\})^M$, pour un ensemble arbitraire M, est le treillis de fonctions de M dans $\mathbb{R}_{\geq 0} \cup \{\infty\}$. Nous admettons également qu'il existe une fonction d'*itérateur de distance* $F : \mathsf{Spec} \times \mathsf{Spec} \times \mathbb{L} \to \mathbb{L}$ telle que $d(\sigma, \tau) = F(\sigma_0, \tau_0, d(\sigma^1, \tau^1))$, de manière similaire à la caractérisation récursive développée dans le chapitre 5.

Nous pouvons ensuite introduire une *distance d'affinement modal* abstraite entre les états de deux tels systèmes de transition modaux *structurés* (ou SMTS) $\mathcal{S} = (S, s_0, \dashrightarrow_S, \longrightarrow_S)$ et $\mathcal{T} = (T, t_0, \dashrightarrow_T, \longrightarrow_T)$. Celle-ci est alors le

plus petit point fixe dans $\mathbb{L}$ des équations :

$$d_{\mathsf{m}}(s,t) = \max \begin{cases} \sup\limits_{s \overset{k}{\dashrightarrow}_S s'} \inf\limits_{t \overset{\ell}{\dashrightarrow}_T t'} F(k,\ell,d_{\mathsf{m}}(s',t'))\,, \\ \sup\limits_{t \overset{\ell}{\longrightarrow}_T t'} \inf\limits_{s \overset{k}{\longrightarrow}_S s'} F(k,\ell,d_{\mathsf{m}}(s',t'))\,, \end{cases}$$

avec $d_{\mathsf{m}}(\mathcal{S},\mathcal{T}) = d_{\mathsf{m}}(s_0,t_0)$.

Pour la conjonction de deux SMTS, nous introduisons une propriété de *borne conjonctive* sur les étiquettes dans Spec, ce qui implique dans notre Théoreme 7.33 que la conjonction de SMTS est uniformément bornée : dans le sens que la distance d'affinement modal d'un SMTS $\mathcal{U}$ à une conjonction $\mathcal{S} \wedge \mathcal{T}$ admet une borne supérieure qui est une fonction uniforme de distances $d_{\mathsf{m}}(\mathcal{U},\mathcal{S})$ et $d_{\mathsf{m}}(\mathcal{U},\mathcal{T})$. Malheureusement, il se trouve que les opérateurs usuels de conjonction sur étiquettes *ne sont pas* bornés conjonctivement, alors nous proposons une autre propriété de *borne conjonctive faible* qui est vérifiée pour les opérateurs usuels sur étiquettes et montrons dans le théorème 7.35 que ceci implique une propriété similaire pour les conjonctions de SMTS.

Pour la composition structurelle, nous généralisons le travail fait dans le chapitre 6 en introduisant une notion abstraite de composition (partielle) d'étiquettes $\oplus$ sur Spec. En admettant que cet opérateur est récursivement *uniformément borné* dans le sens qu'il existe une fonction $P : \mathbb{L} \times \mathbb{L} \to \mathbb{L}$ telle que :

$$F(k \oplus k', \ell \oplus \ell', P(\alpha,\alpha')) \sqsubseteq_{\mathbb{L}} P(F(k,\ell,\alpha), F(k',\ell',\alpha'))$$

pour tous $k,\ell,k',\ell' \in \mathsf{Spec}$ et $\alpha,\alpha' \in \mathbb{L}$ pour lesquels $k \oplus k'$ et $\ell \oplus \ell'$ sont définis, nous pouvons alors montrer dans le théorème 7.23 que l'implémentabilité indépendante est vérifiée, c'est-à-dire :

$$d_{\mathsf{m}}(\mathcal{S} \| \mathcal{S}', \mathcal{T} \| \mathcal{T}') \sqsubseteq_{\mathbb{L}} P(d_{\mathsf{m}}(\mathcal{S},\mathcal{T}), d_{\mathsf{m}}(\mathcal{S}',\mathcal{T}'))$$

pour tous SMTS $\mathcal{S},\mathcal{T},\mathcal{S}',\mathcal{T}'$. À travers des exemples, nous décrivons plusieurs opérateurs de composition d'étiquettes et nous montrons qu'ils sont uniformément bornés. Nous montrons que l'opérateur quotient / du chapitre 6 admet une généralisation similaire aux SMTS.

Nous montrons également dans le chapitre 7 que l'affinement quantitatif admet une *charactérisation logique*, en étendant le travail du chapitre 3. Nous utilisons la logique Hennessy-Milner classique, avec des formules engendrées par la syntaxe :

$$\phi,\phi_1,\phi_2 := \mathbf{tt} \mid \mathbf{ff} \mid \langle\ell\rangle\phi \mid [\ell]\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \qquad (\ell \in \mathsf{Spec})$$

et avec la sémantique quantitative $S \to \mathbb{L}$ pour un SMTS $\mathcal{S} = (S, s_0, \dashrightarrow_S, \longrightarrow_S)$

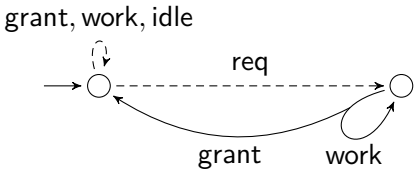

FIGURE 1.9 – DMTS qui correspond à la propriété CTL AG(req ⇒ AX(work AW grant))

donné de la manière suivante :

$$[\![\mathbf{tt}]\!]s = \bot \qquad [\![\mathbf{ff}]\!]s = \top$$
$$[\![(\phi_1 \wedge \phi_2)]\!]s = \max([\![\phi_1]\!]s, [\![\phi]\!]_2 s) \qquad [\![(\phi_1 \vee \phi_2)]\!]s = \min([\![\phi_1]\!]s, [\![\phi_2]\!]s)$$
$$[\![\langle\ell\rangle\phi]\!]s = \inf\{F(k, \ell, [\![\phi]\!]t) \mid s \xrightarrow{k} t, d(k, \ell) \neq \top_{\mathbb{L}}\}$$
$$[\![[\ell]\phi]\!]s = \sup\{F(k, \ell, [\![\phi]\!]t) \mid s \overset{k}{\dashrightarrow} t, d(k, \ell) \neq \top_{\mathbb{L}}\}$$

En écrivant $[\![\phi]\!]\mathcal{S} = [\![\phi]\!]s_0$ nous pouvons alors montrer dans le théorème 7.41 que cette logique est *quantitativement valide* pour la distance d'affinement modal, dans le sens où $[\![\phi]\!]\mathcal{S} \sqsubseteq_{\mathbb{L}} [\![\phi]\!]\mathcal{T} \oplus_{\mathbb{L}} d_{\mathsf{m}}(\mathcal{S}, \mathcal{T})$ pour toutes formules ϕ et tous SMTS $\mathcal{S}$, $\mathcal{T}$. Pour le cas spécial des formules sans disjonction, nous pouvons montrer un résultat complémentaire de complétude dans le théorème 7.42, c'est à dire que $[\![\phi]\!]\mathcal{S} = \sup_{\mathcal{I} \in [\![\mathcal{S}]\!]} [\![\phi]\!]\mathcal{I}$ pour toutes formules ϕ sans disjonction et tous SMTS $\mathcal{S}$; ici nous notons $[\![\mathcal{S}]\!]$ l'ensemble d'implémentations de $\mathcal{S}$.

Le chapitre 7 est basé sur le travail de Sebastian S. Bauer, Claus Thrane, Axel Legay et l'auteur, travail qui a été présenté au 7ème International Computer Science Symposium in Russia (CSR) [BFLT12] et ensuite publié dans Acta Informatica [FL14a].

1.2.8 Chapitre 8, "Spécifications logiques versus spécifications comportementales"

Le chapitre 8 s'écarte du cadre quantitatif de ce mémoire afin d'introduire une généralisation des systèmes de transition modaux qui s'avère plus efficace à la fois dans un cadre qualitatif et également dans le cadre quantitatif du chapitre 9. En extension des travaux de Larsen and Xinxin [LX90], nous définissons un *système de transition modal disjonctifs* (DMTS) comme une structure $\mathcal{D} = (S, S^0, \dashrightarrow, \longrightarrow)$ qui comporte des ensembles finis $S \supseteq S^0$ d'états et d'états initiaux, une relation de transition *may* $\dashrightarrow \subseteq S \times \Sigma \times S$ et une relation de transition *must disjonctive* $\longrightarrow \subseteq S \times 2^{\Sigma \times S}$.

Les DMTS sont donc une généralisation des systèmes de transition modaux dans le sens que le formalisme permet plusieurs (ou zéro) états initiaux et que les transitions *must* sont autorisées à se brancher vers une disjonction d'états

de destination. À titre d'exemple, la figure 1.9 montre un DMTS exprimant la propriété CTL suivante, où on utilise "AW" pour l'opérateur *until faible* :

$$\mathrm{AG}(\mathsf{req} \Rightarrow \mathrm{AX}(\mathsf{work}\ \mathrm{AW}\ \mathsf{grant}))$$

c'est à dire *"à tout instant après avoir exécuté* req*, ni* idle *ni des demandes supplémentaires ne sont autorisées ; que* work *est permis, jusqu'au moment où* grant *est effectué"*. La même propriété peut être exprimée comme un système recursif d'équations dans la logique de Hennessy-Milner [Lar90], comme

$$X = [\mathsf{grant}, \mathsf{idle}, \mathsf{work}]X \wedge [\mathsf{req}]Y$$
$$Y = (\langle \mathsf{work} \rangle Y \vee \langle \mathsf{grant} \rangle X) \wedge [\mathsf{idle}, \mathsf{req}]\mathbf{ff}$$

où la solution est donnée comme étant le plus grand point fixe.

Dans le chapitre 8 nous exposons une équivalence entre DMTS et la logique de Hennessy-Milner avec plus grands points fixes (le *ν-calcul modal*) et aussi avec une extension non-déterministe des *automates d'acceptation* de [Hen85, Rac08]. Cette équivalence permet de passer librement d'un formalisme à l'autre et, plus important, de généraliser les opérations logiques et structurelles des spécifications et d'exposer leur propriétés algébriques.

Nous montrons alors dans le théorème 8.23 que les DMTS (et donc aussi les automates d'acceptation et le ν-calcul modal) admettent des opérations de conjonction et disjonction, *i.e.*, borne inférieure et supérieure, respectivemment, dans la relation d'ordre d'affinenement modal. C'est à dire que des DMTS forment un treillis borné distributif modulo la relation d'équivalence modale.

Ensuite nous généralisons les opérations structurelles de composition et quotient aux DMTS, et nous définissons les quotients $\mathcal{S}_1/\mathcal{S}_2$ aussi pour le cas où $\mathcal{S}_2$ n'est pas déterministe. Le théorème 8.33 montre ensuite que l'opération de quotient a la forme d'une *residuel* à la composition, avec la propriété universelle

$$\mathcal{S}_1 \| \mathcal{S}_2 \leq_{\mathsf{m}} \mathcal{S}_3 \implies \mathcal{S}_2 \leq_{\mathsf{m}} \mathcal{S}_3/\mathcal{S}_1$$

comme avant, mais ici sans restrictions sur les spécifications impliquées.

La combinaison des quatre operations donne aux DMTS une structure de *treillis résiduel commutatif* [JT02] modulo la relation d'équivalence modale. À titre d'exemple, cela implique immédiatement les propriétés suivantes qui peuvent être utilisées dans un calcul de spécifications :

$$\mathcal{S}_1 \| (\mathcal{S}_2/\mathcal{S}_3) \leq_{\mathsf{m}} (\mathcal{S}_1 \| \mathcal{S}_2)/\mathcal{S}_3 \qquad \mathcal{S}_1/\mathcal{S}_2 \leq_{\mathsf{m}} (\mathcal{S}_1 \| \mathcal{S}_3)/(\mathcal{S}_2 \| \mathcal{S}_3)$$
$$(\mathcal{S}_1/\mathcal{S}_2) \| (\mathcal{S}_2/\mathcal{S}_3) \leq_{\mathsf{m}} \mathcal{S}_1/\mathcal{S}_3 \qquad (\mathcal{S}_1/\mathcal{S}_2)/\mathcal{S}_3 \equiv_{\mathsf{m}} (\mathcal{S}_1/\mathcal{S}_3)/\mathcal{S}_2$$
$$\mathcal{S}_1/(\mathcal{S}_2 \| \mathcal{S}_3) \equiv_{\mathsf{m}} (\mathcal{S}_1/\mathcal{S}_2)/\mathcal{S}_3 \qquad \mathcal{S} \| (\mathcal{S}/\mathcal{S}) \equiv_{\mathsf{m}} \mathcal{S}$$

Le chapitre 8 est basé sur le travail de Nikola Beneš, Jan Křetínský, Axel Legay, Louis-Marie Traonouez et l'auteur, travail qui a été présenté au 24ème

International Conference on Concurrency Theory (CONCUR) [BDF+13] et au 11ème International Colloquium on Theoretical Aspects of Computing (ICTAC) [FLT14] et ensuite publié dans Information and Computation [BFK+20].

1.2.9 Chapitre 9, "Compositionnalité pour les spécifications quantitatives"

Le dernier chapitre 9 combine les travaux des chapitres 7 et 8. Il introduit les théories générales de spécification quantitative basées sur les systèmes de transition modaux disjonctifs [LX90] et les automates d'acceptation (non-déterministe) [Hen85, Rac08] d'une part et les distances de trace abstraites d'autre part.

Comme dans le chapitre 7, les étiquettes de spécification sont partiellement ordonnées par une relation $\preceq$ de raffinement d'étiquettes, et les étiquettes d'implémentation sont les étiquettes qui ne peuvent pas être affinées davantage. Nous supposons également qu'il existe des opérateurs partiels de conjonction et de synchronisation sur les étiquettes et travaillons avec des systèmes de transition modaux disjonctifs et des automates d'acceptation étiquetés par des étiquettes de spécification.

Aussi de manière analogue au chapitre 7, nous admettons comme entrée une distance sur des traces de spécification, récursivement donnée, qui prend des valeurs dans un *quantale commutatif* : un treillis complet $\mathbb{L}$ avec une opération commutative $\oplus_{\mathbb{L}}$ qui satisfait une loi de distributivité par rapport aux bornes supérieures. Nous généralisons alors les correspondances entre DMTS, automates d'acceptation et le ν-calcul modal du chapitre 8 dans notre cadre plus abstrait et montrons dans le théorème 9.20 qu'elles respectent des distances d'affinement modal : notant les correspondances par da, dn etc.,

$$
\begin{aligned}
d_{\mathsf{m}}(\mathcal{D}_1, \mathcal{D}_2) &= d_{\mathsf{m}}(da(\mathcal{D}_1), da(\mathcal{D}_2)),\\
d_{\mathsf{m}}(\mathcal{A}_1, \mathcal{A}_2) &= d_{\mathsf{m}}(ad(\mathcal{A}_1), ad(\mathcal{A}_2)),\\
d_{\mathsf{m}}(\mathcal{D}_1, \mathcal{D}_2) &= d_{\mathsf{m}}(dn(\mathcal{D}_1), dn(\mathcal{D}_2)),\\
d_{\mathsf{m}}(\mathcal{N}_1, \mathcal{N}_2) &= d_{\mathsf{m}}(nd(\mathcal{N}_1), nd(\mathcal{N}_2)).
\end{aligned}
$$

Nous nous tournons ensuite vers les propriétés quantitatives des opérations et montrons dans le théorème 9.26 que la disjonction est quantitativement valide et complète dans le sens où $d_{\mathsf{m}}(\mathcal{S}_1 \vee \mathcal{S}_2, \mathcal{S}_3) = \max(d_{\mathsf{m}}(\mathcal{S}_1, \mathcal{S}_3), d_{\mathsf{m}}(\mathcal{S}_2, \mathcal{S}_3))$ pour toutes spécifications $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$. En revanche, la conjonction n'est que quantitatiment valide, pour les mêmes raisons qu'exposé dans le chapitre 6. En admettant une borne uniforme sur la synchronisation d'étiquettes, nous dérivons à nouveau une version quantitative de l'implémentabilité indépendante dans le théorème 9.28. Nous montrons aussi dans le théorème 9.29 qu'en utilisant notre nouvelle définition généralisée du quotient des DMTS on obtient que $d_{\mathsf{m}}(\mathcal{S}_1 \| \mathcal{S}_2, \mathcal{S}_3) = d_{\mathsf{m}}(\mathcal{S}_2, \mathcal{S}_3 / \mathcal{S}_1)$ pour toutes spécifications $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$.

Le chapitre 9 est basé sur le travail de Jan Křetínský, Axel Legay, Louis-Marie Traonouez et l'auteur, travail qui a été présenté au 11ème International

Symposium on Formal Aspects of Component Software (FACS) [FKLT14] et ensuite publié dans Soft Computing [FKLT18].

1.3 Applications

Notre théorie de spécification et de vérification quantitatives trouve des applications concrètes dans la robustesse des systèmes temps-réel, les interactions de fonctionnalités dans les gammes de produits logiciels, la compatibilité des interfaces de services, la séparation de textes et d'autres domaines. Nous présentons ici quatre de ces applications.

1.3.1 Une théorie de spécification robuste pour les automates modaux d'horloges d'événement

L'article [FL12], rédigé par Axel Legay et l'auteur et présenté au 4ème Workshop on Foundations of Interface Technologies, porte sur une application du cadre quantitatif de ce mémoire dans le domaine de spécifications temps-réel. Nous définissons une notion de robustesse pour les spécifications modales d'horloges d'événement (MECS) de [BLPR09, BLPR12].

Nous proposons une nouvelle version de raffinement pour les MECS qui est adéquate pour l'analyse robuste de MECS. Nous exhibons ensuite les propriétés des opérations standards des théories de spécification : conjonction, composition structurale et quotient, par rapport à ce raffinement quantitatif. Nous montrons que la composition structurelle et le quotient ont des propriétés qui sont des généralisations simples de leurs propriétés qualitatives booléennes et peuvent donc être utilisés pour un raisonnement robuste sur MECS. La conjonction, en revanche, n'est généralement pas robuste, mais elle peut être utilisée de manière robuste avec un nouvel opérateur d'élargissement quantitatif.

Les MECS sont des systèmes de transition modaux dans lesquels les transitions *may* et *must* sont étiquetées avec des symboles d'un ensemble Σ et annotées avec des contraintes qui sont utilisées pour activer ou désactiver les transitions en fonction des valeurs des variables. Dans le langage de la section 1.2.7, leur sémantique est donnée comme SMTS sur l'ensemble d'étiquettes de spécification

$$\mathsf{Spec} = (\Sigma \times \{[0,0]\}) \cup (\{\delta\} \times \mathbb{I}) \subseteq (\Sigma \cup \{\delta\}) \times \mathbb{I}.$$

Ici nous notons $\mathbb{I} = \{[x,y] \mid x \in \mathbb{R}_{\geq 0}, y \in \mathbb{R}_{\geq 0} \cup \{\infty\}, x \leq y\}$ l'ensemble d'intervalles fermés avec bornes dans les nombres réels positifs étendus, et $\delta \notin \Sigma$ est un symbole spécial qui indique l'évolution du temps.

L'ordre partiel sur Spec est donné par $(a,[l,r]) \preccurlyeq (a',[l',r'])$ si $a = a'$, $l \geq l'$ et $r \leq r'$ (ce qui implique $[l,r] \subseteq [l',r']$). Les étiquettes d'implémentation sont donc données par $\mathsf{Imp} = \Sigma \times \{0\} \cup \{\delta\} \times \mathbb{R}_{\geq 0}$, et les implémentations sont

des systèmes de transition temporisés habituels avec les transitions discrètes $s \xrightarrow{a,0} s'$ et les transitions de délai $s \xrightarrow{\delta,d} s'$.

Nous utilisons la distance d'avance maximale pour mesurer les différences entre les traces temporisées. Pour la composition structurelle, nous utilisons une synchronisation comme dans le CSP pour les étiquettes et l'intersection d'intervalles de temps ; dans une composition, les contraintes temporelles sont donc des conjonctions des contraintes des composants. Nous montrons ensuite que la composition structurelle est bornée et que la conjonction est faiblement bornée ; l'opérateur quotient satisfait les propriétés usuelles.

1.3.2 Mesure de la similarité globale entre textes

L'article [FBC+14], rédigé par Fabrizio Biondi, Kevin Corre, Cyrille Jégourel, Simon Kongshøj, Axel Legay et l'auteur et présenté au 2ème International Conference on Statistical Language and Speech Processing, contient une application d'une partie de la théorie présentée ici à un problème de traitement statistique du langage naturel. Nous introduisons un nouveau type de distance entre textes et montrons que ceci peut être utilisé pour séparer différents types de texte dans des corpus d'articles scientifiques.

Nous mesurons la similarité de deux textes en utilisant une distance accumulative avec escompte. Étant donné deux textes $A = (a_1, a_2, \ldots, a_{N_A})$ et $B = (b_1, b_2, \ldots, b_{N_B})$, vus comme des séquences finies de mots (et donc dénués de ponctuation), nous commençons en définissant une fonction indicatrice $\delta_{i,j}$, pour $i, j \geq 0$, par

$$\delta_{i,j} = \begin{cases} 0 & \text{si } i \leq N_A, j \leq N_B \text{ et } a_i = b_j \,, \\ 1 & \text{sinon} \,, \end{cases}$$

et après,

$$d(i, j, \lambda) = \sum_{k=0}^{\infty} \lambda^k \delta_{i+k,j+k} \,,$$

pour un facteur d'escompte $\lambda \in \mathbb{R}_{\geq 0}$ avec $\lambda < 1$. Cela mesure à quel point les textes A et B "se ressemblent" en commençant par des tokens a_i dans A et b_j dans B. Cette distance de *concordance de coordonnées* est alors résumée et symétrisée comme suit :

$$d'(A, B, \lambda) = \frac{1}{N_A} \sum_{i=1}^{N_A} \min_{j=1,\ldots,N_B} d(i, j, \lambda)$$
$$d(A, B, \lambda) = \max(d'(A, B, \lambda), d'(B, A, \lambda))$$

Nous avons implémenté ce calcul, puis utilisé notre implémentation pour trier statistiquement différents types d'articles scientifiques. Dans une première expérience nous avons réussi à séparer 42 articles scientifiques de 8

"faux" articles scientifiques générés automatiquement (à l'aide de l'outil SCIgen [1]). Avec un facteur d'escompte très élevé nous obtenons une classification dans laquelle les articles qui partagent les mêmes auteurs ou qui sont autrement similaires sont classés comme l'étant. Dans une deuxième expérience, nous comparons 97 articles scientifiques avec 100 "faux" générés par des méthodes différentes. Encore une fois nous obtenons une classification complète. Pour des facteurs d'escompte élevés, nos classifications sont meilleures que celles obtenues par d'autres travaux utilisant des distances par sac de mots.

1.3.3 Mesure des interactions de comportements entre les fonctionnalités d'une gamme de produits

L'article [AFL15], rédigé par Joanne M. Atlee, Axel Legay et l'auteur et présenté au 3ème IEEE/ACM FME Workshop on Formal Methods in Software Engineering, propose une nouvelle méthode pour mesurer le degré d'interaction des fonctionnalités dans les gammes de produits logiciels.

Nous commençons en introduisant une distance entre les systèmes de transition similaire à la distance accumulative (sans escompte), sauf pour le fait que chaque paire d'états n'est traitée qu'une seule fois. C'est à dire que la fonction qui calcule $d(s, s')$ essaie de faire correspondre chaque transition $s \xrightarrow{a} t$ dans le premier système $\mathcal{S}$ à une transition $s' \xrightarrow{a} t'$ dans le second $\mathcal{S}'$. S'il n'en existe pas, un comportement manquant est détecté et 1 est ajouté au score ; s'il y a des transitions $s' \xrightarrow{a} t'$, la distance est calculée récursivement en prenant la paire t, t' qui correspond le mieux. Une fois qu'une paire d'états a été vérifiée pour les inadéquations de comportement de cette manière, elle est ajoutée à une liste *Passed* d'états qui n'ont pas besoin d'être vérifiés à nouveau.

Nous modélisons les gammes de produits logiciels à l'aide de systèmes de transition aux attributs, qui sont des systèmes de transition dans lesquels les transitions sont conditionnées à la présence ou à l'absence de fonctionnalités distinctes. Un produit est alors simplement un ensemble d'attributs, et un produit p a une interaction comportementale avec une fonctionnalité f dans un système de transition aux attributs $\mathcal{S}$ si la projection sur p de $\mathcal{S}$ et la projection sur p de la projection sur $p \cup \{f\}$ de $\mathcal{S}$ ne sont pas bisimilaires.

Nous généralisons ensuite cette notion à une distance d'interaction comportementale, en utilisant la distance introduite ci-dessus entre systèmes de transition. Nous montrons qu'il s'agit d'une notion féconde pour évaluer le degré d'interactions des fonctionnalités et qu'elle peut être calculée efficacement directement sur le système de transition aux attributs donné, sans recourir aux projections.

1. `http://pdos.csail.mit.edu/scigen/`

1.3.4 Flot de compatibilité : Mesure des interactions des modèles comportementaux

L'article [OFLS17], rédigé par Meriem Ouederni, Axel Legay, Gwen Salaün et l'auteur et présenté au 32ème ACM SIGAPP Symposium on Applied Computing, traite de la vérification de la compatibilité des interfaces de service, en se concentrant sur le niveau du protocole d'interaction.

Vérifier la compatibilité des protocoles d'interaction est une tâche fastidieuse et difficile, même s'il est de la plus haute importance pour éviter les erreurs d'exécution, par exemple les situations de blocage ou les messages sans correspondance. La plupart des approches existantes renvoient un résultat "vrai" ou "faux" pour détecter si les services sont compatibles ou non, mais pour de nombreux sujets une telle réponse qualitative n'est pas très utile. Dans les situations réelles, il y aura rarement une correspondance parfaite, et lorsque les protocoles de service ne sont pas compatibles, il est utile de pouvoir faire la différence entre les services légèrement incompatibles et ceux qui le sont totalement. Notre travail vise à quantifier le degré de compatibilité des interfaces de service d'un point de vue sémantique.

Les incompatibilités sont mesurées entre des systèmes de transition modélisant des interfaces de service, à l'aide d'une version de distance de bisimulation accumulative avec escompte pour laquelle les différences sont propagées à la fois à l'avant et à l'arrière. La distance prend en compte la compatibilité des paramètres et des étiquettes et est définie pour deux scénarios différents, un dans lequel on cherche à faire correspondre tous les messages envoyés et reçus, et un autre, asymétrique, dans lequel l'un des composants peut envoyer et recevoir d'autres messages qui ne sont pas pertinents pour la composition des services.

1.4 Conclusion et perspectives

Nous avons développé une théorie générale de la vérification et de la spécification quantitative. La théorie est indépendante de la manière dont les différences quantitatives sont mesurées et applicable à une large classe de distances utilisées dans la litérature. Le formalisme de spécification quantitative introduit au dernier chapitre 9 est aussi plutôt robuste, admettant des traductions entre plusieurs formalismes de spécification différents, et possède de propriétés algébriques et géométriques solides.

Sur le plan théorique, on pourrait se demander plus généralement ce qu'*est*, exactement, une théorie de spécification. Même s'il existe un certain consensus sur ce point au niveau qualitatif, la manière de l'étendre au monde quantitatif n'est pas claire. Cette question est importante non seulement sur le plan théorique, mais aussi dans les applications, étant donné que les propriétés algébriques d'un formalisme déterminent comment il peut être utilisé dans la pratique.

En relation avec la question ci-dessus, le problème est aussi de savoir comment traiter les transitions *spontanées*. Dans les applications, il est courant de modéliser l'incertitude ou l'ambiguïté avec des transitions spontanées, et celles-ci sont plutôt bien appréhendées dans le cadre qualitatif ; mais encore une fois, on ne sait pas comment les amener au monde quantitatif.

Au niveau des applications, le fait que tous les formalismes traités ici soient basés sur des *systèmes de transition* pose problème. Lorsque l'on considère des applications dans des systèmes temps-réel ou hybrides, ce modèle discret n'est pas suffisant et un traitement du temps continu est nécessaire. Il existe quelques travaux sur les théories de spécification pour les systèmes temps-réel, mais pour les systèmes hybrides ces théories font défaut, et de plus, il n'est pas clair comment les relier aux théories de spécification quantitative que nous avons exposées ici.

Ci-dessous, nous traitons ces questions et problèmes plus en détail et essayons de montrer quelques pistes pour poursuivre les travaux sur ces sujets.

1.4.1 Theories de spécification

Les travaux présentés ici ont soulevé des questions plus fondamentales quant à savoir ce qui exactement *est*, ou *devrait être*, une théorie de spécification. C'est à cela que nous avons tenté de répondre, avec Axel Legay, dans [FL17], présenté au 43ème International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM) et ensuite publié dans Journal of Logical and Algebraic Methods in Programming [FL20b], et dans [FL20a], présenté au ISoLA Symposium 2021.

Nous proposons ici qu'une théorie de spécification pour un ensemble de modèles $\mathfrak{M}$ se compose des ingrédients suivants :

— un ensemble $\mathfrak{S}$ de spécifications ;

— une application $\chi : \mathfrak{M} \to \mathfrak{S}$; et

— un préordre de raffinement $\leq$ sur $\mathfrak{S}$ qui est une relation d'équivalence sur l'image de χ dans $\mathfrak{S}$.

Il s'ensuit que pour tout $M \in \mathfrak{M}$, $\chi(M)$ est la *formule caractéristique* [Pnu85] pour M.

Des opérations logiques sur les spécifications sont alors obtenues en stipulant que $\mathfrak{S}$ forme un *treillis distributif borné* modulo $\equiv$, la relation d'équivalence sur $\mathfrak{S}$ donnée par $S_1 \equiv S_2$ si $S_1 \leq S_2$ et $S_2 \leq S_1$. La composition structurelle et le quotient sont définis par une opération supplémentaire $\|$ sur les spécifications qui transforme $\mathfrak{S}$ en un *treillis résiduel commutatif* (distribué borné). Cela place les théories de spécification dans un contexte algébrique bien compris, voir par exemple [JT02], qui apparaît également en logique linéaire [Gir87] et dans d'autres domaines.

Reste posée la question de comment transférer ce point de vue algébrique au cadre *quantitatif*. Il est clair que l'ordre de raffinement doit être remplacé

par une hémimétrique d sur $\mathfrak{S}$, et aussi que d doit être symétrique sur l'image de χ dans $\mathfrak{S}$; mais nous ne savons pas comment introduire correctement des formules caractéristiques dans ce cadre.

1.4.2 Transitions spontanées

Un autre sujet d'étude serait de savoir comment gérer les *transitions spontanées* dans le cadre quantitatif. Van Glabbeek définit un spectre temps-linéaire–temps-branchant pour les "processus avec pas spontanés" dans [vG93], mais il n'est pas clair comment traduire cela dans un cadre de jeu afin de l'adapter au travail contenu dans le chapitre 5.

Des transitions spontanées sont obtenues chaque fois que deux processus se synchronisent sur des actions internes, elles sont donc importantes du point de vue applicatif. Les avancées récentes sur les approches coalgébriques aux transitions spontanées [Bre15] et sur les jeux de codensité [KKH+19] semblent offrir une voie intéressante.

1.4.3 Applications

Beaucoup de travail reste à faire pour appliquer nos recherches à des systèmes temps-réel, hybrides ou embarqués. Des théories de spécification pour les systèmes temps-réel et probabilistes existent, voir ci-dessous, mais elles ont toutes des déboires avec la robustesse. Pour les systèmes hybrides, aucun travail sur les théories de spécification ne semble disponible. D'une manière générale, le problème avec les systèmes temps-réel et hybrides est que le temps lui-même fournit un mécanisme de synchronisation implicite, de sorte que la compositionnalité est difficile à atteindre pour les systèmes temps-réel et hybrides.

Spécifications modales d'horloges d'événement

Nous avons déjà mentionné les spécifications modales d'horloges d'événement (MECS) dans la section 1.3.1. Introduites dans [BLPR09, BLPR12], celles-ci forment une théorie de spécification pour les automates d'horloges d'événement [AFH99], une sous-classe déterminisable des automates temporisés [AD94], avec comme relation d'équivalence la bisimilarité temporisée. Les modèles et les spécifications sont supposés déterministes, donc $\mathcal{S}_1 \leq \mathcal{S}_2$ si et seulement si $[\![\mathcal{S}_1]\!] \subseteq [\![\mathcal{S}_2]\!]$.

Dans [BLPR12] il est montré que les MECS admettent une conjonction, formant ainsi un demi-treillis modulo $\equiv$. Les auteurs définissent également une composition structurelle et un quotient ; mais le calcul du quotient entraîne une explosion exponentielle d'états. En utilisant la distance d'avance maximale pour mesurer les différences entre les traces temporisées, nous avons montré dans [FL12] comment développer un cadre pour un raisonnement quantitatif robuste sur les MECS.

Automates temporisés entrée/sortie

[DLL+15, DLL+12a] introduisent une théorie de spécification basée sur une variante des automates temporisés entrée/sortie (TIOA) de [KLSV10, KLSV03]. Les modèles et les spécifications sont des TIOA déterministes et prêts à accepter toute entrée ; mais les modèles sont encore plus restreints par des conditions d'urgence de sortie et de progrès indépendant. L'équivalence considérée sur les modèles est la bisimilarité temporelle.

Dans [DLL+15] il est montré que les TIOA admettent une opération de conjonction. L'article introduit également une opération de composition structurelle et un quotient, mais le quotient ne satisfait la propriété

$$\mathcal{S}_1 \| \mathcal{M} \leq \mathcal{S}_3 \Leftrightarrow \mathcal{M} \leq \mathcal{S}_3 / \mathcal{S}_1$$

que pour des *modèles* $\mathcal{M}$ (et des spécifications $\mathcal{S}_1, \mathcal{S}_3$). Aucune théorie de spécification quantitative robuste pour TIOA n'est disponible.

Automates probabilistes abstraits

Les automates probabilistes abstraits (APA), introduits dans [DKL+13, DFLL14], forment une théorie de spécification pour les automates probabilistes [SL95] modulo la bisimilarité probabiliste. Ils s'appuient sur des modèles antérieurs de chaînes de Markov à intervalles (IMC) [DLL+12b], voir aussi [BDF+18, DLP16] pour des travaux connexes.

Dans [DKL+13] il est montré que les APA admettent une opération de conjonction, contrairement aux IMC. Aussi une opération de composition est introduite dans [DKL+13], et les auteurs montrent que la composition de deux APA avec des contraintes d'intervalle (donc des IMC) peut donner un APA avec des contraintes *polynomiales* (pas un IMC) ; mais les APA avec contraintes polynomiales sont fermés sous composition. Aucune théorie de spécification quantitative robuste pour les APA n'est disponible.

1.5 Remerciements

L'auteur tient à remercier tous les collègues impliqués dans les travaux qui constituent la base de ce mémoire. Dans l'ordre alphabétique, il s'agit des personnes suivantes : Sebastian S. Bauer, Munich, Allemagne ; Nikola Beneš, Brno, Tchéquie ; Line Juhl, Aalborg, Danemark ; Jan Křetínský, Munich, Allemagne ; Kim G. Larsen, Aalborg, Danemark ; Axel Legay, Louvain-la-Neuve, Belgique ; Claus Thrane, Copenhague, Danemark ; Louis-Marie Traonouez, Rennes, France.

References

[AB07] Charalambos D. Aliprantis and Kim C. Border. *Infinite Dimensional Analysis : A Hitchhiker's Guide*. Springer, 2007.

[ACH+95] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1) :3–34, 1995.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2) :183–235, 1994.

[AFH99] Rajeev Alur, Limor Fix, and Thomas A. Henzinger. Event-clock automata : A determinizable class of timed automata. *Theoretical Computer Science*, 211(1-2) :253–273, 1999.

[AFL15] Joanne M. Atlee, Uli Fahrenberg, and Axel Legay. Measuring behaviour interactions between product-line features. In *FormaliSE*, pages 20–25. IEEE, 2015.

[AHL+08] Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman, and Andrzej Wąsowski. 20 years of modal and mixed specifications. *Bulletin of the EATCS*, 95 :94–129, 2008.

[ASSB00] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert K. Brayton. Model-checking continous-time Markov chains. *ACM Transactions on Computational Logics*, 1(1) :162–170, 2000.

[AT11] Rajeev Alur and Ashutosh Trivedi. Relating average and discounted costs for quantitative analysis of timed systems. In Chakraborty et al. [CJBF11], pages 165–174.

[BBB+10] Ananda Basu, Saddek Bensalem, Marius Bozga, Benoît Caillaud, Benoît Delahaye, and Axel Legay. Statistical abstraction and model-checking of large heterogeneous systems. In John Hatcliff and Elena Zucca, editors, *FMOODS/FORTE*, volume 6117 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 2010.

[BBLM13] Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. On-the-fly exact computation of bisimilarity distances. In Nir Piterman and Scott A. Smolka, editors, *TACAS*, volume 7795 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2013.

[BDF+13] Nikola Beneš, Benoît Delahaye, Uli Fahrenberg, Jan Křetínský, and Axel Legay. Hennessy-Milner logic with greatest fixed points as a complete behavioural specification theory. In D'Argenio and Melgratti [DM13], pages 76–90.

[BDF+18] Anicet Bart, Benoît Delahaye, Paulin Fournier, Didier Lime, Eric Monfroy, and Charlotte Truchet. Reachability in parametric in-

terval Markov chains using constraints. *Theoretical Computer Science*, 747 :48–74, 2018.

[BFJ+11] Sebastian S. Bauer, Uli Fahrenberg, Line Juhl, Kim G. Larsen, Axel Legay, and Claus Thrane. Quantitative refinement for weighted modal transition systems. In Filip Murlak and Piotr Sankowski, editors, *MFCS*, volume 6907 of *Lecture Notes in Computer Science*, pages 60–71. Springer, 2011.

[BFJ+13] Sebastian S. Bauer, Uli Fahrenberg, Line Juhl, Kim G. Larsen, Axel Legay, and Claus Thrane. Weighted modal transition systems. *Formal Methods in System Design*, 42(2) :193–220, 2013.

[BFK+20] Nikola Beneš, Uli Fahrenberg, Jan Křetínský, Axel Legay, and Louis-Marie Traonouez. Logical vs. behavioural specifications. *Information and Computation*, 271 :104487, 2020.

[BFLM11] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, and Nicolas Markey. Quantitative analysis of real-time systems using priced timed automata. *Communications of the ACM*, 54(9) :78–87, 2011.

[BFLT12] Sebastian S. Bauer, Uli Fahrenberg, Axel Legay, and Claus Thrane. General quantitative specification theories with modalities. In Edward A. Hirsch, Juhani Karhumäki, Arto Lepistö, and Michail Prilutskii, editors, *CSR*, volume 7353 of *Lecture Notes in Computer Science*, pages 18–30. Springer, 2012.

[BJL+12a] Sebastian S. Bauer, Line Juhl, Kim G. Larsen, Axel Legay, and Jiří Srba. Extending modal transition systems with structured labels. *Mathematical Structures in Computer Science*, 22(4) :581–617, 2012.

[BJL+12b] Sebastian S. Bauer, Line Juhl, Kim G. Larsen, Jiří Srba, and Axel Legay. A logic for accumulated-weight reasoning on multi-weighted modal automata. In Tiziana Margaria, Zongyan Qiu, and Hongli Yang, editors, *TASE*, pages 77–84. IEEE, 2012.

[BJS09] Joakim Byg, Kenneth Yrke Jørgensen, and Jiří Srba. TAPAAL : editor, simulator and verifier of timed-arc Petri nets. In Zhiming Liu and Anders P. Ravn, editors, *ATVA*, volume 5799 of *Lecture Notes in Computer Science*, pages 84–89. Springer, 2009.

[BKL+12] Nikola Beneš, Jan Křetínský, Kim G. Larsen, Mikael H. Møller, and Jiří Srba. Dual-priced modal transition systems with time durations. In Nikolaj Bjørner and Andrei Voronkov, editors, *LPAR*, volume 7180 of *Lecture Notes in Computer Science*, pages 122–137. Springer, 2012.

[BLM+11] Patricia Bouyer, Kim G. Larsen, Nicolas Markey, Ocan Sankur, and Claus Thrane. Timed automata can always be made implementable. In Joost-Pieter Katoen and Barbara König, editors,

CONCUR, volume 6901 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2011.

[BLPR09] Nathalie Bertrand, Axel Legay, Sophie Pinchinat, and Jean-Baptiste Raclet. A compositional approach on modal specifications for timed systems. In Karin Breitman and Ana Cavalcanti, editors, *ICFEM*, volume 5885 of *Lecture Notes in Computer Science*, pages 679–697. Springer, 2009.

[BLPR12] Nathalie Bertrand, Axel Legay, Sophie Pinchinat, and Jean-Baptiste Raclet. Modal event-clock specifications for timed component-based design. *Science of Computer Programming*, 77(12) :1212–1234, 2012.

[Bre15] Tomasz Brengos. Weak bisimulation for coalgebras over order enriched monads. *Logical Methods in Computer Science*, 11(2), 2015.

[BvBR98] Marcello M. Bonsangue, Franck van Breugel, and Jan J. M. M. Rutten. Generalized metric spaces : Completion, topology, and powerdomains via the Yoneda embedding. *Theoretical Computer Science*, 193(1-2) :1–51, 1998.

[ČCH+11] Pavol Černý, Krishnendu Chatterjee, Thomas A. Henzinger, Arjun Radhakrishna, and Rohit Singh. Quantitative synthesis for concurrent programs. In Gopalakrishnan and Qadeer [GQ11], pages 243–259.

[ČCHR14] Pavol Černý, Martin Chmelik, Thomas A. Henzinger, and Arjun Radhakrishna. Interface simulation distances. *Theoretical Computer Science*, 560 :348–363, 2014.

[CdAF+06] Krishnendu Chatterjee, Luca de Alfaro, Marco Faella, Thomas A. Henzinger, Rupak Majumdar, and Mariëlle Stoelinga. Compositional quantitative reasoning. In *QEST*, pages 179–188. IEEE Computer Society, 2006.

[CdAHM02] Arindam Chakrabarti, Luca de Alfaro, Thomas A. Henzinger, and Freddy Y. C. Mang. Synchronous and bidirectional component interfaces. In *CAV*, volume 2404 of *Lecture Notes in Computer Science*, pages 414–427, 2002.

[CdAMR10] Krishnendu Chatterjee, Luca de Alfaro, Rupak Majumdar, and Vishwanath Raman. Algorithms for game metrics. *Logical Methods in Computer Science*, 6(3), 2010.

[CE81] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In Dexter Kozen, editor, *Logic of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981.

[ČH11] Pavol Černý and Thomas A. Henzinger. From Boolean to quantitative synthesis. In Chakraborty et al. [CJBF11], pages 149–154.

[CHP11] Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Timed parity games : Complexity and robustness. *Logical Methods in Computer Science*, 7(4), 2011.

[ČHR10] Pavol Černý, Thomas A. Henzinger, and Arjun Radhakrishna. Simulation distances. In Paul Gastin and François Laroussinie, editors, *CONCUR*, volume 6269 of *Lecture Notes in Computer Science*, pages 253–268. Springer, 2010.

[ČHR12] Pavol Černý, Thomas A. Henzinger, and Arjun Radhakrishna. Simulation distances. *Theoretical Computer Science*, 413(1) :21–35, 2012.

[ČHR13] Pavol Černý, Thomas A. Henzinger, and Arjun Radhakrishna. Quantitative abstraction refinement. In Roberto Giacobazzi and Radhia Cousot, editors, *POPL*, pages 115–128. ACM, 2013.

[CJBF11] Samarjit Chakraborty, Ahmed Jerraya, Sanjoy K. Baruah, and Sebastian Fischmeister, editors. *Proceedings of the 11th International Conference on Embedded Software, EMSOFT 2011, part of the Seventh Embedded Systems Week, ESWeek 2011, Taipei, Taiwan, October 9-14, 2011*. ACM, 2011.

[dAFH+05] Luca de Alfaro, Marco Faella, Thomas A. Henzinger, Rupak Majumdar, and Mariëlle Stoelinga. Model checking discounted temporal properties. *Theoretical Computer Science*, 345(1) :139–170, 2005.

[dAFS04] Luca de Alfaro, Marco Faella, and Mariëlle Stoelinga. Linear and branching metrics for quantitative transition systems. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *ICALP*, volume 3142 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 2004.

[dAFS09] Luca de Alfaro, Marco Faella, and Mariëlle Stoelinga. Linear and branching system metrics. *IEEE Transactions on Software Engineering*, 35(2) :258–273, 2009.

[dAH05] Luca de Alfaro and Thomas A. Henzinger. Interface-based design. In Manfred Broy, Johannes Grünbauer, David Harel, and Tony Hoare, editors, *Engineering Theories of Software Intensive Systems*, volume 195 of *NATO Science Series II : Mathematics, Physics and Chemistry*, pages 83–104. Springer, 2005.

[dAHM03] Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. Discounting the future in systems theory. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 1022–1037. Springer, 2003.

[dAMRS07] Luca de Alfaro, Rupak Majumdar, Vishwanath Raman, and Mariëlle Stoelinga. Game relations and metrics. In *LICS*, pages 99–108. IEEE Computer Society, 2007.

[dAMRS08] Luca de Alfaro, Rupak Majumdar, Vishwanath Raman, and Mariëlle Stoelinga. Game refinement relations and metrics. *Logical Methods in Computer Science*, 4(3), 2008.

[Del10] Benoît Delahaye. *Modular Specification and Compositional Analysis of Stochastic Systems*. PhD thesis, Université de Rennes 1, 2010.

[DFLL14] Benoît Delahaye, Uli Fahrenberg, Kim G. Larsen, and Axel Legay. Refinement and difference for probabilistic automata. *Logical Methods in Computer Science*, 10(3), 2014.

[DGJP99] Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labeled Markov systems. In Jos C. M. Baeten and Sjouke Mauw, editors, *CONCUR*, volume 1664 of *Lecture Notes in Computer Science*, pages 258–273. Springer, 1999.

[DGJP04] Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled Markov processes. *Theoretical Computer Science*, 318(3) :323–354, 2004.

[DJGP02] Josee Desharnais, Radha Jagadeesan, Vineet Gupta, and Prakash Panangaden. The metric analogue of weak bisimulation for probabilistic processes. In LICS [LIC02], pages 413–422.

[DKL+13] Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract probabilistic automata. *Information and Computation*, 232 :66–116, 2013.

[DLL+10] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wąsowski. Timed I/O automata : a complete specification theory for real-time systems. In Karl Henrik Johansson and Wang Yi, editors, *HSCC*, pages 91–100. ACM, 2010.

[DLL+12a] Alexandre David, Kim G. Larsen, Axel Legay, Mikael H. Møller, Ulrik Nyman, Anders P. Ravn, Arne Skou, and Andrzej Wąsowski. Compositional verification of real-time systems using Ecdar. *International Journal on Software Tools for Technology Transfer*, 14(6) :703–720, 2012.

[DLL+12b] Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wąsowski. Consistency and refinement for interval Markov chains. *Journal of Logic and Algebraic Programming*, 81(3) :209–226, 2012.

[DLL+15] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, Louis-Marie Traonouez, and Andrzej Wąsowski. Real-time specifications. *International Journal on Software Tools for Technology Transfer*, 17(1) :17–45, 2015.

[DLP16] Benoît Delahaye, Didier Lime, and Laure Petrucci. Parameter synthesis for parametric interval Markov chains. In Barbara Jobstmann and K. Rustan M. Leino, editors, *VMCAI*, volume 9583 of *Lecture Notes in Computer Science*, pages 372–390. Springer, 2016.

[DLT08] Josée Desharnais, François Laviolette, and Mathieu Tracol. Approximate analysis of probabilistic processes. In *QEST*, pages 264–273. IEEE Computer Society, 2008.

[DM13] Pedro R. D'Argenio and Hernán C. Melgratti, editors. *CONCUR 2013 - Concurrency Theory - 24th International Conference, CONCUR 2013, Buenos Aires, Argentina, August 27-30, 2013. Proceedings*, volume 8052 of *Lecture Notes in Computer Science*. Springer, 2013.

[EM79] Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8 :109–113, 1979.

[Fah22] Uli Fahrenberg. *A Generic Approach to Quantitative Verification*. Habilitation thesis, Paris-Saclay University, 2022. `https://arxiv.org/abs/2204.11302`.

[FBC+14] Uli Fahrenberg, Fabrizio Biondi, Kevin Corre, Cyrille Jégourel, Simon Kongshøj, and Axel Legay. Measuring global similarity between texts. In Laurent Besacier, Adrian Horia Dediu, and Carlos Martín-Vide, editors, *SLSP*, volume 8791 of *Lecture Notes in Computer Science*, pages 220–232. Springer, 2014.

[FGD+11] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. SpaceEx : Scalable verification of hybrid systems. In Gopalakrishnan and Qadeer [GQ11], pages 379–395.

[FH07] Martin Fränzle and Christian Herde. HySAT : An efficient proof engine for bounded model checking of hybrid systems. *Formal Methods in System Design*, 30(3) :179–198, 2007.

[FJLS11] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jiří Srba. Energy games in multiweighted automata. In Antonio Cerone and Pekka Pihlajasaari, editors, *ICTAC*, volume 6916 of *Lecture Notes in Computer Science*, pages 95–115. Springer, 2011.

[FKLT14] Ulrich Fahrenberg, Jan Křetínský, Axel Legay, and Louis-Marie Traonouez. Compositionality for quantitative specifications. In

Ivan Lanese and Eric Madelaine, editors, *FACS*, volume 8997 of *Lecture Notes in Computer Science*, pages 306–324. Springer, 2014.

[FKLT18] Uli Fahrenberg, Jan Křetínský, Axel Legay, and Louis-Marie Traonouez. Compositionality for quantitative specifications. *Soft Computing*, 22(4) :1139–1158, 2018.

[FL12] Uli Fahrenberg and Axel Legay. A robust specification theory for modal event-clock automata. In Sebastian S. Bauer and Jean-Baptiste Raclet, editors, *FIT*, volume 87 of *Electronic Proceedings in Theoretical Computer Science*, pages 5–16, 2012.

[FL14a] Uli Fahrenberg and Axel Legay. General quantitative specification theories with modal transition systems. *Acta Informatica*, 51(5) :261–295, 2014.

[FL14b] Uli Fahrenberg and Axel Legay. The quantitative linear-time–branching-time spectrum. *Theoretical Computer Science*, 538 :54–69, 2014.

[FL17] Uli Fahrenberg and Axel Legay. A linear-time-branching-time spectrum of behavioral specification theories. In Bernhard Steffen, Christel Baier, Mark van den Brand, Johann Eder, Mike Hinchey, and Tiziana Margaria, editors, *SOFSEM*, volume 10139 of *Lecture Notes in Computer Science*, pages 49–61. Springer, 2017.

[FL20a] Uli Fahrenberg and Axel Legay. Behavioral specification theories : An algebraic taxonomy. In Tiziana Margaria and Bernhard Steffen, editors, *ISoLA*, volume 12476 of *Lecture Notes in Computer Science*, pages 262–274. Springer, 2020.

[FL20b] Uli Fahrenberg and Axel Legay. A linear-time-branching-time spectrum for behavioral specification theories. *Journal of Logic and Algebraic Methods in Programming*, 110, 2020.

[FLT09] Uli Fahrenberg, Kim G. Larsen, and Claus Thrane. A quantitative characterization of weighted Kripke structures in temporal logic. In *MEMICS*, 2009. Best paper award.

[FLT10] Uli Fahrenberg, Kim G. Larsen, and Claus Thrane. A quantitative characterization of weighted Kripke structures in temporal logic. *Computing and Informatics*, 29(6+) :1311–1324, 2010.

[FLT11] Uli Fahrenberg, Axel Legay, and Claus Thrane. The quantitative linear-time–branching-time spectrum. In Supratik Chakraborty and Amit Kumar, editors, *FSTTCS*, volume 13 of *LIPIcs*, pages 103–114. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011.

[FLT14] Uli Fahrenberg, Axel Legay, and Louis-Marie Traonouez. Structural refinement for the modal nu-calculus. In Gabriel Ciobanu

and Dominique Méry, editors, *ICTAC*, volume 8687 of *Lecture Notes in Computer Science*, pages 169–187. Springer, 2014.

[FPP05] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for Markov decision processes with infinite state spaces. In *UAI*, pages 201–208. AUAI Press, 2005.

[FT11] Uli Fahrenberg and Stavros Tripakis, editors. *Formal Modeling and Analysis of Timed Systems - 9th International Conference*, volume 6919 of *Lecture Notes in Computer Science*. Springer, 2011.

[FTL11] Uli Fahrenberg, Claus Thrane, and Kim G. Larsen. Distances for weighted transition systems : Games and properties. In Mieke Massink and Gethin Norman, editors, *QAPL*, volume 57 of *Electronic Proceedings in Theoretical Computer Science*, pages 134–147, 2011.

[GH94] Stephen Gilmore and Jane Hillston. The PEPA workbench : A tool to support a process algebra-based approach to performance modelling. In Günter Haring and Gabriele Kotsis, editors, *CPE*, volume 794 of *Lecture Notes in Computer Science*, pages 353–368. Springer, 1994.

[GHJ01] Patrice Godefroid, Michael Huth, and Radha Jagadeesan. Abstraction-based model checking using modal transition systems. In Larsen and Nielsen [LN01], pages 426–440.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50 :1–102, 1987.

[Gir10] Antoine Girard. Synthesis using approximately bisimilar abstractions : Time-optimal control problems. In *CDC*, pages 5893–5898. IEEE, 2010.

[GLLS05] Orna Grumberg, Martin Lange, Martin Leucker, and Sharon Shoham. Don't know in the μ-calculus. In *VMCAI*, volume 3385 of *Lecture Notes in Computer Science*, pages 233–249. Springer, 2005.

[GLMR05] Guillaume Gardey, Didier Lime, Morgan Magnin, and Olivier H. Roux. Romeo : A tool for analyzing time Petri nets. In Kousha Etessami and Sriram K. Rajamani, editors, *CAV*, volume 3576 of *Lecture Notes in Computer Science*, pages 418–423. Springer, 2005.

[GLS08] Alexander Gruler, Martin Leucker, and Kathrin D. Scheidemann. Modeling and model checking software product lines. In Gilles Barthe and Frank S. de Boer, editors, *FMOODS*, volume 5051 of *Lecture Notes in Computer Science*, pages 113–131. Springer, 2008.

[GP07] Antoint Girard and George J. Pappas. Approximation metrics for discrete and continuous systems. *IEEE Transactions on Automatic Control*, 52(5) :782–798, 2007.

[GQ11] Ganesh Gopalakrishnan and Shaz Qadeer, editors. *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*. Springer, 2011.

[Hal00] Robert J. Hall. Feature interactions in electronic mail. In Muffy Calder and Evan H. Magill, editors, *FIW*, pages 67–82. IOS Press, 2000.

[Han93] Hans-Michael Hanisch. Analysis of place/transition nets with timed arcs and its application to batch process control. In Marco Ajmone Marsan, editor, *ATPN*, volume 691 of *Lecture Notes in Computer Science*, pages 282–299. Springer, 1993.

[Hen85] Matthew Hennessy. Acceptance trees. *Journal of the ACM*, 32(4) :896–928, 1985.

[HHWT97] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. HYTECH : A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1-2) :110–122, 1997.

[Hil96] Jane Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[HJ94] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5) :512–535, 1994.

[HM85] Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1) :137–161, 1985.

[HMP05] Thomas A. Henzinger, Rupak Majumdar, and Vinayak S. Prabhu. Quantifying similarities between timed systems. In Paul Pettersson and Wang Yi, editors, *FORMATS*, volume 3829 of *Lecture Notes in Computer Science*, pages 226–241. Springer, 2005.

[HNSY94] Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111(2) :193–244, 1994.

[JLS12] Line Juhl, Kim G. Larsen, and Jirí Srba. Modal transition systems with weight intervals. *Journal of Logic and Algebraic Programming*, 81(4) :408–421, 2012.

[JT02] Peter Jipsen and Constantine Tsinakis. A survey of residuated lattices. In *Ordered algebraic structures*, volume 7, pages 19–56. Kluwer Acad. Publ., 2002.

[KKH+19] Yuichi Komorida, Shin-ya Katsumata, Nick Hu, Bartek Klin, and Ichiro Hasuo. Codensity games for bisimilarity. In *LICS*, pages 1–13. IEEE, 2019.

[KLSV03] Dilsun Kirli Kaynar, Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. Timed I/O automata : A mathematical framework for modeling and analyzing real-time systems. In *RTSS*, pages 166–177. IEEE Computer Society, 2003.

[KLSV10] Dilsun Kirli Kaynar, Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. *The Theory of Timed I/O Automata*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, second edition, 2010.

[KNP02] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Probabilistic symbolic model checking with prism : A hybrid approach. In Joost-Pieter Katoen and Perdita Stevens, editors, *TACAS*, volume 2280 of *Lecture Notes in Computer Science*, pages 52–66. Springer, 2002.

[Koy90] Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4) :255–299, 1990.

[Lar89] Kim G. Larsen. Modal specifications. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1989.

[Lar90] Kim G. Larsen. Proof systems for satisfiability in Hennessy-Milner logic with recursion. *Theoretical Computer Science*, 72(2&3) :265–288, 1990.

[Law73] F. William Lawvere. Metric spaces, generalized logic, and closed categories. *Rendiconti del seminario matématico e fisico di Milano*, XLIII :135–166, 1973.

[LFT11] Kim G. Larsen, Uli Fahrenberg, and Claus Thrane. Metrics for weighted transition systems : Axiomatization and complexity. *Theoretical Computer Science*, 412(28) :3358–3369, 2011.

[LIC02] *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*. IEEE Computer Society, 2002.

[LLTW11] Kim G. Larsen, Axel Legay, Louis-Marie Traonouez, and Andrzej Wąsowski. Robust specification of real time components. In Fahrenberg and Tripakis [FT11], pages 129–144.

[LMP12] Kim G. Larsen, Radu Mardare, and Prakash Panangaden. Taking it to the limit : Approximate reasoning for Markov processes. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *MFCS*, volume 7464 of *Lecture Notes in Computer Science*, pages 681–692. Springer, 2012.

[LN01] Kim G. Larsen and Mogens Nielsen, editors. *CONCUR 2001 - Concurrency Theory, 12th International Conference, Aalborg, Denmark, August 20-25, 2001, Proceedings*, volume 2154 of *Lecture Notes in Computer Science*. Springer, 2001.

[LPY97] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1(1-2) :134–152, 1997.

[LT89] Nancy Lynch and Mark R. Tuttle. An introduction to input/output automata. *CWI-Quarterly*, 2(3), 1989.

[LX90] Kim G. Larsen and Liu Xinxin. Equation solving using modal transition systems. In *LICS*, pages 108–117. IEEE Computer Society, 1990.

[MF76] Philip M. Merlin and David J. Farber. Recoverability of communication protocols–implications of a theoretical study. *IEEE Transactions on Communications*, 24(9) :1036 – 1043, 1976.

[Mun00] James R. Munkres. *Topology*. Prentice-Hall, 2000.

[Nym08] Ulrik Nyman. *Modal Transition Systems as the Basis for Interface Theories and Product Lines*. PhD thesis, Aalborg University, 2008.

[OFLS17] Meriem Ouederni, Uli Fahrenberg, Axel Legay, and Gwen Salaün. Compatibility flooding : measuring interaction of services interfaces. In Ahmed Seffah, Birgit Penzenstadler, Carina Alves, and Xin Peng, editors, *SAC*, pages 1334–1340. ACM, 2017.

[Pan09] Prakash Panangaden. *Labelled Markov Processes*. Imperial College Press, 2009.

[Pnu85] Amir Pnueli. Linear and branching structures in the semantics and logics of reactive systems. In Wilfried Brauer, editor, *ICALP*, volume 194 of *Lecture Notes in Computer Science*, pages 15–32. Springer, 1985.

[QFD11] Jan-David Quesel, Martin Fränzle, and Werner Damm. Crossing the bridge between similar games. In Fahrenberg and Tripakis [FT11], pages 160–176.

[Rac08] Jean-Baptiste Raclet. Residual for component specifications. *Electronic Notes in Theoretical Computer Science*, 215 :93–110, 2008.

[RLS06] Jacob Illum Rasmussen, Kim G. Larsen, and K. Subramani. On using priced timed automata to achieve optimal scheduling. *Formal Methods in System Design*, 29(1) :97–114, 2006.

[Sif11] Joseph Sifakis. A vision for computer science – the system perspective. *Central European Journal of Computer Science*, 1(1) :108–116, 2011.

[SL94] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. In Bengt Jonsson and Joachim Parrow, editors, *CONCUR*, volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer, 1994.

[SL95] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. *Nord. J. Comput.*, 2(2) :250–273, 1995.

[SPE] SPEEDS : Speculative and Exploratory Design in Systems Engineering. `http://www.speeds.eu.com/`.

[Ste94] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[Str] STREP COMBEST : Component-based embedded systems design techniques. `http://www.combest.eu/home/`.

[TFL08] Claus Thrane, Uli Fahrenberg, and Kim G. Larsen. Quantitative simulations of weighted transition systems. In *NWPT*, 2008.

[TFL10] Claus Thrane, Uli Fahrenberg, and Kim G. Larsen. Quantitative analysis of weighted transition systems. *Journal of Logic and Algebraic Programming*, 79(7) :689–703, 2010.

[Thr11] Claus Thrane. *Quantitative Models and Analysis For Reactive Systems*. PhD thesis, Aalborg University, 2011.

[vB96] Franck van Breugel. A theory of metric labelled transition systems. *Annals of the New York Academy of Sciences*, pages 69–87, 1996.

[vB01] Franck van Breugel. An introduction to metric semantics : Operational and denotational models for programming and specification languages. *Theoretical Computer Science*, 258(1-2) :1–98, 2001.

[vB05] Franck van Breugel. A behavioural pseudometric for metric labelled transition systems. In Martín Abadi and Luca de Alfaro, editors, *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 141–155. Springer, 2005.

[vBW01] Franck van Breugel and James Worrell. An algorithm for quantitative verification of probabilistic transition systems. In Larsen and Nielsen [LN01], pages 336–350.

[vBW05] Franck van Breugel and James Worrell. A behavioural pseudometric for probabilistic transition systems. *Theoretical Computer Science*, 331(1) :115–142, 2005.

[vBW06] Franck van Breugel and James Worrell. Approximating and computing behavioural distances in probabilistic transition systems. *Theoretical Computer Science*, 360(1-3) :373–385, 2006.

[vG93] Rob J. van Glabbeek. The linear time - branching time spectrum II. In Eike Best, editor, *CONCUR*, volume 715 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 1993.

[vG01] Rob J. van Glabbeek. The linear time - branching time spectrum I. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, Chapter 1, pages 3–99. Elsevier, 2001.

[WME93] Farn Wang, Aloysius K. Mok, and E. Allen Emerson. Symbolic model checking for distributed real-time systems. In Jim Woodcock and Peter Gorm Larsen, editors, *FME*, volume 670 of *Lecture Notes in Computer Science*, pages 632–651. Springer, 1993.

[ZG09] Gang Zheng and Antoine Girard. Bounded and unbounded safety verification using bisimulation metrics. In Rupak Majumdar and Paulo Tabuada, editors, *HSCC*, volume 5469 of *Lecture Notes in Computer Science*, pages 426–440. Springer, 2009.

[ZP96] Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2) :343–359, 1996.