JOSEPH ASSOURAMOU

Processus de Markov étiquetés et Systèmes Hybrides probabilistes

Mémoire présenté
à la Faculté des études supérieures et postdoctorales de l'Université Laval
dans le cadre du programme de maîtrise en informatique
pour l'obtention du grade de Maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL
QUÉBEC

2012

Résumé

Dans ce mémoire, nous comparons deux modèles de processus probabilistes évoluant dans un environnement continu. Les processus de Markov étiquetés sont des systèmes de transitions pour lesquels l'ensemble des états est non-dénombrable, mais qui évoluent de manière discrète dans le temps. Les mesures de probabilité définies sur l'ensemble des états peuvent avoir un support infini. Les processus hybrides sont une combinaison d'un processus à espace d'états continu qui évolue de manière continue dans le temps et une composante discrète qui intervient pour contrôler l'évolution. Les extensions probabilistes des processus hybrides présentes dans la littérature restreignent le comportement probabiliste à la composante discrète. Nous utilisons deux exemples de systèmes, un avion et un bateau, pour faire ressortir les divergences entre les deux modèles ainsi que leurs limitations, et nous définissons une généralisation qui peut modéliser fidèlement ces exemples. Nous avons également pu montrer, dans un article publié dans un atelier international, comment utiliser, dans le contexte probabiliste, la «substitution d'horloge» et l'«approximation par portrait» qui sont des techniques proposées par Henzinger et al. pour les processus non probabilistes. Ces techniques permettent, sous certaines conditions, de définir un processus probabiliste rectangulaire à partir d'un qui est non rectangulaire, rendant ainsi possible la vérification formelle de toute classe de système hybride probabiliste.

Abstract

We compare two models of processes involving uncountable space. Labelled Markov processes are probabilistic transition systems that can have uncountably many states, but still make discrete time steps. The probability measures on the state space may have uncountable support and a tool has been developed for verification of such systems. Hybrid processes are a combination of a continuous space process that evolves continuously with time and of a discrete component, such as a controller. Existing extensions of Hybrid processes with probability restrict the probabilistic behavior to the discrete component. We have also shown, in a paper, how to compute for probabilistic hybrid systems, the clock approximation and linear phase-portrait approximation that have been proposed for non probabilistic processes by Henzinger et al. The techniques permit, under some conditions, to define a rectangular probabilistic process from a non rectangular one, hence allowing the model-checking of any class of systems. To highlight the differences between Labelled Markov processes and probabilistic hybrid systems, we use two examples, the ones of a boat and an aircraft, and we define a generalization of both that can model all the features of those examples.

Remerciements

Avant tout, je tiens à remercier ma directrice de recherche, Josée Desharnais, pour son support exceptionnel durant le temps de ma maîtrise. Son enthousiasme pour ce sujet et sa confiance en mes habiletés m'ont été très précieux. De plus, sa patience, son écoute et ses nombreuses lectures et corrections de mes textes m'ont permis de beaucoup apprendre et de développer le goût à la rédaction. Merci Josée.

Ensuite, je tiens à remercier Pierre Marchand et sa famille pour leur soutien et leur grande attention.

Je voudrais également remercier mes amis et mes collègues qui m'ont aidé de différentes façons, de par leur affection et leurs encouragements entre autres.

Enfin, j'aimerais remercier ma mère, ma soeur, mes frères, sans oublier ma conjointe et mon fils pour leur support et leurs encouragements.

À Lui, qui a toujours veillé sur moi.

«Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world, and all there ever will be to know and understand.»
Albert Einstein

Table des matières

Table des figures

Chapitre 1

Introduction

L'automatisation, consistant à seconder l'humain dans la réalisation de ses tâches par un système informatique, est plus que jamais d'actualité. Du fait que les besoins en informatisation augmentent, le nombre et la complexité des systèmes grandissent de la même manière. C'est ainsi que dans le domaine de l'aéronautique, les avions sont, depuis une décennie, équipés de systèmes de guidage automatique destinés à remplacer les pilotes lors des vols. De la même façon, en médecine, des systèmes ont été créés pour permettre et faciliter les interventions chirurgicales à distance. Cette omniprésence des systèmes informatiques dans l'environnement de l'humain a donc suscité un intérêt de taille afin d'améliorer leur performance et leur fiabilité. Dans cette optique, plusieurs techniques ont été mises au point en vue d'étudier et de mieux cerner le comportement de systèmes. Une des techniques les plus connues est la *vérification formelle* qui consiste en l'utilisation des lois mathématiques dans la description du comportement de systèmes.

Étant donné que les systèmes ont chacun leurs caractéristiques pouvant être ou non partagées avec d'autres, plusieurs classes et sous-classes de systèmes ont dès lors vu le jour. Ainsi, on parle de *systèmes discrets* pour désigner les systèmes qui mettent en jeu des informations n'étant prises en compte qu'en des unités de temps données. Un exemple de système discret est le système d'allumage d'une lampe électrique où on ne tient compte que du fait que la lampe est allumée ou éteinte ou la barrière d'un passage à niveau où celle-ci est soit levée, soit abaissée. De même, on distingue *les systèmes à temps réel* ou *systèmes réactifs* qui regroupent les systèmes fonctionnant de manière continue. On les retrouve, entre autres, au niveau de la commande et du contrôle de systèmes, du contrôle de processus, du contrôle aérien, des systèmes de défense et des systèmes embarqués. On regroupe les systèmes à temps réel en plusieurs catégories selon leur comportement.

Certains systèmes à temps réel fonctionnent seulement en continu. D'une manière générale, ces derniers ne sont soumis à aucune contrainte de temps lors de leur fonctionnement. C'est le cas d'un robinet dont le débit varie durant son ouverture. D'autres systèmes, par contre, font intervenir une composante continue et une composante discrète, combinant ainsi un comportement continu avec un comportement discret. La composante discrète intervient pour jouer le rôle de contrôleur pour d'une part régulariser l'évolution des variables physiques et d'autre part assurer les changements d'états. Du fait que ces systèmes font intervenir deux comportements différents, on les appelle des *systèmes hybrides*. Contrairement aux systèmes fonctionnant uniquement en continu, les systèmes hybrides sont des systèmes dont le fonctionnement est étroitement lié à des contraintes de temps, tel qu'un radiateur sujet à la contrainte «une minute après son allumage, le radiateur ne doit pas pouvoir être éteint», ou encore un avion en déplacement soumis à la condition «pendant les deux minutes après son décollage, si le pilote fait pivoter l'avion, celui-ci s'écrasera». Plus concrètement, considérons l'exemple d'une voiture où la composante discrète est la boîte à vitesse automatique vu sa nature numérique (les choix des vitesse étant *1, 2, 3, 4, 5, Marche arrière* dans la plupart des cas) tandis que la composante continue est représentée par le déplacement de la voiture. En deuxième vitesse, le déplacement de la voiture peut se faire à la vitesse $v = 50km/h$, soit suivant l'équation différentielle $\frac{dx}{dt} = 50$, en considérant la loi de la cinématique selon laquelle $v = \frac{dx}{dt}$. Le comportement discret est représenté par un système de transitions fini alors que les composantes continues sont représentées par un ensemble fini de variables réelles [26, 25, 2].

À chaque type de système, un *modèle formel* décrivant exactement les comportements possibles des systèmes est associé. Ainsi, les systèmes à temps discret sont représentés par les systèmes de transitions tandis que les automates hybrides de Alur et al. [2, 36] sont utilisés pour modéliser les systèmes hybrides. Ainsi, un modèle est généralement conçu à la mesure des caractéristiques propres à des systèmes donnés et qui permettent de les distinguer d'autres. De plus, ces caractéristiques offrent une base de connaissances pour évaluer et comparer les classes de systèmes entre elles, en vue de formuler des résultats sur plusieurs plans dont l'*expressivité* et le *langage*. L'une de ces caractéristiques est la nature de l'espace d'états, c'est-à-dire la nature de l'ensemble des valeurs que peuvent prendre les paramètres physiques d'un système. La nature d'un espace d'états peut être soit *discrète* soit *continue*. Contrairement à un espace d'états discret, l'espace d'états continu est un sous-ensemble de l'ensemble des réels, autrement dit, les valeurs physiques appartiennent à un intervalle continu comme $[0, 1]$. Lorsque l'espace d'états d'un système est discret, on dit de celui-ci qu'il est un système à *espace d'états discret*, tandis qu'il est un système à *espace d'états continu* lorsque son espace d'états est continu. Parmi les systèmes fonctionnant de manière discrète, certains sont à espace d'états discret et d'autres à espace d'états continu. Les systèmes à temps réel, quant à eux, sont tous à

espace d'états continu.

Une tendance dans le domaine de la vérification formelle est celle des systèmes probabilistes qui ont fait l'objet de nombreux travaux de recherche dont ceux de Desharnais [14, 12, 16, 17] et Sproston [41, 40, 42]. Il s'agit de systèmes pouvant avoir un comportement incertain lors de leur fonctionnement. Cette incertitude est quantifiée et intervient pour préciser la probabilité que le système réponde à une action demandée par l'utilisateur. Sur de tels systèmes, on s'intéresse à vérifier la validité de certaines assertions comme «le radiateur doit toujours pouvoir être éteint dans plus de 90% des cas» ou «si le pilote fait pivoter l'avion, il y a plus de 20% de chance que celui-ci s'écrase». Plus concrètement, considérons le système d'allumage présenté un peu plus haut en supposant cette fois-ci que lorsque l'utilisateur essaie d'allumer la lampe, dans 10% des cas, celle-ci ne s'allumera pas. Le système d'allumage se comporte dans ce cas comme un système probabiliste puisqu'il y a une incertitude dans l'exécution de l'action *allumer*. Comme dans le cas non probabiliste, il existe une hiérarchie de classes et donc de modèles probabilistes. Ainsi, on distingue, entre autres, les systèmes de transitions probabilistes dédiés à la modélisation de systèmes probabilistes discrets et les automates temporisés probabilistes pour représenter des systèmes probabilistes temps réel [33, 10].

Parmi toutes les classes de systèmes existantes, deux modèles de représentation formelle ayant des affinités avec les systèmes de transitions probabilistes d'une part et les automates hybrides d'autre part nous intéressent. Chacune d'elles repose sur la théorie des chaînes de Markov, et par conséquent, respecte la *propriété de Markov* : *pour tout état, le choix de l'état suivant est fait suivant une distribution de probabilité.*

Le premier modèle de représentation est celui des Processus de Markov étiquetés ou LMP (de l'anglais *Labelled Markov Processes*) qui a été initié par Blute et al [12]. Comme les systèmes de transitions probabilistes, les processus de Markov étiquetés modélisent les systèmes probabilistes à temps discret. De plus, cette plateforme permet de supporter des systèmes ayant un espace d'états continu et des distributions complexes caractérisant les transitions. Plusieurs exemples de systèmes ont été étudiés et modélisés avec le formalisme des LMP. Parmi ces systèmes, on distingue la machine à café, le thermostat [38] et d'autres de nature plus théorique. D'une manière générale, les LMP modélisent des systèmes probabilistes qui effectuent des transitions sans délai et ayant un espace d'états continu. Les distributions de probabilité dans ces systèmes peuvent être discrètes, ou définies grâce à des lois de probabilité continues (loi Uniforme, loi Normale, Poisson, Gamma, etc.) ou exprimées par des fonctions de densité comme la fonction τ_a tirée d'un exemple de [14] :

$$\tau_a : \mathbb{R} \to [0,1] \text{ telle que } \tau_a(x,[u,v]) = \int_u^v \exp(-(x-x_0)^2)dx.$$

L'outil développé par Richard [38], CISMO, permet de modéliser et de vérifier formellement ces systèmes probabilistes à espace d'états continu. Le langage donné en entrée à l'outil est un LMP dont les transitions probabilistes sont des combinaisons de distributions bien connues dont les valeurs sont stockées dans une table. L'approche utilisée pour la vérification est celle exploratoire c'est-à-dire que tous les états sont parcourus et sont pris en compte dans les calculs.

Le second modèle, quant à lui, est celui des automates hybrides probabilistes. Il s'agit de la version probabiliste des automates hybrides. Les automates hybrides sont dédiés à la modélisation formelle des systèmes hybrides. Ils ont fait l'objet de plusieurs travaux, notamment ceux de Alur et al. [2], Henzinger et al. [29, 28, 1] et Kotini et al. [25]. Plusieurs outils de vérification formelle des automates hybrides ont été créés dont HyTech qui a été développé par le groupe de recherche de Henzinger [27, 26]. Dans le cas probabiliste, les automates hybrides ont également été étudiés notamment par Sproston [41, 40, 42] et Zhang et al. [23]. Plusieurs algorithmes s'appliquant aux chaînes de Markov [5] ont été également développés pour la vérification de propriétés exprimées dans des logiques temporelles [24] comme PBTL [9, 42]. Un exemple d'une telle propriété est «pendant les deux minutes aprés son décollage, si le pilote fait pivoter l'avion, il y a plus de 20% de chance que celui-ci s'écrase» ou «dans 50% des cas, la température de la chambre dépasse dix degrés après son allumage». Des outils de vérification des systèmes hybrides probabilistes ont également été développés. C'est le cas de ProHVer (Probabilistic Hybrid automata Verifier), développé par le groupe de recherche de Zhang et al. [23], qui prend en entrée un automate hybride probabiliste et, grâce à des méthodes d'approximation bien définies [3, 37, 22], construit une abstraction fidèle sur laquelle la vérification formelle peut être effectuée.

Notre but, dans ce document, est de comparer deux modèles de systèmes à espace d'états continu : les LMP tels que définis par Desharnais [14] et les automates hybrides probabilistes de Sproston [41, 40, 42]. Durant des années, ces deux modèles de processus continus ont évolués indépendemment. Pour chacun d'eux, plusieurs notions telles que la bisimulation et la simulation, des logiques, des techniques de vérification formelle et des outils ont été développés au sein de différentes communautés de recherche. Ces modèles font face à des défis similaires, puisque leur principale complexité est d'évidence leur nature continue. Par exemple, des notions d'approximation ont été développées des deux côtés [7, 16, 26, 23]. Cependant, aucune comparaison systématique entre les LMP et les automates hybrides probabilistes n'a été faite jusque là, et il n'est pas évident à première vue d'affirmer qu'un modèle de systèmes à temps continu ne peut englober un modèle de systèmes à temps discret. De plus, étant donné que CISMO,

l'outil de vérification formelle des LMP, prend en entrée un langage fini, on pourrait se demander si cette finitude a un quelconque lien avec la composante discrète des automates hybrides. Il faut également préciser que les automates hybrides probabilistes ont été définis et étudiés avec des distributions de probabilité discrètes, ce qui pourrait limiter les systèmes modélisés à un sous-ensemble de systèmes hybrides probabilistes. Si les deux modèles ont évolué de manière indépendante, cela voudrait-il alors dire qu'ils sont conçus à des fins différentes et, par conséquent, qu'il n'y a aucun besoin de les unifier en une nouvelle plateforme ? D'un autre côté, on pourrait se demander s'il y existe une certaine transformation satisfaisante de l'un vers l'autre.

Dans ce document dont une partie a été publiée dans un atelier international [6] (voir le détail à l'annexe B), nous nous attelons à répondre à ces questions à travers des études de cas. Notre but est de faire ressortir les caractéristiques propres des systèmes modélisés par les processus de Markov étiquetés et les automates hybrides probabilistes, et de montrer que, malgré leurs natures continues, ces deux modèles ne sont pas assez généraux pour modéliser certains types de systèmes continus. L'approche que nous utilisons est la suivante : nous montrerons comment modéliser avec les deux plateformes deux processus représentant le déplacement d'un avion et un bateau en mouvement sur une grille. Ces processus, même s'ils ont été simplifiés, sont suffisamment intéressants pour faire ressortir les différences et les limites des deux plateformes considérées. Ces observations nous amèneront à définir une généralisation des processus de Markov étiquetés et des processus probabilistes hybrides, que nous appelons *LMP hybrides*. Par conséquent, ce nouveau modèle combinera les caractéristiques distinctives des deux classes de systèmes continus, c'est-à-dire, le temps continu et les distributions continues.

Notre travail comprend deux grandes parties. La première est consacrée à la définition des processus de Markov étiquetés et des systèmes hybrides probabilistes. Quant à la seconde partie, elle présente notre comparaison des deux classes de systèmes ainsi que les résultats obtenus.

Le chapitre 2 fait une description générale des systèmes probabilistes et présente deux systèmes simplifiés de processus de Markov étiquétés et de systèmes hybrides probabilistes. Le chapitre 3 est dédié à la présentation de la théorie des processus de Markov étiquetés. Le chapitre 4 est consacré à la définition des automates hybrides probabilistes. Il présente également le résumé d'un de nos travaux sur la vérification des automates hybrides probabilistes qui a été publié dans un atelier international [7] (voir le détail à l'annexe C).

Le chapitre 5 présente la comparaison entre les processus de Markov étiquetés et les automates hybrides probabilistes à travers deux études de cas. Les études de cas

exposent deux systèmes continus, et chacun d'eux a été modélisé comme un processus de Markov étiqueté et comme un automate hybride probabiliste. Ainsi, les forces et les faiblesses des modèles ont pu être décelées.

Enfin, le chapitre 6 introduit le nouveau modèle que nous avons proposé et qui combine les traits propres aux processus de Markov et aux automates hybrides probabilistes. Il présente également les modélisations des systèmes utilisés lors des études de cas en utilisant le nouveau modèle. Enfin, ce chapitre propose des pistes pour la vérification formelle du nouveau modèle.

Première partie

Préliminaires

Chapitre 2

Les systèmes probabilistes

Au cours des années, plusieurs techniques de spécification et de modélisation de différents types de systèmes ont vu le jour. Certaines de ces techniques traitent des systèmes réels pour lesquels il n'est pas toujours évident de déterminer avec certitude le prochain état après l'exécution d'une action donnée : les systèmes probabilistes.

Les systèmes probabilistes se retrouvent un peu partout : de la machine à café au système de pilotage d'un avion. Dans le cas d'une machine à café, on pourrait considérer l'incertitude de la quantité de café servi, tandis que dans le cas du système de pilotage d'un avion, l'incertitude pourrait être attachée à l'altitude perdue après une rotation effectuée par le pilote. Étant donné qu'une incertitude cache généralement des risques, il s'avère donc nécessaire qu'une vérification formelle des systèmes probabilistes soit faite. Par ailleurs, il faut également remarquer que, dans certains cas, lorsqu'une information est fournie sans être accompagnée d'une certaine probabilité de réalisation, elle paraît incomplète. C'est le cas lorsque l'on indique qu'*une machine distributrice pourrait ne pas répondre à une demande sans retourner la pièce à l'utilisateur*. On pourrait donc se demander à quelle fréquence cet incident pourrait se produire, pour ainsi évaluer si on peut prendre le risque d'utiliser la machine. Une reformulation de cette information pourrait être : *il y a 1% de chance que la machine ne réponde pas à une demande sans retourner la pièce*. Dans ce dernier cas, on peut, avec une certaine confiance, prendre le risque d'utiliser la machine.

Nous nous intéressons à déterminer si un système aura un certain comportement avec une certaine probabilité. Par exemple, nous pouvons chercher à savoir si la probabilité qu'une machine distributrice serve du café après qu'un utilisateur en ait fait la demande est supérieure à 0.99. Pour pouvoir faire cette vérification, on se sert d'une *représentation formelle* encore appelée *modèle formel* du système, sur laquelle on applique certaines

méthodes ou techniques bien établies.

On distingue plusieurs types de systèmes probabilistes selon que leur espace d'états est discret ou continu et selon qu'ils fonctionnent en temps discret ou continu. Quelque soit le type auquel il appartient, un système probabiliste a des états, donnant ainsi l'information du système à tout moment. De plus, il effectue des *transitions*, c'est-à-dire des changements d'états, chacune d'elles associée ou non à une *action* et une valeur de probabilité. Dans ce travail, nous considérons les systèmes qui réagissent aux stimuli de l'environnement externe, et par conséquent, nous associons à chaque transition, une action donnée. Nous introduisons également le *non-déterminisme* dans le choix des transitions, dans le sens où nous permettrons que plusieurs transitions étiquetées par la même action partent d'un état.

Ce chapitre présente une introduction sommaire des systèmes probabilistes. Mais avant, nous présentons quelques notions indispensables à la définition de ces systèmes.

2.1 Quelques définitions et notations

Dans cette section, nous donnons des définitions de notions et notations qui nous serons utiles dans le cadre de ce travail. Nous commencerons par la définition d'un système de transition.

Définition 2.1.1 ([8])**.** *Un système de transition (ou système concurrent) est un tuple* $TS = (S, \mathrm{Act}, \rightarrow, I, \mathrm{AP}, L)$ *où*

- S *est un ensemble d'états,*
- Act *est un ensemble fini d'actions,*
- $\rightarrow \subseteq S \times \mathrm{Act} \times S$ *est une relation appelée relation de transition,*
- $I \subseteq S$ *est un ensemble d'états initiaux,*
- AP *est un ensemble de propositions atomiques, et*
- $L : S \rightarrow 2^{\mathrm{AP}}$ *est une fonction de propositions atomiques qui associe à tout état, un ensemble fini de propositions atomiques.*

Les systèmes de transitions constituent une référence en informatique parce qu'ils sont souvent utilisés comme modèles pour décrire le comportement de systèmes. Ils sont généralement représentés par des graphes orientés où les noeuds symbolisent les états, et les flèches, les transitions.

Exemple 2.1.2. *Comme exemple de système de transition, considérons une lampe*

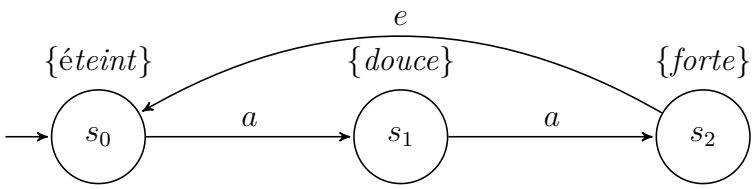

FIGURE 2.1 – Un système de transition : une lampe électrique

éléctrique qui peut éclairer d'une lumière douce ou d'une lumière forte dépendamment de la demande faite. La figure 2.1 présente le système de transition d'une telle lampe.

Le modèle de cette lampe peut être représenté par le tuple $TS := (S, \text{Act}, \rightarrow, \{i\}, \text{AP}, L)$ *où :*

- $S := \{s_0, s_2, s_3\}$;
- $\text{Act} := \{a, e\}$;
- $\text{AP} := \{éteint, douce, forte\}$;
- L *est telle que* $L(s_0) = \{éteint\}$, $L(s_1) = \{douce\}$ *et* $L(s_2) = \{forte\}$;
- *la relation* $\rightarrow \subseteq S \times \text{Act} \times S$ *est définie telle que* $\rightarrow := \{(s_0, a, s_1), (s_1, a, s_2), (s_2, e, s_0)\}$.

Dans le souci de simplification, nous écrivons $s_0 \xrightarrow{a} s_1$ au lieu de $(s_0, a, s_1) \in \rightarrow$. Si s et s' sont des états, et a est une action de TS, alors nous écrivons $s \xrightarrow{a} s'$ pour signifier que si s est l'état courant alors l'action a est exécutée permettant au système de se rendre de s dans l'état s'.

Lorsqu'on accompagne l'exécution des actions d'un système de transition d'une incertitude, on obtient un système de transition probabiliste. Comme dans le cas non probabiliste, les systèmes de transitions probabilistes sont des modèles de référence pour les systèmes probabilistes, étant des modèles de systèmes probabilistes les plus simplifiés. Pour modéliser formellement les comportements des systèmes probabilistes, nous avons besoin de quelques outils mathématiques que nous définissons dans les lignes suivantes.

Définition 2.1.3 ([11]). *Soit* Ω *un ensemble. Une* σ*-algèbre* $\Sigma \subseteq 2^\Omega$ *sur* Ω *est telle que :*

1. $\Omega \in \Sigma$;
2. $\forall (A_i)_{i \in \mathbb{N}} \in \Sigma, \bigcup_{i \in \mathbb{N}} A_i \in \Sigma$;
3. $\forall A \in \Sigma, \overline{A} := \Omega - A \in \Sigma$.

Les éléments d'une σ-algèbre sont appelés des *ensembles mesurables*, et (Ω, Σ) est appelé *espace mesurable*. Dans ce travail, nous nous limitons à l'espace des nombres réels avec la σ-algèbre engendrée par les intervalles.

Définition 2.1.4 ([11])**.** *Soit* (Ω, Σ) *un espace mesurable. Une fonction* $\mu : \Sigma \to [0,1]$ *est une* mesure de probabilité *(ou distribution) sur* (Ω, Σ) *si :*

1. $\mu(\emptyset) = 0$ *;*
2. $\mu(\cup A_i) = \sum_i \mu(A_i)$ *où* $\cup_i A_i$ *est une union dénombrable d'ensembles disjoints ;*
3. $\mu(\Omega) = 1$.

(Ω, Σ, μ) *est un* espace de mesure de probabilité.

La définition suivante est celle d'une fonction de transition *partielle* sur un espace mesurable. Dans ce travail, nous utiliserons cette fonction pour calculer la probabilité d'une exécution dans un système probabiliste. La notation $\mu(x, \cdot)$ représente la fonction μ à laquelle le paramètre x est fixé, mais où le second paramètre demeure libre.

Définition 2.1.5 ([11])**.** *Une* fonction de transition probabiliste *sur un espace mesurable* (Ω, Σ) *est une fonction* $\mu : \Omega \times \Sigma \to [0,1]$ *telle que pour chaque* $x \in \Omega$*, la fonction* $\mu(x, \cdot)$ *est une mesure de probabilité et pour chaque* $E \in \Sigma$*,* $\mu^{-1}(\cdot, E)(A) \in \Sigma$*, pour* $A \in \mathcal{B}$ [1].

$\mu(x, E)$ représente la probabilité que le système prenne une transition à partir de l'état x vers un état dans E. Si a est une action, alors $\mu_a(x, \Sigma)$ représente la probabilité que le système exécute une action a.

La définition suivante est celle d'une distribution de probabilité discrète.

Définition 2.1.6. *Soit* μ *une distribution sur* (Ω, Σ)*.* μ *est une distribution discrète si :*

1. $\Sigma = \mathcal{P}(\Omega)$ *où* $\mathcal{P}(\Omega)$ *est l'ensemble des parties de* Ω *;*
2. *son support est fini ou dénombrable, c'est-à-dire, s'il existe un ensemble dénombrable* $E = \text{supp}(\mu) := \{s \in \Omega : \mu(\{s\}) > 0\} \in \Sigma$ *tel que* $\mu(E) = \mu(\Omega)$*.*

On écrira $\text{Dist}(\Omega)$ pour désigner l'ensemble des distributions discrètes sur Ω.

À part les distributions discrètes, il existe des *distributions continues*. Ainsi, on appelera *distribution continue*, toute distribution qui n'est pas discrète.

Notre contexte étant implanté, nous introduisons, dans les prochaines sections, deux grandes catégories de systèmes probabilistes étudiés dans la littérature : les systèmes probabilistes à temps discret et les systèmes probabilistes à temps continu.

1. C'est la plus petite σ-algèbre qui contient les intervalles.

2.2 Les systèmes à temps discret

Dans cette section, nous présentons les systèmes probabilistes à temps discret. Ce sont généralement des extensions probabilistes des systèmes de transitions. Parmi eux, on distingue les chaînes de Markov (de l'anglais *Markov Chains*) qui ont fait l'objet de nombreux travaux [33, 4]. D'une manière générale, une chaîne de Markov est un système probabiliste à espace d'états fini qui respecte la propriété de Markov. Celle-ci stipule que la prédiction du futur, sachant le présent, n'est pas rendue plus précise par des éléments d'informations supplémentaires concernant le passé. On peut distinguer deux sortes de chaînes de Markov [4, 33] : les *chaînes de Markov à temps discret* (DTMC, de l'anglais *Discrete Time Markov Chains*) et les *chaînes de Markov à temps continu* (CTMC, de l'anglais Continuous Time Markov Chains).

Les chaînes de Markov à temps discret sont des systèmes probabilistes qui évoluent suivant un temps discret. Ils peuvent se substituer aux systèmes de transitions probabilistes ayant des espaces d'états finis. Ces modèles, introduits par Lehmann et Shelahen [35], sont les plus simples représentants des systèmes probabilistes. Plus intuitivement, si **S** est un système appartenant à la classe des DTMC, alors **S** est un ensemble d'états étiquetés par des propositions atomiques assorties d'une fonction d'évolution qui décrit comment le changement d'état a lieu. De plus, **S** respecte la propriété de Markov. Autrement dit, si $P(s, s')$ est la probabilité que le processus soit dans l'état s' à l'instant $t+1$ étant donné qu'à l'instant t, il se trouve à l'état s, alors P est une distribution discrète de probabilité et $P(s, s')$ est indépendante de t et du parcours du processus avant s. Un tel système est une chaîne de Markov à temps discret.

Plus formellement, nous donnons la définition suivante sur la base de celles de Katoen [4] et Kwiatkowska et al. [33] :

Définition 2.2.1 ([33, 4])**.** *Une chaîne de Markov à temps discret $\mathcal{D}$ est un tuple $(S, i, \mathrm{Act}, \mathrm{AP}, P, L)$ tel que :*

- *S est un ensemble fini d'états ;*
- *$i \in S$ est l'état initial ;*
- *Act est un ensemble fini d'actions ;*
- *AP est un ensemble fini de propositions atomiques ;*
- *$P : S \times \mathrm{Act} \times S \to [0, 1]$ est telle que $P(s, a, \cdot)$ est une distribution discrète pour tout $s \in S$ et $a \in \mathrm{Act}$; on écrit $P_a(s, s')$ plutôt que $P(s, a, s')$.*
- *$L : S \to 2^{\mathrm{AP}}$ est la fonction qui assigne à chaque état un sous-ensemble fini de AP.*

Soient $s \in S$, $a \in \mathrm{Act}$ et $P_a(s, \cdot) \in \mathrm{Dist}(S)$. On écrit $s \xrightarrow{a} s'$ si et seulement s'il existe

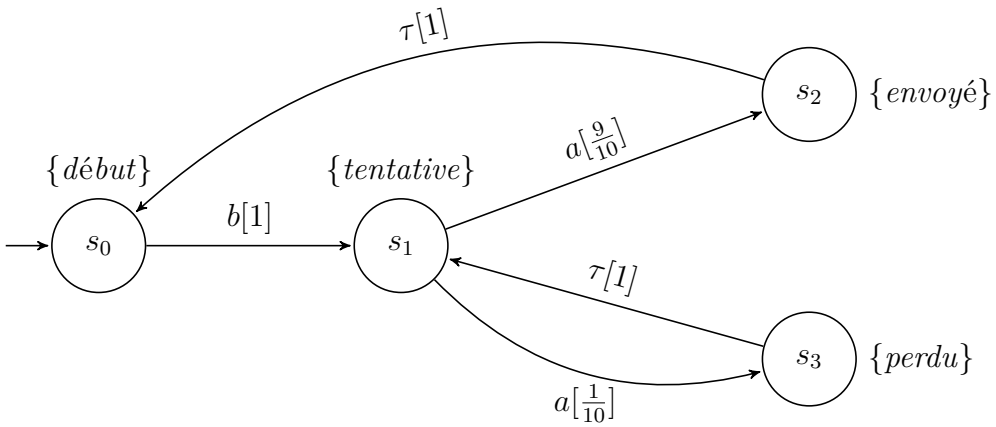

FIGURE 2.2 – Un DTMC à quatre états : un processus de communication via un canal

$s' \in S$ tel que $P_a(s, s') > 0$.

L'exemple 2.2.2 permet de mieux comprendre le fonctionnement d'un DTMC.

Exemple 2.2.2. *Comme exemple de chaîne de Markov à temps discret, considérons un simple protocole de communication fonctionnant par le biais d'un canal. On suppose que l'envoi des messages n'est pas très fiable si bien qu'il peut arriver que des messages se perdent à l'envoi. La figure 2.2 est une représentation du protocole. Supposons qu'en certains temps* $0, 1, 2, ..., T$ *avec* $T \in \mathbb{R}_{\geq 0}$*, nous observons le système. Au début, un message est généré et le processus tente de l'envoyer via le canal. La probabilité que le message soit perdu est* $\frac{1}{10}$*, et dans ce cas, le message est renvoyé de nouveau jusqu'à ce qu'il soit éventuellement arrivé à destination. Dès que le message a été correctement expédié, le système retourne à son état initial. Ce processus de communication est une chaîne de Markov à temps discret. En effet, il peut être modélisé par le tuple* $\mathcal{D} = (S, i, \text{Act}, \text{AP}, P, L)$ *où :*

- $S = \{s_0, s_1, s_2, s_3\}$;
- $i = s_0$;
- $\text{Act} = \{\tau, a, b\}$ *où* τ *est une action interne au processus, a l'action par laquelle un message est envoyé et b l'action qui permet de générer le message ;*
- $\text{AP} = \{$*début, tentative, perdu, envoyé*$\}$;
- P *est tel que* $P_b(s_0, s_1) = 1$, $P_a(s_1, s_2) = \frac{9}{10}$, $P_a(s_1, s_3) = \frac{1}{10}$, $P_\tau(s_3, s_1) = 1$ *et* $P_\tau(s_2, s_1) = 1$;
- $L(s_0) = \{$*début*$\}$, $L(s_1) = \{$*tentative*$\}$, $L(s_2) = \{$*envoyé*$\}$ *et* $L(s_3) = \{$*perdu*$\}$.

Nous pouvons remarquer que la propriété de Markov est respectée sur $\mathcal{D}$ *: le fait de pouvoir aller à l'état* s_3 *ou à l'état* s_2 *à partir de l'état* s_1 *ne dépend pas du parcours effectué précédemment.*

Les chaînes de Markov à temps discret ne permettent pas de représenter strictement le comportement de tous les systèmes. En effet, la plupart des systèmes du monde réel fonctionnent en continu, si bien qu'une modélisation discrète ne permet pas d'être fidèle aux spécifications de ces systèmes. Ainsi, la modélisation faite dans l'exemple 2.2 ne traduit pas strictement le fonctionnement d'un protocole de communication. Entre autres, cette modélisation sous-entend que l'envoi d'un message se fait une unité de temps après que le processus soit rendu à l'état s_1. Dans la réalité, l'envoi du message se fait avec un certain délai. Dans certains cas comme celui du processus de communication, ce facteur n'est pas pertinenent, mais dans d'autres où la variation dans le temps des paramètres physiques est considérée, il est crucial. C'est le cas des systèmes critiques (les avions, les machines utilisées dans les hopitaux, etc.) dont le fonctionnement est strictement lié à l'évolution du temps.

La section suivante est consacrée à la présentation des systèmes fonctionnant en continu.

2.3 Les systèmes à temps continu

Contrairement aux systèmes à temps discret où les transitions probabilistes sont observées à toutes les unités de temps, les systèmes à temps continu ont un fonctionnement continu [30, 4]. Dès lors, une variable réelle t, représentant le temps, est prise en compte dans le fonctionnement du système. Il existe plusieurs modèles de représentation formelle pour les différents types de systèmes à temps continu. Ainsi, on distingue des systèmes qui fonctionnent uniquement à temps continu, et d'autres qui combinent le temps continu et le temps discret. Parmi les systèmes dont le fonctionnement est purement en continu, il y a les chaînes de Markov à temps continu (CTMC pour Continuous Time Markov Chains). Les CTMC sont des processus probabilistes à espace d'état fini et pouvant effectuer des transitions d'état à n'importe quel moment. L'incertitude réside dans l'évolution du temps et, comme les DTMC, ces systèmes respectent la loi de Markov.

Nous nous intéressons plus particulièrement aux systèmes probabilistes qui combinent le temps continu et le temps discret : les systèmes hybrides probabilistes. Comme introduction, nous présentons une classe de systèmes à temps réel qui sont moins généraux que les systèmes hybrides probabilistes, mais qui se comportent comme eux : les automates temporisés probabilistes [31, 41, 34]. Dans ce qui suit, nous présentons brièvement les automates temporisés probabilistes. Notons que cette partie n'est pas essentielle pour comprendre le reste de notre travail. Par conséquent, le lecteur peut sauter au chapitre suivant.

Comme les systèmes hybrides, les automates temporisés font interagir deux composantes lors de leur fonctionnement : une composante continue et une composante discrète. La composante continue assure l'évolution de variables qui sont des horloges représentant le temps. Par conséquent, si x est une variable dans un automate temporisé, son évolution ou sa vitesse peut être représentée par l'équation différentielle de degré 1 $\dot{x} = 1$, où $\dot{x}$ est la première dérivée de x par rapport au temps[2]. La composante discrète, quant à elle, régularise le système et intervient pour faire passer d'un état à un autre par le biais de transitions discrètes. Lors de ces transitions, les variables peuvent être *réinitialisées* à zéro. Pour cela, on dit que la composante discrète joue le rôle de *contrôleur* et on la représente comme un système de transition fini probabiliste dans lequel chaque état, encore appelé *location* ou *mode*, représente l'ensemble des «sous-états» d'une situation de la composante continue et un arc, une transition probabiliste. Contrairement aux systèmes dont le fonctionnement est purement continu, le comportement incertain a lieu lors de l'exécution des transitions discrètes dans les automates temporisés probabilistes. Chaque transition discrète est donc étiquetée par une probabilité qui représente la probabilité que l'action associée soit exécutée. Pour permettre une interaction entre les composantes continue et discrète, des contraintes s'appliquant aux transitions discrètes sont définies. Dans les automates temporisés, ces contraintes, encore appelées préconditions, étiquètent les transitions discrètes et sont généralement des conjonctions (ou disjonctions) de clauses de la forme $x \sim m$ où x est une horloge qui représente le temps, m est un entier naturel et $\sim$ est l'une des relations de l'ensemble $\{<, \leq, =, >, \geq\}$. Dès lors, une transition discrète est dite *active* lorsque la précondition qui l'étiquette est respectée, et c'est seulement à cette condition qu'elle peut être exécutée.

Exemple 2.3.1. *Considérons l'exemple probabiliste d'un interrupteur de lampe électrique inspiré d'un exemple de [8]. L'allumage ou l'extinction de la lampe se fait en appuyant sur l'interrupteur. Lorsqu'elle est éteinte, la lampe peut être allumée à tout moment. L'utilisateur peut éteindre la lampe au moins une unité de temps après le dernier allumage de la lampe. Après deux unités de temps, la lampe s'éteint automatiquement. L'horloge x est utilisée pour garder une trace du temps écoulé depuis le dernier allumage. Le diagramme des modes de la figure 2.3 indique un comportement possible que l'on pourrait avoir.*

On distingue deux modes dans ce système : on*, qui caractérise un état dans lequel la lampe est allumée, et* off *lorsque celle-ci est éteinte. Les lignes continues sur les axes* on *et* off *correspondent aux périodes durant lesquelles la lampe est allumée ou éteinte respectivement. Les lignes discontinues, quant à elle, correspondent aux moments où les transitions discrètes ont lieu. Modifions à présent le système en introduisant*

2. Dans la littérature, il arrive que l'on utilise aussi la notation $\frac{dx}{dt}$ pour désigner la première dérivée de x par rapport au temps.

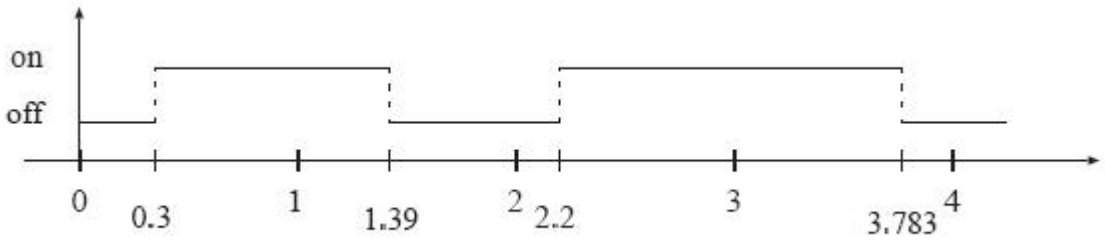

FIGURE 2.3 – Un comportement possible de la lampe électrique

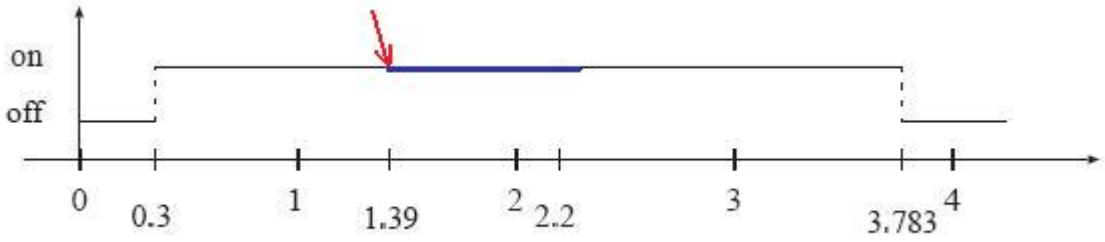

FIGURE 2.4 – Un comportement possible de la lampe électrique probabiliste

une incertiude lors de l'extinction de la lampe, dans le sens où il y a 1% *de chance que la lampe ne s'éteigne pas lorsque l'extinction est possible. La figure 2.4 illustre un comportement de la lampe. Par la ligne bleue, on représente le cas où la lampe ne s'est pas éteinte.*

Les automates temporisés, dans leur fonctionnement, assurent l'évolution d'un nombre fini horloges et à tout instant t, les valeurs de ces horloges sont connues. Ainsi, si on désigne par $X = \{x_1, \ldots, x_n\}$ pour $n \geq 1$ l'ensemble des horloges d'un automate temporisé, alors on définit la notion de *valuation* de X comme suit.

Définition 2.3.2. *Si* $X = \{x_0, x_1, ..., x_n\}$ *est un ensemble de variables et que* $\mathsf{a} = (\mathsf{a}_0, \mathsf{a}_1, ..., \mathsf{a}_n) \in \mathbb{R}^n$ *est tel que chaque valeur* a_i *correspond à la valeur de la variable* x_i*, alors* a *est une valuation de* X*.*

Présentons la définition formelle d'un automate temporisé probabiliste. Pour cela, introduisons la notion de *rectangle* sur un ensemble fini.

Définition 2.3.3 ([6])**.** *Soit* X *un ensemble fini de variables* x_i*. Un ensemble* U *de valuations de* X *est rectangulaire s'il existe une famille d'intervalles* $(I_x)_{x \in X}$ *avec des*

bornes rationnelles telles que $U = \{\mathsf{a} \in \mathbb{R}^n \mid \mathsf{a}(x) \in I_x \text{ pour tout } x \in X\}$. *Notons* $R(X)$ *l'ensemble des rectangles sur* X.

Dans la suite, nous notons $\mathsf{a}(x_i)$ la valeur correspondante à la variable x_i dans a et si γ est un rectangle sur X, alors nous écrivons γ_i pour désigner le rectangle associé à la variable x_i dans γ (le rectangle se trouvant à la i-ème dimension de γ). Comme nous le verrons un peu plus loin, une valuation est l'une des caractéristiques d'un état dans un automate temporisé.

Nous introduisons la notation $*$ qui servira un peu plus loin à indiquer la constance au niveau de la valeur d'une variable après une transition. Ainsi, si a et a' sont deux valuations sur X telles que le système quitte un état avec la valuation a pour un autre avec la valuation a', alors nous substituerons la valeur de la variable x_i dans a' par $*$ pour indiquer que celle-ci est la même avant et après la transition, autrement dit $\mathsf{a}(x_i) = \mathsf{a}'(x_i)$. Par exemple, supposons que $X := \{x_1, x_2\}$ et qu'à l'instant t le système se trouve dans un état avec la valuation $(1, 2)^T$, et qu'à l'instant $t + 1$, il se trouve dans un état avec la valuation $(2, 2)^T$. Nous écrirons donc $(2, *)^T$ pour indiquer que la valeur de x_2 n'a pas changé après la transition.

Définition 2.3.4 ([41]). *Un automate temporisé probabiliste est un tuple* $T = (X, V, L, \text{Act}, \text{Init}, \text{inv}, prob, \langle pre_v \rangle_{v \in V})$ *tel que :*

- $X = \{x_1, \ldots, x_n\}$ *est l'ensemble fini d'horloges pour* $n \geq 1$ *;*
- $V = \{v_1, \ldots, v_m\}$ *est l'ensemble fini des modes ;*
- $L : V \to 2^{\text{AP}}$ *est la fonction des propositions atomiques ;*
- Act *est l'ensemble des actions ;*
- $\text{Init} : V \to \mathbb{N}^n \cup \{\emptyset\}$ *associe à tout mode un ensemble de valeurs initiales dans* $R(X)$ *;*
- $\text{inv} : V \to R(X)$ *est la fonction qui associe à tout mode l'ensemble des invariants associé qui est un rectangle sur* X *;*
- $prob : V \times \text{Act} \to \mathcal{P}(\text{Dist}(V \times R(X \cup \{*\}) \times \mathcal{P}(X)))$ *est la fonction qui associe tout mode* v *et toute action* $a \in \text{Act}$ *à un ensemble fini de distributions de probabilité sur* $V \times R(X) \times \mathcal{P}(X)$. *Par conséquent, pour* $v \in V$ *et* $a \in \text{Act}$ *; on associe un ensemble de distributions* $prob(v, a) = \{\mu_v^1, \ldots, \mu_v^m\}$ *pour* $m \geq 1$.
- *pour tout mode* v, *on définit* $pre_{v,a} : prob(v, a) \to R(X)$ *comme la fonction qui associe à toute distribution* μ, *definie pour un mode* v *et une action* a, *l'ensemble des préconditions, qui est un rectangle sur* X.

Soient T un automate temporisé probabiliste et a une action. D'une manière générale, on suppose que le point initial est unique ; c'est-à-dire qu'il existe un mode $v_0 \in V$ tel que la condition initiale $\text{Init}(v_0)$ est un singleton dans $\mathbb{R}^n$, et $\text{Init}(v) = \emptyset$ pour tout autre

$v \in V \setminus \{v_0\}$. Par conséquent, le contrôle du modèle commence dans le mode v_0 avec la valuation de la variable donnée par $\text{Init}(v_0)$. Lorsque l'automate temporisé se retrouve dans un mode donné $v \in V$, les valeurs des variables réelles dans X évoluent de façon continue au rythme d'une horloge puisqu'en ce moment c'est la composante continue qui a le contrôle. Le système demeure dans le mode v tant que la condition $\text{inv}(v)$ est respectée par la valuation sur X, c'est-à-dire que si $\mathsf{a} \in \mathbb{R}^n$ désigne la valuation courante alors le système reste dans le mode v si et seulement si $\mathsf{a} \in \text{inv}(v)$. Une transition discrète de v peut avoir lieu si et seulement s'il existe une distribution $\mu \in \mathit{prob}(v, a)$, choisie de façon non-déterministe, telle que la condition définie par $\mathit{pre}_{v,a}(\mu)$ est satisfaite par la valuation courante, c'est-à-dire si $\mathsf{a} \in \mathit{pre}_v(\mu)$. Dans un tel cas, on dit que μ is *active*.

En supposant qu'il existe une transition discrète pour une distribution de probabilité active $\mu \in \mathit{prob}(v)$, alors un choix probabiliste vers un mode cible w, et une réinitialisation de certaines horloges, sont faits. Plus précisement, avec la probabilité $\mu(w, \mathsf{post}, \mathcal{X})$, une transition est exécutée vers le mode $w \in V$ avec une valuation b dans l'ensemble $\mathsf{post} \subseteq R(X \cup \{*\})$ telle que $\mathsf{b}(x_i) = \mathsf{a}(x_i)$ pour toute variable $x_i \in X \setminus \mathcal{X}$. L'ensemble post représente les postconditions, tandis que $\mathcal{X}$ est l'ensemble des variables réinitialisées. Nous utilisons, ici, la notation $*$ dans les transitions. Ainsi, dans une transition a de probabilité $\mu(w, \mathsf{post}, \mathcal{X})$, nous écrirons $\mathsf{post}_i = \{*\}$ pour signifier qu'aucune réinitialisation n'a été faite sur la variable $x_i \in X$, autrement dit, sa valeur reste la même qu'avant la transition. Ensemble, les postconditions et l'ensemble des variables réinitialisées déterminent les effets sur les variables continues après les transitions. Ces ensembles sont sujets à la règle suivante : pour tout $v \in V$, tout $\mu \in \mathit{prob}(v)$, et tout $(w, \mathsf{post}, \mathcal{X}) \in \text{supp}(\mu)$, nous avons $\mathsf{post} \subseteq \text{inv}(w)$.

Dans un automate temporisé, l'état du système, à un moment donné, est défini en fonction du mode et de la valuation courante. Plus formellement, on le définit comme suit.

Définition 2.3.5. *Un état est un couple* $(v, \mathsf{a}) \in V \times \mathbb{R}^n$. *Si* $\mathsf{a} \in \text{inv}(v)$, *alors on dit que l'état* (v, a) *est* admissible *en* v.

Illustrons maintenant ces notions avec l'exemple 2.3.6. Cet exemple, même s'il est assez simple, permet de mettre en évidence les principales caractéristiques d'un automate temporisé probabiliste.

Exemple 2.3.6. *La figure 2.5 montre une représentation graphique de l'automate temporisé probabiliste de la lampe électrique de l'exemple 2.3.1. Le modèle formel de cet automate temporisé est* $T = (X, V, L, \text{Act}, \text{Init}, \text{inv}, \mathit{prob}, \mathit{pre}_{v,a})$ *tel que :*

- $X := \{x\}$;
- $V := \{\text{on}, \text{off}\}$;

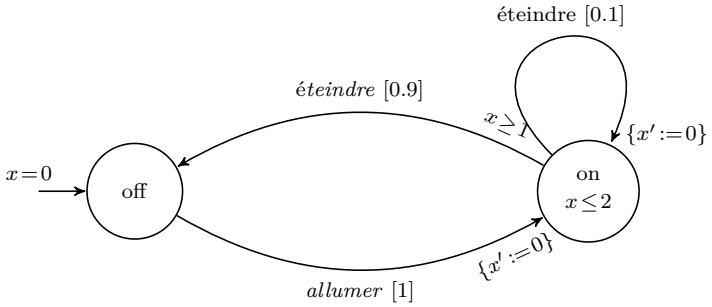

FIGURE 2.5 – L'automate temporisé probabiliste de la lampe électrique

- *L est telle que* $L(\text{on}) := \{allumé\}$ *et* $L(\text{off}) := \{éteint\}$ *;*
- $\text{Act} := \{éteindre, allumer\}$ *où éteindre et allumer représentent respectivement le fait d'éteindre et d'allumer la lampe ;*
- *à l'état initial, la lampe est éteinte, donc* $\text{Init}(\text{on}) := \{0\}$ *et* $\text{Init}(\text{off}) := \emptyset$ *;*
- inv *est la fonction telle que* $\text{inv}(\text{on}) := [0, 2]$ *et* $\text{inv}(\text{off}) := \mathbb{R}_+$ *;*
- *prob définit les ensembles de distributions suivants pour chacun des modes comme suit :* $prob_{éteindre}(\text{on}) := \{\mu_1\}$ *et* $prob_{allumer}(\text{off}) := \{\mu_2\}$ *avec* $pre_{\text{on}}(\mu_1) := [1, \infty)$ *et* $pre_{\text{off}}(\mu_2) := \mathbb{R}_+$ *tels que :*
  - $\mu_1(\text{off}, \{*\}, \{x\}) := 0.9$ *et* $\mu_1(\text{on}, \{0\}, \{x\}) := 0.1$,
  - $\mu_2(\text{on}, \{0\}, \{x\}) := 1$.

La vérification des automates temporisés probabilistes a été étudiée par plusieurs groupes de recherche notamment par Sproston et al. [34, 41], Beauquier et al. [10] et Jurdziński et al. [31]. Les propriétés vérifiées ont été exprimées dans plusieurs logiques temporelles probabilistes, telles que PBTL [41], qui permettent de formuler des propriétés de la forme «une seconde après que la lampe soit éteinte, elle peut s'allumer dans 10% des cas». Ces formules sont ensuite vérifiées à l'aide d'outils, tels que PRISM [34].

Dans ce chapitre, nous avons introduit les systèmes probabilistes. Parmi ces systèmes, certains sont caractérisés par un espace d'états discret tandis que d'autres par un espace d'états continu. Parmi eux, on peut distinguer les systèmes qui fonctionnent d'une manière discrète dans le temps, et les systèmes qui fonctionnent en continu. Le présent travail a pour but de faire une étude comparative entre deux types de systèmes probabilistes à espace d'états continu : les processus de Markov étiquetés (de l'anglais Labelled Markov Processes) de Desharnais [14] et les systèmes hybrides probabilistes de Sproston [41]. Les processus de Markov étiquetés et les systèmes hybrides probabilistes, comme nous l'avons mentionné plus haut, héritent respectivement des systèmes probabilistes à temps discret et des systèmes probabilistes à temps réel, classes de systèmes ayant été présentées dans ce chapitre. Notre comparaison portera essentiellement sur la base

des langages générés, des techniques utilisées pour la vérification, et des fondements des théories qui sous-tendent ces deux types de systèmes probabilistes à espace d'états continu. Les chapitres suivants seront consacrés à la définition des processus de Markov étiquetés et des systèmes hybrides probabilistes.

Chapitre 3

Les Processus de Markov Étiquetés (LMP)

Les processus de Markov étiquetés (LMP, provenant de l'anglais *Labelled Markov Processes*) sont utilisés pour représenter des systèmes réactifs et probabilistes pouvant avoir un espace d'états continu, c'est-à-dire des systèmes qui réagissent aléatoirement aux événements de leur environnement et qui possèdent une infinité non-dénombrable d'états. De plus, comme leur nom l'indique, les LMP sont des chaînes de Markov, et plus précisement une extension des DTMC, à la différence que l'espace d'états est continu. Ainsi, dans un LMP, les transitions sont exécutées à toutes les unités t de temps.

Les LMP ont fait l'objet de nombreux travaux dont ceux de Desharnais et al. [12, 14, 16, 17, 15] et de Richard [38]. Il en a résulté que la vérification formelle des LMP, en tant que modèles de systèmes probabilistes à espace d'états continu, peut être faite par des techniques d'approximation [16], par la recherche de modèles semblables (*simulation/bissimulation*) [17, 15] ou par la vérification directe [38]. L'application de la vérification directe a permis de mettre au point un vérificateur formel, CISMO (de l'anglais *ContInuous State space MOdel-checker*) [38], qui, étant donné un modèle et une propriété encodée dans une *logique* bien définie, détermine si le modèle satisfait ou non cette propriété.

Dans cette section, nous présentons les processus de Markov étiquetés du point de vue du langage qu'ils expriment, et de la logique qui les accompagne.

3.1 Définition et exemples

En tant que modèle de représentation formelle de systèmes probabilistes à temps discret, les LMP se comportent en partie comme les DTMC. En particulier, les LMP respectent la propriété de Markov, et par conséquent, pour tout état et toute action, la probabilité qu'une transition soit exécutée dépend seulement de l'état courant et non de l'historique du processus. De plus, lorsque l'espace des états est fini, un LMP se comporte exactement comme un DTMC, et dès lors, une transition peut être spécifiée en donnant, pour chaque action, une probabilité de se rendre d'un état dans un autre. Toutefois, dans le cas où l'espace d'états est continu, on ne pourra plus se permettre de définir des transitions probabilistes d'un état vers un autre parce que, dans de nombreux cas, les valeurs de probabilité de ces transitions sont zéro. Par exemple, considérons le choix uniforme d'un nombre entre 0 et 1 : chaque nombre a la probabilité 0 d'être choisi, mais un intervalle a une probabilité égale à sa longueur. Dès lors, on ne peut plus déterminer la probabilité de tout ensemble, ou de n'importe quel ensemble en additionnant la probabilité de ses éléments parce que dans le cas où les ensembles ne sont pas disjoints, la somme des probabilités ne donne pas la mesure de l'union. Pour cela, on parlera de la probabilité d'un état s vers un ensemble d'états E, autrement dit les fonctions de transitions probabilistes seront continues.

Exemple 3.1.1. *Pour illustrer un système probabiliste possédant un ensemble d'états non-dénombrable, considérons l'exemple de la distributrice de café inspiré de [38]. Il s'agit d'une distributrice portant deux boutons c et C pour servir respectivement une petite et une grande tasse de café. Une fois le café servi, dans* 99% *des cas, le système est prêt à servir à nouveau du café dès qu'une pièce est insérée. Nous restreindrons l'espace d'états à* $\{\text{init}, 0\} \cup [200, 750]$ *: initiallement, la machine est dans l'état* init *et la quantité de café servi est représentée par des valeurs réelles lorsque le client appuie sur un des deux boutons. La petite et la grande tasse de café correspondent aux états des intervalles* $[200, 400]$ *et* $(400, 750]$ *respectivement. On associe une distribution* μ_c *qui suit une loi Normale de moyenne* 250 *et de variance* 25 *au bouton de la petite tasse de café et une autre distribution Normale* μ_C *de moyenne* 550 *et de variance* 49 *au bouton de la grande tasse de café. Le paramètre, la quantité de café, est une variable réelle, et par conséquent l'espace d'états est continu. On pourrait évaluer certaines probabilités comme la probabilité que la quantité de café servie soit comprise dans l'intervalle* $[350, 450]$*, ou la probabilité que le système ne réponde pas après avoir servi une petite tasse de café.*

Plus formellement, un LMP est défini de la façon suivante :

Définition 3.1.2. *Un LMP est un tuple de la forme* $(S, \Sigma, i, \text{Act}, \text{AP}, L, \{\mu_a | a \in \text{Act}\})$ *où*

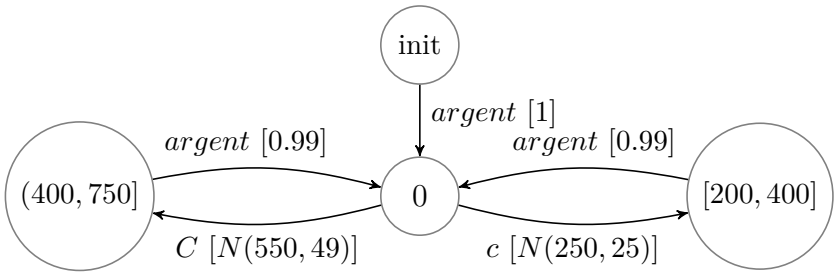

FIGURE 3.1 – Le LMP d'une distributrice de café 3.1.1

- $S \in \Sigma$ *est l'ensemble des états,*
- Σ *est une* σ*-algèbre sur* S*,*
- $i \in S$ *est l'état initial,*
- Act *est l'ensemble des actions,*
- AP *est l'ensemble dénombrable des étiquettes sur les états,*
- $L : S \to 2^{\text{AP}}$ *est la fonction de propositions atomiques,*
- $\forall a \in \text{Act}, \mu_a : S \times \Sigma \to [0, 1]$ *est une fonction de transition probabiliste (Définition 2.1.5).*

Comme mentionné dans la définition 2.1.5, $\mu_a(x, E)$ désigne la probabilité de se rendre dans un état $s' \in E$ à partir de x en faisant l'action a. Une particularité au niveau des LMP est le fait que les fonctions de transitions probabilistes peuvent être *partielles*, c'est-à-dire que leur image n'est pas 0 ou 1, mais $[0, 1]$. Pour cela, des situations du genre $0 < \mu_a(x, S) < 1$ sont permises. Le fait que $\mu_a(x, S)$ soit strictement supérieure à 0 sous-entend que de l'état x, il est possible d'exécuter l'action a et de tomber dans l'ensemble S. Par contre, le fait que la valeur de la probabilité soit strictement inférieure à 1 implique qu'il est possible que le système ne réponde pas suite à l'action demandée, tel qu'illustré dans l'exemple suivant.

Exemple 3.1.3. *Reprenons l'exemple de la distributrice de café de l'exemple 3.1.1. Un modèle LMP de ce système est défini sur* $\text{AP} := \{petit, grand\}$*, comme* $(\{\text{init}, 0\} \cup [200, 750], \Sigma, 0, \{\mu_c, \mu_C, \mu_{argent}\}, L)$*, où* $L(s) = \{petit\}$ *pour tout* $s \in [200, 400]$ *et* $L(s) = \{grand\}$ *pour tout* $s \in]400, 700]$*. La figure 3.1 est une représentation graphique du modèle. La figure 3.2, quant à elle, représente la modélisation du système dans* CISMO*, un outil de vérification des LMP dont nous reparlerons plus loin.*

Les étiquettes c *et* C *encodent respectivement l'action par laquelle une petite ou grande tasse est demandée tandis que l'étiquette argent représente l'action par laquelle une pièce est insérée dans la machine. Chaque transition est étiquétée avec une action suivie, dans des crochets, par la distribution de probabilité sur l'ensemble cible. Comme on ne peut plus déterminer les transitions en donnant les probabilités d'un point à l'autre, nous*

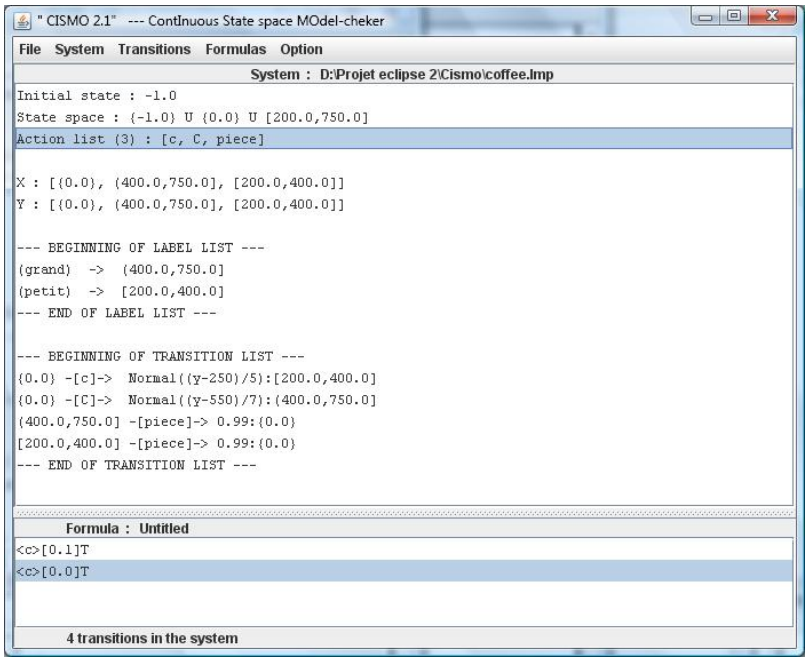

FIGURE 3.2 – Le modèle du café dans CISMO

devons utiliser des notations qui encodent des distributions continues. Ainsi, la transition de l'état 0 *vers* $[200, 400]$*, étiquetée par* $c[N(250, 25)]$ *dans la figure 3.1, traduit une transition* c *de l'état* 0 *dont la distribution de probabilité suit une distribution normale sur l'ensemble* $[200, 400]$*. On peut remarquer que la distribution de probabilité associée aux transitions* τ *à la sortie des états est partielle, puisque la somme des probabilités est inférieure à 1. Ceci peut s'interprêter par le fait que la machine ne réponde pas lorsque l'usager fait sa demande de café. Cela pourrait s'expliquer, non pas par le fait que la machine aurait changé d'état (panne, plus de café, etc.), mais simplement par le fait que la machine n'a pas réagi au stimuli de l'environnement externe.*

Calculons, à présent, quelques valeurs de probabilité. Puisque la capacité des grandes tasses de café est inférieure à 700*, la probabilité que le café déborde est égale à celle d'avoir une quantité de café comprise entre* 700 *et* 750 *:*

$$\begin{aligned} \mu_c(0, [700, 750]) &= \Phi(\frac{750 - 250}{25^2}) - \Phi(\frac{700 - 250}{25^2}) \\ &= 1 - 1 \\ &= 0 \end{aligned}$$

$$\begin{aligned}\mu_C(0,[700,750]) &= \Phi(\frac{750-550}{49^2}) - \Phi(\frac{700-550}{49^2}) \\ &= 0.99997764 - 0.99889783 \\ &= 0.00108\end{aligned}$$

Il a donc 0.1% *de chance que le café déborde.*

De manière analogue, on calcule la probabilité que la machine serve une quantité de café comprise entre 350 *et* $450mL$*. Cela revient à calculer la probabilité que la machine serve une petite tasse de* 350 *à* $400mL$*, ou une grande tasse de* 400 *à* $450mL$*. Désignons par* $P(x,[350,450])$ *cette probabilité. On a donc :*

$$P(x,[350,450]) = \mu_c(0,[350,400]) + \mu_C(0,]400,450]).$$

Les calculs de $\mu_c(0,[350,400])$ *et de* $\mu_C(0,]400,450])$ *se font comme suit :*

$$\begin{aligned}\mu_c(0,[350,400]) &= \Phi(\frac{400-250}{25^2})) - (\Phi(\frac{350-250}{25^2}) \\ &= 1 - 0.999968329 \\ &= 3.16703E-05,\end{aligned}$$

$$\begin{aligned}\mu_C(0,]400,450]) &= [(\Phi(\frac{450-550}{49^2}) - \Phi(\frac{400-550}{49^2}))] \\ &= 0.020634544 - 0.001102169 \\ &= 1.9532375E-02.\end{aligned}$$

Il y a donc $0,003\%$ *de chance que la machine serve entre 350 et 450ml de café lorsqu'on demande une petite tasse de café et* 2% *lorsqu'on demande une grande tasse.*

Nous présentons l'exemple suivant dans lequel on modélise un système plus abstrait et plus complexe que celui de la distributrice.

Exemple 3.1.4. *Soit* $\mathbf{S} = ([0,3],\mathcal{B},1,\{a,b\},\{\mu_a,\mu_b\})$ *le système tel que les fonctions de transitions* μ_a *et* μ_b *sont définies de la façon suivante :*

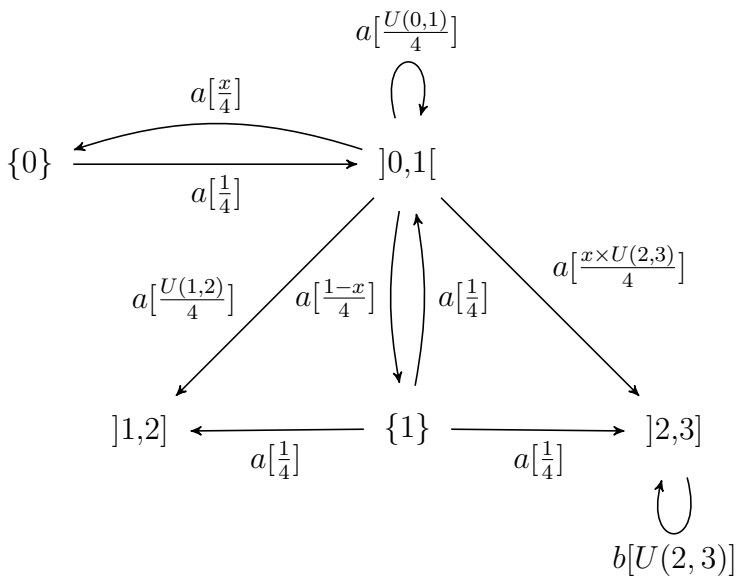

FIGURE 3.3 – Exemple d'un LMP avec des distributions continues

– *si* $x \in [0,1]$,

$$\begin{aligned}
\mu_a(x,\{0\}) &= \frac{x}{4} \\
\mu_a(x,\{1\}) &= \frac{1-x}{4} \\
\mu_a(x,\cdot) &= \frac{U(0,1)}{4} \ \textit{sur}\ ]0,1[\\
\mu_a(x,\cdot) &= \frac{U(1,2)}{4} \ \textit{sur}\ ]1,2] \\
\mu_a(x,\cdot) &= \frac{x \times U(2,3)}{4} \ \textit{sur}\ ]2,3]
\end{aligned}$$

– *si* $x \in]2,3]$,

$$\mu_b(x,\cdot) = U(2,3)$$

– *pour toutes les autres valeurs de* x, $\mu_a(x,\cdot)$ *et* $\mu_b(x,\cdot)$ *valent 0.*

La figure 3.3 représente d'une façon informelle le système. Chacune des transitions y est représentée et est étiquetée par l'action ainsi que la valeur de la probabilité associée.

Procédons, à présent, au calcul de quelques valeurs de probabilité.

Soit $x \in [0,1]$. *Calculons* $\mu_a(x,S)$ *qui représente la probabilité de se rendre à partir de l'état* x *vers un autre état du système. Cette valeur est donc égale à la probabilité de*

se rendre de x dans un état appartenant à l'un des ensembles $]0,1[$, $\{1\}$, et $]1,3]$. Par conséquent, la valeur de $\mu_a(x,S)$ s'obtient de la façon suivante :

$$\begin{aligned}\mu_a(x,S) &= \mu_a(x,\{0\}) + \mu_a(x,]0,1[) + \mu_a(x,\{1\}) + \mu_a(x,]1,2]) + \mu_a(x,]2,3]) \\ &= \frac{x}{4} + \frac{1}{4} + \frac{1-x}{4} + \frac{1}{4} + \frac{x}{4} \\ &= \frac{3+x}{4}.\end{aligned}$$

Il s'agit d'une situation dans laquelle on a $\mu_a(x,S) > 0$ pour tout x appartenant à $[0,1]$. Par conséquent, il est toujours possible de prendre une transition étiquetée a et de se rendre dans un autre état du système. De même, de l'état 1, la valeur de probabilité est 1, ce qui implique que de l'état 1, on est certain qu'une transition associée à l'action a est toujours faite. Enfin, il y a des chances que le système ne prenne aucune transition et reste dans le même état (cas où $\mu_a(x,S) < 1$). La probabilité que cela se produise est le complémentaire de la probabilité qu'une transition étiquetée a soit prise, soit $1 - \mu_a(x,S) = \frac{1-x}{4}$.

Nous pouvons également calculer $\mu_a(x,[1,2])$ qui est la probabilité de se rendre de l'état x dans un état de $[1,2]$ en faisant l'action a, en supposant une fois de plus que x appartient à $[0,1]$. Cette valeur est égale à la somme des probabilités de se rendre dans un état de $\{1\}$ ou dans un état de $]1,2]$:

$$\begin{aligned}\mu_a(x,[1,2]) &= \mu_a(x,\{1\}) + \mu_a(x,]1,2]) \\ &= \frac{1-x}{4} + \frac{1}{4} \\ &= \frac{2-x}{4}.\end{aligned}$$

Contrairement à l'exemple de la distributrice, dans l'exemple précédent, les valeurs de probabilité obtenues dépendent de la valeur du paramètre de l'état actuel, c'est-à-dire x. Cela nous permet de définir la probabilité des transitions de manière générique, pour un nombre non dénombrable de valeurs de départ.

Exemple 3.1.5. *Considérons l'exemple 3.1.4. Nous pouvons chercher à déterminer l'ensemble des états x appartenant à $[0,1]$ à partir desquels on peut exécuter, avec une*

probabilité supérieure à 0.9*, l'action a. Cela revient à résoudre l'inéquation* $\mu_a(x, S) > 0.9$ *pour* $x \in [0, 1]$.

Pour $x \in [0, 1]$,

$$\begin{aligned} \mu_a(x, S) > 0.9 &\Leftrightarrow \frac{3+x}{4} > 0.9 \\ &\Leftrightarrow x > 0.6 \\ &\Leftrightarrow x \in]\frac{6}{10}, 1]. \end{aligned}$$

L'ensemble des états x *appartenant à* $[0, 1]$ *qui satisfont cette propriété sont ceux de l'ensemble* $]\frac{6}{10}, 1]$.

Le modèle et le langage des LMP étant présentés, nous présentons, dans la section suivante, la logique qui permet de spécifier les propriétés à vérifier.

3.2 Vérification formelle

Pour valider les systèmes modélisés par les LMP, on a besoin d'une logique qui servira à décrire les propriétés à vérifier. La logique $\mathcal{L}_0$ est celle utilisée pour la vérification des LMP et est inspirée de celle de Larsen et Skou [19]. La syntaxe de $\mathcal{L}_0$ se définit comme suit :

$$\phi := \top |\ \mathrm{p} |\ \neg\phi |\ \phi_1 \wedge \phi_2 |\ \langle a \rangle_{\mathsf{q}} \phi.$$

où $a \in \mathrm{Act}$ est une action, $\mathrm{p} \in \mathrm{AP}$ une proposition atomique, et q un nombre rationnel compris entre $[0, 1]$.

Soit $\mathbf{S} = (S, \Sigma, i, \mathrm{Act}, \mathrm{AP}, L, \{\mu_a | a \in \mathrm{Act}\})$. Soit $s \in S$ un état. Si ϕ est une propriété de la logique $\mathcal{L}_0$, alors $s \vDash \phi$ signifie que s satisfait ϕ. D'une manière générale, nous écrivons $[\![\phi]\!]$ pour dénoter l'ensemble des états qui satisfont ϕ, c'est-à-dire l'ensemble $\{s \in S \mid s \vDash \phi\}$. Sachant que ϕ est une propriété de $\mathcal{L}_0$, $[\![\phi]\!]$ se définit comme suit :

$$\begin{aligned} [\![\top]\!] &= S \\ [\![\mathrm{p}]\!] &= \{s \in S | p \in L(s)\} \\ [\![\phi_1 \wedge \phi_2]\!] &= [\![\phi_1]\!] \cap [\![\phi_2]\!] \\ [\![\langle a \rangle_{\sim \mathsf{q}} \phi]\!] &= \{s \in S | \mu_a(s, [\![\phi]\!]) \sim \mathsf{q}\} \text{ où } \sim \in \{>, \geq\}. \end{aligned}$$

Pour $\mathbf{S} = (S, \Sigma, i, \text{Act}, \text{AP}, L, \{\mu_a | a \in \text{Act}\})$, la sémantique de $\mathcal{L}_0$ s'interprète de la manière suivante. Si l'état initial i est tel que $i \vDash \phi$, alors on dira que le système $\mathbf{S}$ satisfait ϕ. On écrira, pour un ensemble E, $E \vDash \phi$ pour traduire le fait que $\forall s \in E, s \vDash \phi$. La définition de la relation $\vDash$ se fait par induction. Ainsi, tous les états de S satisfont la formule $\top$. $s \vDash \text{p}$ si et seulement si $s \in L(\text{p})$. $s \vDash \neg\phi$ si et seulement si s ne satisfait pas ϕ. $s \vDash \phi_1 \wedge \phi_2$ si et seulement si $s \vDash \phi_1$ et $s \vDash \phi_2$. De plus, $s \vDash \langle a \rangle_{\sim \mathsf{q}} \phi$ si et seulement s'il existe un ensemble $E \in \Sigma$ tel que $P_a(s, [\![\phi]\!]) \sim \mathsf{q}$. Intuitivement, cela veut dire il est possible de faire une transition a, avec une probabilité en relation $\sim$ avec q, vers un état qui satisfait ϕ. Enfin, pour toute propriété ϕ de $\mathcal{L}_0$, nous avons la relation $[\![\phi]\!] \in \Sigma$, ce qui se démontre par une induction structurelle simple [14].

Soient a et b deux actions, p et q deux nombres rationnels, et soit ϕ une propriété de la logique $\mathcal{L}_0$. Pour vérifier une propriété de la forme $\langle a \rangle_{>\mathsf{p}} \langle b \rangle_{>\mathsf{q}} \phi$, on détermine d'abord les états qui satisfont la propriété ϕ. Cet ensemble est $[\![\phi]\!]$. Ensuite, on détermine l'ensemble des états pour lesquels il est possible de faire l'action b et d'atteindre un état de $[\![\phi]\!]$ avec une probabilité (strictement) supérieure à q. Soit B ce dernier ensemble. Enfin, on détermine l'ensemble des états pour lesquels il est possible de faire l'action a et de se rendre dans un état de B avec une probabilité supérieure à p.

Présentons maintenant quelques exemples de vérification de propriétés.

Exemple 3.2.1. *Considérons, le LMP de l'exemple 3.1.4. Vérifions la formule* $\langle \text{a} \rangle_{>0.9} \top$, *c'est-à-dire, vérifions si, à partir de l'état initial, il est possible de prendre une transition* a *avec une probabilité supérieure à* 0.9. *Pour ce faire, vérifions si l'état initial appartient à l'ensemble des états* x *pour lesquels* $\mu_a(x, S) > 0.9$. *Si c'est le cas, alors nous pourrons conclure que* $\mathbf{S}$ *vérifie la formule* $\langle \text{a} \rangle_{>0.9} \top$. *De l'exemple 3.1.4, nous avons* $\mu_a(x, S) > 0.9$ *pour* $x \in]\frac{6}{10}, 1]$. *Puisque l'état initial est* 1 *et que* 1 *appartient à* $]\frac{6}{10}, 1]$, *alors* $\mathbf{S}$ *vérifie la propriété* $\langle a \rangle_{>0.9} \top$.

Du fait que l'espace d'états est infini, la vérification formelle des LMP ne peut être faite que sur des sous-familles de processus qui manifestent une certaine forme de régularité dans leur comportement. De plus, notons que la logique $\mathcal{L}_0$ ne définit que des propriétés du type « *NEXT* », c'est-à-dire, des propriétés ne faisant appel qu'à une seule exécution. CISMO [1], développé par Richard [38], est un outil de vérification des propriétés probabilistes. Un langage pour les LMP a été défini dans le but d'utiliser CISMO : les espaces d'états permis sont des puissances de $\mathbb{R}$. Le langage en entrée à l'outil est un LMP dont les fonctions de transitions probabilistes sont des combinaisons de distributions connues telles que les lois uniformes, normales, ou toute autre fonction

1. Pour plus d'informations, consulter le site de CISMO : http://www2.ift.ulaval.ca/~jodesharnais/VPS/.

de répartition. La vérification des propriétés faite par CISMO se base sur l'exploration des états. Il existe une autre méthode qui consiste à utiliser la relation entre deux modèles, l'un étant la propriété à vérifier, pour déduire les résultats de vérification formelle. Cette relation peut être soit une simulation ou bisimulation. On définit ces notions comme suit.

Définition 3.2.2. *Soient* $\mathbf{S}_1 := (S_1, i_1, \mathrm{Act}, \mathrm{AP}, L, \{\mu_a | a \in \mathrm{Act}\})$ *et* $\mathbf{S}_2 := (S_2, i_2, \mathrm{Act}, \mathrm{AP}, L, \{\mu_a | a \in \mathrm{Act}\})$ *deux LMP. Une relation* $R \subseteq S_1 \times S_2$ *est une* simulation *de* $\mathbf{S}_1$ *par* $\mathbf{S}_2$ *si :*

- i_1 *est en relation avec* i_2 *;*
- *chaque fois que* $s_1 R s_2$*, alors si* $s_1 \xrightarrow{a} \mu_1$*, pour* $a \in \mathrm{Act}$ *alors* $s_2 \xrightarrow{a} \mu_2$ *et* $\mu_1(s_1, X) \leq \mu_2(s_2, R(X))$ *pour tout* X*.*

On dira donc que $\mathbf{S}_1$ *est simulé par* $\mathbf{S}_2$*, noté* $\mathbf{S}_1 \preceq \mathbf{S}_2$*. Si* R *est symétrique, on dit que* R *est une bisimulation. De manière équivalente, une relation d'équivalence est une* bisimulation *si dans la condition ci-dessus, on a* $\mu_1(X) = \mu_2(X)$ *pour la classe d'équivalence* X*. On dit alors que* $\mathbf{S}_1$ *et* $\mathbf{S}_2$ *sont bisimilaires, noté* $\mathbf{S}_1 \equiv \mathbf{S}_2$*.*

Comme corollaire, ces relations permettent de déduire des résultats de vérification de propriétés sur des modèles. Ainsi, si $\mathbf{S}_1$ et $\mathbf{S}_2$ sont tels que $\mathbf{S}_1 \preceq \mathbf{S}_2$, alors si $\mathbf{S}_1$ satisfait une propriété de $\mathcal{L}_0$ alors $\mathbf{S}_2$ la satisfait aussi. Dans le cas où on a $\mathbf{S}_1 \equiv \mathbf{S}_2$, alors $\mathbf{S}_1$ et $\mathbf{S}_2$ satisfont les mêmes propriétés de $\mathcal{L}_0$ [14].

Dans ce chapitre, nous avons présenté la théorie qui sous-tend les processus de Markov étiquetés. Ceux-ci peuvent être utilisés pour modéliser aussi bien des systèmes à espace d'états discret que continu. Dans le premier cas, les LMP se comportent exactement comme les DTMC. Par contre, lorsque leur espace d'états est continu, ils englobent les DTMC. En effet, les ensembles d'états dans les LMP sont plus importants en taille, ce qui induit un nombre de transitions plus élevé. De plus, les fonctions de transitions de probabilité sont continues, permettant d'exprimer les probabilités des prochains états en fonction des paramètres de l'état courant, et d'utiliser toute fonction réelle (loi de distribution, fonctions exponentielles, etc.) pour exprimer la valeur des probabilités. Nous avons également présenté la logique sur laquelle se base CISMO pour vérifier des propriétés sur les LMP.

Dans le chapitre suivant, nous présentons une autre classe de systèmes probabilistes à espace d'états non-dénombrable : la classe des systèmes hybrides probabilistes.

Chapitre 4

Les Systèmes Hybrides Probabilistes

4.1 Introduction

Dans ce chapitre, nous présentons le formalisme pour la description des systèmes hybrides ayant des comportements probabilistes. De façon informelle, un système hybride est un système dynamique faisant intervenir explicitement et simultanément des phénomènes ou des modèles de types dynamiques continus et événementiels. Ainsi, tout comme un système modélisé par un automate temporisé, un système hybride est la combinaison de deux composantes, une continue et une discrète. La composante continue assure l'évolution des paramètres physiques dans le temps tandis que la composante discrète contrôle leur évolution via des transitions discrètes. Toutefois, malgré leurs comportements communs, les systèmes hybrides sont plus généraux que les automates temporisés. En effet, contrairement aux automates temporisés, les systèmes hybrides supportent des variables plus complexes, et des évolutions plus générales pour celles-ci. Ainsi, les paramètres physiques sont des réels et leurs évolutions peuvent être définies par des équations différentielles de degré supérieur à 1. De plus, suite à l'exécution d'une action discrète dans un système hybride, les paramètres physiques peuvent prendre n'importe quelle valeur dans $\mathbb{R}$, contrairement aux automates temporisés, où les valeurs possibles sont 0 ou $*$.

Considérons l'exemple suivant, très souvent utilisé dans la littérature [27, 26]. Un thermostat contrôle la variation de la température dans une chambre par le biais d'un radiateur. On désigne par x la variable de la température et $\dot{x}$ sa première dérivée

par rapport au temps. Le système complet possède trois modes de fonctionnement représentant respectivement le fait que le radiateur est soit allumé, éteint ou en panne. Lorsque le radiateur est allumé, la température croît suivant l'équation $\dot{x} = -x + 5$ tandis qu'elle décroît suivant l'équation $\dot{x} = -x$ lorsque le radiateur est éteint. Lorsque le système est en panne, aucune variation de température n'a lieu. Les valeurs de la température évoluent dans l'intervalle [1,3]. De plus, lorsque la température atteint 3 unités, le radiateur doit s'éteindre alors qu'il doit s'allumer lorsque la température est 1. Enfin, lorsque l'on essaie d'allumer le radiateur, il peut soit s'allumer soit tomber en panne.

Dans l'exemple précédent, la composante continue englobe toutes les situations dans lesquelles la température est sujette à une variation dans le temps. Chaque situation, qui est un mode (ou une location), est constituée d'un ensemble d'états dans lesquels le système a le même comportement en évoluant à un rythme donné. On distingue donc trois modes : celui dans lequel le radiateur est allumé, celui dans lequel le radiateur est éteint, et celui dans lequel il est en panne. La composante discrète fait passer le système d'un mode à un autre.

Comme autres exemples de systèmes hybrides, on peut citer les contrôleurs de machines d'usine, les équipements médicaux, etc.

Les systèmes hybrides probabilistes représentent les systèmes dynamiques dont le comportement est incertain et pour lesquels on dispose d'une quantification de cette incertitude. Ainsi, considérons l'exemple du thermostat et supposons que dans 1% des cas, le radiateur tombe en panne lorsqu'on essaie de l'allumer. Il s'agit là d'un système hybride probabiliste. Par sa nature, un système hybride probabiliste se trouve être la combinaison d'une composante continue et d'une composante discrète probabiliste qui peut être représenté par un DTMC.

L'un des formalismes les plus utilisés pour la spécification et la vérification formelles des systèmes hybrides (non probabilistes) est celui des *automates hybrides* qui ont été introduits par Alur et al. [1] et Nicollin et al. [36]. Ce formalisme, généralisant celui des automates temporisés probabilistes, a été adapté dans le cas probabiliste, donnant ainsi naissance aux automates hybrides probabilistes [41, 40, 42]. Dès lors, les transitions entre états sont accompagnées de distributions de probabilité. Plusieurs études ont été faites sur les automates hybrides dont celles de Henzinger et al. [26, 27], Kotini et al [25], Kopre et al. [32]. Il a été montré que la classe des *automates hybrides rectangulaires* [32] est la plus grande classe pour laquelle la vérification formelle est décidable. Du côté des probabilistes, peu d'ouvrages ont été dédiés à l'étude des automates hybrides. Parmi ceux existants, nous pouvons distinguer ceux de Sproston [41, 40, 42] qui a étudié la décidabilité de

plusieurs sous-classes d'automates hybrides probabilistes rectangulaires à distribution discrète. De ses travaux, il en a découlé que la plus grande classe d'automates hybrides probabilistes à distribution discrète pouvant être vérifiés est celle des rectangulaires. Un de nos travaux [7] a également porté sur la vérification des systèmes hybrides probabilistes *non linéaires* à distribution discrète. Il s'agit de systèmes hybrides qui possèdent au moins une variable dont l'évolution peut être représentée par une équation différentielle de la forme $\dot{x} = ax + b$ où $a,\ et\ b$ sont réels et a non nul. Nous avons montré comment utiliser la substitution d'horloges et l'approximation par portrait, deux techniques proposées par Henzinger et al. [27] pour la vérification des automates non linéaires non probabilistes, sur des automates hybrides probabilistes non linéaires [7]. Comme résultat, il est possible de vérifier tout automate hybride probabiliste non linéaire à distribution discrète en utilisant l'une de ces deux méthodes d'approximation. Par conséquent, les automates hybrides probabilistes non linéaires à distribution discrète sont vérifiables, ce qui agrandit la classe des automates hybrides probabilistes vérifiables. Nous présentons ces résultats dans la section 4.5 et plus en détail à l'annexe C.

Dans ce chapitre, nous présentons les systèmes hybrides probabilistes, en utilisant la définition de Sproston [41, 40, 42]. Nous présentons également nos résultats sur l'extension des techniques de Henzinger et al. [27] qui ont été publiés dans un atelier international.

4.2 Définition et exemples

Nous définissons, dans cette section, les automates hybrides probabilistes (PHA, de l'anglais *Probabilistic Hybrid Automata*), et plus précisement ceux rectangulaires.

Comme pour tout système continu, la vérification des automates hybrides est une tâche ardue si bien que les travaux dans ce sens se limitent généralement à des sous-classes [29]. Il existe plusieurs classes d'automates hybrides, dont les principales sont : la classe des *rectangulaires* et celle des *non rectangulaires*. Chacune de ces sous-classes se différencie de l'autre par son comportement discret et/ou continu.

Les automates hybrides rectangulaires englobent plusieurs sous-classes d'automates hybrides qui présentent des similitudes au niveau comportemental, et pour lesquelles la vérification formelle est décidable. Au nombre de ces sous-classes, on peut citer les automates linéaires, les automates temporisés, les stop-watch, etc [42]. D'une manière générale, un automate hybride est dit rectangulaire lorsque les contraintes sur ses paramètres physiques sont des prédicats rectangulaires (voir la définition 2.3.3),

autrement dit, tout paramètre physique évolue suivant une équation de la forme $x \sim k$ où $\sim \in \{\leq, <, =, >, \geq\}$ et $k \in \mathbb{R}$. Ainsi, les automates linéaires assurent une évolution linéaire des paramètres physiques de la forme $\dot{x} = k$ où $k \in \mathbb{R}$. Les automates temporisés, quant à eux, sont un sous-ensemble des automates hybrides linéaires, parce que l'évolution se fait suivant une horloge, c'est-à-dire que le réel k y prend la valeur 1.

Quant aux automates hybrides non rectangulaires, ils regroupent un ensemble d'automates pour lequels l'évolution d'au moins un des paramètres ne peut être représentée par un prédicat rectangulaire. Un exemple de sous-classe d'automates non rectangulaires est celle des non linéaires, qui comme le nom l'indique, assurent une évolution non linéaire d'au moins un paramètre. Si x est ce paramètre, alors son évolution est décrite par une équation différentielle $\dot{x} = ax + b$ où $a,\ et\ b$ sont réels et a non nul.

La définition formelle d'un automate hybride rectangulaire probabiliste est la suivante :

Définition 4.2.1 ([41]). *Un* automate hybride rectangulaire probabiliste (PHA) *est un tuple* $H = (X, V, \text{Init}, A, \text{inv}, \mathit{flow}, \mathit{prob}, \langle \mathit{pre} \rangle_{v \in V, a \in A})$ *tel que :*

- X *est un ensemble fini de variables réelles. Soit* n *la cardinalité de* X *;*
- V *est un ensemble fini de modes ;*
- $\text{Init} : V \to R(X)$ *est la fonction qui associe à chaque mode un ensemble de valuations initiales dans* $R(X)$ *;*
- A *est un ensemble fini d'actions ;*
- $\text{inv} : V \to R(X)$ *est la fonction qui associe à tout mode, un ensemble d'invariants dans* $R(X)$ *;*
- $\mathit{flow} : V \times \mathbb{R}^n \to R(\dot{X})$ *est la fonction qui définit l'évolution à partir d'un mode et d'une valuation ;*
- $\mathit{prob} : V \times A \to \mathcal{P}_{\mathit{fin}}(\text{Dist}(V \times R(X \cup \{*\}) \times \mathcal{P}(X)))$ *est la fonction qui associe tout mode et toute action à un ensemble fini non-vide de distributions de probabilités discrètes sur l'ensemble* $V \times R(X) \times \mathcal{P}(X)$*. Par conséquent, chaque couple* (v, a) *aura un ensemble de distributions de probabilité associées, désigné par* $\mathit{prob}(v, a) = \{\mu_1, ..., \mu_m\}$ *pour* $m \geq 0$ *;*
- $\mathit{pre}_{v,a} : \mathit{prob}(v, a) \to R(X)$ *définit les* préconditions *pour les distributions de probabilité.*

La définition des automates hybrides probabilistes présente des similitudes avec celle des automates temporisés probabilistes donnée au chapitre 2, surtout en ce qui concerne la définition des états et la sémantique des fonctions inv, Init, *prob* et *pre*. En effet, comme dans le cas des automates temporisés, un état est un couple de la forme (v, a) où v est un mode et a une valuation sur X. De plus, les valuations dans $\text{inv}(v)$ sont celles

qui peuvent être prises dans le mode v, et le système pourra rester dans le mode tant que l'état courant est admissible (voir la définition 2.3.5).

Pour des raisons de simplicité, nous considérons que nos PHA ont un seul état initial. Par conséquent, il existe un seul mode $v \in V$, tel que $\text{Init}(v)$ est un singleton, et pour tout autre $v' \in V$, on a $\text{Init}(v') = \emptyset$.

L'évolution continue des variables dans le temps est déterminée par la fonction d'évolution *flow*. Si (v, a) est un état admissible alors $\mathit{flow}(v, \mathsf{a})$ est un rectangle sur $\dot{X} = \{\dot{x}_1, ..., \dot{x}_n\}$, où $x_i \in X$ pour tout $i \leq n$; il s'agit de l'ensemble des valeurs que peut prendre la première dérivée des variables de X à partir de l'état (v, a). Lorsque $\mathit{flow}(v, \mathsf{a})(x_i) = 1$ pour toutes les variables x_i, on obtient un automate hybride *linéaire* probabiliste et plus précisement un automate temporisé probabiliste [42]. On peut donc remarquer que les automates hybrides probabilistes généralisent les automates temporisés probabilistes puisqu'ils définissent des contraintes plus générales sur les paramètres du système. Cela permet, entre autres, de rendre les transitions discrètes actives ou non, et de définir une évolution plus générale des variables par des équations ou inéquations différentielles plus complexes. C'est ainsi que dans l'exemple du thermostat, l'évolution de la température est représentée par les équations $\dot{x} = -x + 5$ et $\dot{x} = -x$. De plus, dans la définition 4.2.1, la fonction d'évolution est plutôt générale, en ce sens où, pour tout mode, elle permet de spécifier un rectangle cible différent pour n'importe quelle valuation.

Maintenant, définissons plus formellement les transitions dans un PHA. À l'instar d'un automate temporisé, un système hybride effectue des transitions continues et des transtions discrètes. Lorsqu'une transition continue s'exécute, le mode de l'automate reste fixe et seules les variables changent dans le temps conformément à la condition d'évolution définie par *flow*. Ainsi, une transition continue de durée $\sigma \in \mathbb{R}_{\geq 0}$ entre les états (v, a) et (v, a'), notée $(v, \mathsf{a}) \xrightarrow{\sigma} (v, \mathsf{a}')$, a lieu si

- soit $\sigma = 0$ et $\mathsf{a} = \mathsf{a}'$;
- soit $\sigma > 0$ et il existe une fonction d'équation différentielle $\gamma : [0, \sigma] \to \mathbb{R}^n$ avec $\dot{\gamma} : (0, \sigma) \to \mathbb{R}^n$ tel que $\gamma(0) = \mathsf{a}$, $\gamma(\sigma) = \mathsf{a}'$, et $\gamma(\epsilon) \in \text{inv}(v)$ pour tout $\epsilon \in (0, \sigma)$.

Quant aux transitions discrètes, leurs exécutions se font instantanément. Étant donné $a \in A$, une transition discrète étiquetée a peut être prise d'un état (v, a) s'il existe une distribution $\mu \in \mathit{prob}(v, a)$, choisie de façon non-déterministe, telle que $\mathsf{a} \in \mathit{pre}_{v,a}(\mu)$, c'est-à-dire si la précondition associée à la transition est satisfaite. On dit que μ_a est *active* en (v, a). Dans ce cas, la valeur $\mu_a(v', \mathsf{post}, \mathcal{X})$ est la probabilité associée à la transition étiquetée a de v vers v', où $\mathsf{post} \subseteq R(X \cup \{*\})$ est le rectangle des valuations

dans lequel un choix non-déterministe peut se faire. Comme dans la définition 2.3.4, la notation $*$ est utilisée pour désigner aucun changement sur la valeur de variables après une transition. Notons que pour pouvoir écrire $\mu_a(v', \mathsf{post}, \mathcal{X})$, il faut que l'état cible dans le mode v' soit admissible. Ceci permet donc de restreindre l'ensemble des postconditions d'une transition aux valuations des états admissibles du mode cible, c'est-à-dire aux invariants du mode cible. Plus formellement, on écrira $\mu_a(v', \mathsf{post}, \mathcal{X})$ si et seulement si $\mathsf{post} \subseteq \text{inv}(v')$. Supposons que $\mathsf{post} \subseteq \text{inv}(v')$. Seules les variables appartenant à $\mathcal{X}$ peuvent prendre une valeur choisie suivant l'ensemble des postconditions post. Les valeurs des autres variables restent inchangées.

Une autre restriction que nous imposons à nos automates hybrides probabilistes et qui agit essentiellement sur ses transitions est celle de son initialisation. On définit un *automate hybride initialisé* comme suit.

Définition 4.2.2. *H est dit* initialisé *suivant x si et seulement si pour toute transition, x reste inchangé ou x prend une et une seule nouvelle valeur pour toute valuation.*

Illustrons ces différents concepts à travers l'exemple suivant.

Exemple 4.2.3. *La figure 4.1 montre la représentation de l'automate hybride probabiliste du thermostat. Les trois modes sont représentés par OFF, ON, et DOWN, et l'ensemble des variables est réduit au singleton $\{x\}$. Les actions possibles sont "allumer" et "éteindre". L'état initial est $(OFF, 2)$ et on a donc $\text{Init}(OFF) = \{2\}$. Pour le mode OFF, l'ensemble des invariants est donné par $\text{inv}(OFF) = [1, 3]$, et l'évolution de la variable x est exprimée par l'équation différentielle flow(OFF) : $\dot{x} = -x$. Cet automate hybride est donc non-linéaire. Chaque transition discrète est étiquetée par un nombre réel qui est la probabilité d'exécution de l'action associée. Pour le mode OFF, on a prob$(OFF, allumer) = \{\mu\}$ où $pre_{OFF}(\mu) = \{1\}$, tels que $\mu(ON, \mathbb{R}, \{\}) = 0.9$ et $\mu(DOWN, \{0\}, \{x\}) = 0.1$. Remarquons que pour la transition vers le mode ON, les postconditions* post *sont telles que* post $:= \mathbb{R}$ *parce que $\mathcal{X} := \{\}$, autrement dit, parce que x n'est pas réinitialisée en ON. Par contre, en DOWN, la variable x est réinitialisée à 0, ce qui explique le fait que* post $:= \{0\}$. □

Considérons l'exemple suivant dans lequel on traite de plusieurs variables dans un automate hybride probabiliste.

Exemple 4.2.4. *Considérons un système qui mesure la performance d'une machine dans une usine de fabrication de ciment (inspiré de [6]). On suppose que cette performance est mesurée de façon continue sur une échelle de $[0, 1]$. Ainsi, lorsque la performance est 0.5, la machine est exploitée à 50% de ses ressources maximales. Nous supposons qu'il existe trois modes de fonctionnement des machines :*

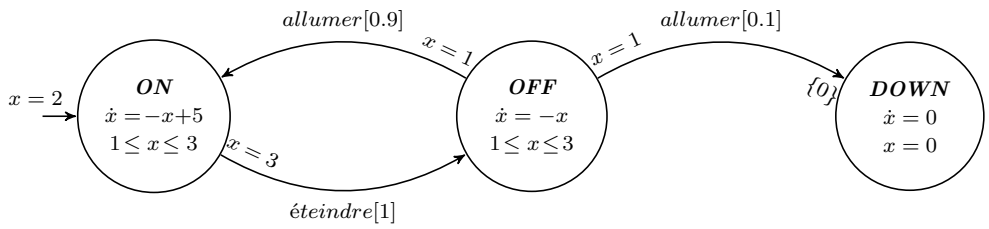

FIGURE 4.1 – La représentation graphique du PHA du thermostat

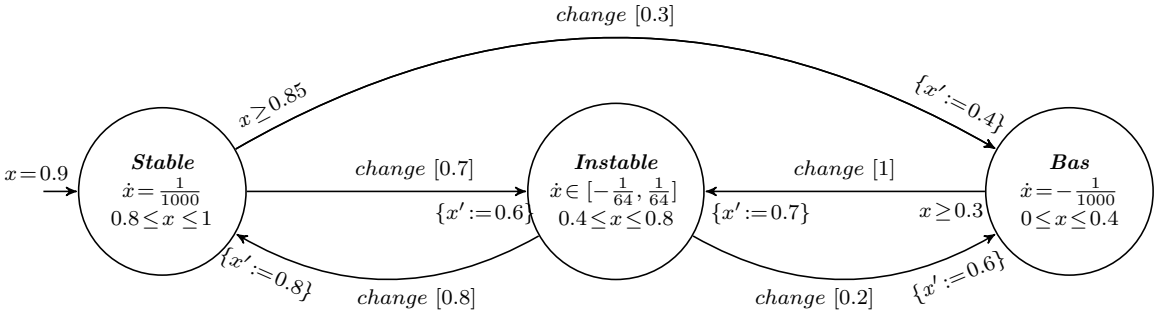

FIGURE 4.2 – Le PHA du système de surveillance de la performance

- ***Stable*** *qui est le mode normal de fonctionnement ; la performance de la machine croît de façon linéaire dans l'intervalle* $[0.8, 1]$ *;*
- ***Instable*** *où la performance varie à un rythme compris entre* $[-\frac{1}{64}, \frac{1}{64}]$ *dans l'intervalle* $[0.4, 0.8]$ *;*
- ***Bas*** *qui est le mode dans lequel la performance de la machine est plus faible que* 0.4*, tout en évoluant à un rythme linéaire ;*

La figure 4.2 montre une représentation graphique du système. La variable x *représente la performance de la machine. L'automate est composé de trois modes, et par conséquent,* $V := \{\textbf{\textit{Stable}}, \textbf{\textit{Instable}}, \textbf{\textit{Bas}}\}$*. L'état initial est* $(\textbf{\textit{Stable}}, 0.9)$ *; ainsi* $\mathrm{Init}(\textbf{\textit{Stable}}) := \{0.9\}$ *et* $\mathrm{Init}(v)$ *est* $\emptyset$ *pour les autres modes* $v \in V$*. Pour l'action change à partir de* ***Stable****, l'ensemble des distributions associé est* $\{\mu\}$ *tel que* $\mu(\textbf{\textit{Instable}}, \{0.6\}, \{x\}) = 0.7$ *et* $\mu(\textbf{\textit{Bas}}, \{0.4\}, \{x\}) = 0.3$*. L'ensemble des préconditions de la distribution* μ *est* $pre_{\textbf{\textit{Stable}}}(\mu) = [0.85, 1]$*. Les états admissibles du mode* ***Stable*** *sont de la forme* $(\textbf{\textit{Stable}}, \mathsf{a})$ *où* $\mathsf{a} \in [0.8, 1]$ *; par conséquent* $\mathrm{inv}(\textbf{\textit{Stable}}) := \{x : 0.8 \leq x \leq 1\}$*. Pour l'évolution de flux sur* ***Stable****, les paramètres* a *des états admissibles évoluent au rythme* $\frac{1}{1000}$ *; donc* $flow(\textbf{\textit{Stable}}, \mathsf{a}) \in \{\frac{1}{1000}\}$ *pour* $\mathsf{a} \in \mathbb{R}$*. Cependant, dans le mode* ***Instable****, l'évolution est, cette fois, non-déterministe parce qu'en tout état admissible, l'évolution du paramètre* x *peut varier à n'importe quel rythme à l'intérieur du rectangle* $[-\frac{1}{64}, \frac{1}{64}]$*, si bien qu'on a, pour tout* $\mathsf{a} \in [0.4, 0.8]$*,* $flow(\textbf{\textit{Instable}}, \mathsf{a}) \in [-\frac{1}{64}, \frac{1}{64}]$*.*

Dans les exemples précédents, nous pouvons remarquer que les automates hybrides probabilistes sont effectivement une généralisation des automates temporisés probabilistes. C'est ainsi que certaines méthodes de vérification des automates hybrides probabilistes s'inspirent de celles utilisées pour vérifier les automates temporisés probabilistes. Nous abordons, dans la section suivante, quelques techniques utilisées pour vérifier les PHA.

4.3 Vérification formelle

À cause de leur nature continue, la vérification formelle des systèmes hybrides probabilistes est un processus ardu si bien qu'elle ne peut être faite que sur des sous-classes de systèmes hybrides. Ainsi, pour les non-probabilistes, Henzinger et al. [26] ont proposé deux méthodes qui permettent de faire la vérification des propriétés de sureté sur des automates hybrides initialisés. Quant aux probabilistes, Sproston a proposé des méthodes pour vérifier des sous-classes de ces systèmes [42] représentées par des PHA rectangulaires munis de distributions discrètes de probabilité [41, 40]. Les propriétés vérifiées sont exprimées dans la logique ∀-PBTL qui regroupe un ensemble de propriétés de sûreté probabilistes en rapport avec le temps. Un exemple de propriété ∀-PBTL pouvant être vérifiée sur le système de l'exemple 4.2.4 est **P** :«dans 99% des cas, le système reste stable avant de passer dans un régime bas». Plus formellement, la logique utilisée par Sproston, notée $\mathcal{T}$, a la syntaxe est la suivante :

$$\Phi ::= \top \mid \mathrm{p} \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid [\Phi_1 \forall\mathcal{U} \Phi_2]_{\backsim \mathsf{q}}$$

où p est une proposition atomique, $\mathsf{q} \in [0,1]$, et $\backsim \in \{\geq, >\}$.

Suivant cette logique, la propriété **P** est exprimée par la formule $[(\boldsymbol{Stable})\forall\mathcal{U}(\boldsymbol{Bas})]_{\geq 0.99}$, où ***Stable*** et ***Bas*** sont les propositions atomiques d'états.

La sémantique et la satisfaction de $\top$, p, $\wedge$ et $\neg$ sont les mêmes que celles de la logique $\mathcal{L}_0$ présentée dans le chapitre précédent à la section 3.2. Par contre, la sémantique de l'opérateur probabiliste $[\Phi_1 \forall\mathcal{U} \Phi_2]_{\backsim q}$ est définie en utilisant l'interprétation de l'opérateur $\mathcal{U}$ sur un chemin. Sans entrer dans les détails, nous définissons un chemin comme une suite d'état. Ainsi, l'interprétation de $\mathcal{U}$ est la suivante pour un chemin ω :

$$\omega \vDash \Phi_1 \mathcal{U} \Phi_2 \Leftrightarrow \exists i \text{ tel que } \omega_i \vDash \Phi_2, \ \textit{et } \forall j \leq i, \omega_j \vDash \Phi_1.$$

Autrement dit, w satisfait $\Phi_1 \mathcal{U} \Phi_2$ si et seulement s'il existe i tel que ω_i satisfait Φ_2 et Φ_1 est satisfait par tous les points précédents ω_i. Pour définir la relation de satisfaction de la formule $[\Phi_1 \forall \mathcal{U} \Phi_2]_{\backsim q}$ fait appel à la probabilité d'un chemin. La probabilité d'un chemin est donc la probabilité que cette suite d'états soit parcourue par le système. La relation de satisfaction de la formule $[\Phi_1 \forall \mathcal{U} \Phi_2]_{\backsim q}$ est donc :

$$s \vDash [\Phi_1 \forall \mathcal{U} \Phi_2]_{\backsim q} \Leftrightarrow \Pr(\omega \mid \omega \vDash \Phi_1 \mathcal{U} \Phi_2) \backsim q,$$

où Pr est la probabilité de l'ensemble des chemins w. Cette probabilité se définit de façon standard [40] mais puisque nous n'en avons pas besoin dans ce mémoire, nous n'entrerons pas dans les détails.

La vérification des propriétés de ∀-PBTL sur des PHA rectangulaires à distribution discrète faite par Sproston se base sur un résultat : les propriétés ∀-PBTL préservent les relations d'équivalence telle que la bisimilation. De ce fait, le problème de vérification d'une propriété ϕ sur un PHA peut être simplifié à celui d'un PHA qui lui est équivalent par ϕ.

À part les méthodes développées pour la vérification des PHA, il en existe une autre qui consiste en l'utilisation de la sémantique des PHA selon laquelle tout PHA a un système de transition probabiliste sous-jacent. Cette sémantique peut servir lors de la vérification des systèmes hybrides probabilistes à distribution discrète. La vérification se fait sur le système de transition sous-jacent grâce à un model-checker de systèmes de transitions probabilistes à espace d'état continu. C'est ainsi que le model-checker de systèmes de transition probabilistes PRISM a été utilisé pour vérifier des systèmes hybrides probabilistes à distribution discrète [41]. Nous présentons la sémantique des automates hybrides probabilistes dans la section suivante.

4.4 Sémantique

Dans sa définition, Sproston a proposé une description formelle de la sémantique des automates hybrides probabilistes, en définissant l'interprétation de ceux-ci comme des systèmes de transitions probabilistes [40]. Dans ces systèmes, des actions silencieuses traduisent la nature du temps-continu dans les automates hybrides probabilistes. On dérive, par conséquent, des systèmes probabilistes à espace d'états continu, mais pour lesquels les transitions sont exécutées de manière discrète. Ils sont essentiellement des LMP avec des fonctions de transitions non-déterministes et discrètes avec un nombre infini d'actions.

Les LMP dérivés sont essentiellement utilisés dans un but théorique. En effet, les PHA sont un langage pour modéliser les phénomènes dynamiques continus, et on a besoin de formaliser ce que la syntaxe de ce langage représente. De plus, la sémantique en tant que LMP est également utilisée pour définir les relations entre les PHA, la plus connue étant la notion d'équivalence de bisimulation. Les LMP obtenus ne peuvent être utilisés en pratique et ne peuvent pas être formellement vérifiés vu la taille de leurs espaces d'états et le grand nombre de leurs transitions. En effet, tous les changements d'états qui ont lieu en continu sont non-dénombrables et indépendants des transitions non-déterministes. C'est pour cela que, dans la pratique, l'utilisation des LMP dérivés pour la vérification des systèmes hybrides probabilistes fait appel à des relations d'équivalence, la simulation et la bisimulation, comme défini à la section 3.2.2 : à partir du LMP obtenu, on dérive un système moins général qui est utilisé pour la vérification.

Soit H un automate hybride probabiliste. Avant de présenter plus formellement l'interprétation de H, nous avons besoin de quelques notions qui permettent de définir l'ensemble des états cibles d'une transition dans un PHA [42]. Nous avons légèrement changé la notation de Sproston [42], dans le but de rendre la définition plus simple. Soit μ une distribution de probabilité. Rappelons que $\text{supp}(\mu)$ est l'ensemble des éléments c pour lesquels $\mu(c) > 0$. Ainsi, dans un automate hybride probabiliste, $\text{supp}(\mu)$ est l'ensemble des éléments $(v', \mathsf{post}, \mathcal{X}) \in V \times 2^{\mathbb{R}^n} \times 2^X$ tels que $\mu(v', \mathsf{post}, \mathcal{X}) > 0$. Soit Q l'ensemble des états admissibles d'un PHA. Pour chaque $(v, \mathsf{a}) \in Q$ et chaque action $a \in A$, nous voulons définir un ensemble (possiblement non-dénombrable) de distributions discrètes dans $\text{Dist}(Q) \subseteq \text{Dist}(V \times \mathbb{R}^n)$, à partir de l'ensemble $prob(v, a)$, qui contient des distributions dans $\text{Dist}(V \times \mathcal{P}(\mathbb{R}^n) \times X)$. Chacune des distributions $\mu \in prob(v, a)$ donnera naissance à plusieurs distributions sur $\text{Dist}(Q)$ puisqu'elles sont des combinaisons $\langle \mathsf{b} \rangle$ telles que $\langle \mathsf{b} \rangle := \langle \mathsf{b}_1, \mathsf{b}_2, \ldots, \mathsf{b}_m \rangle$, avec $\mathsf{b}_i \in \mathsf{post}_i$ et $\text{supp}(\mu) = \{(v_i, \mathsf{post}_i, \mathcal{X}_i)\}_{i=1}^m$. Désignons par $Cible(\mu)$ l'ensemble de toutes ces combinaisons. Cet ensemble est nécessaire puisque les postconditions post_i peuvent se chevaucher et peuvent avoir en commun le même mode cible v_i. Toute combinaison représente un choix non déterministe défini à partir de μ.

La sémantique d'un PHA est définie comme suit :

Définition 4.4.1. *Étant donné un PHA* $H = (X, V, \text{init}, A, \text{inv}, flow, prob, \langle pre_v \rangle_{v \in V})$, *on dérive le système de transition probabiliste infini associé*

$$C = (Q, I, A \cup \mathbb{R}, Steps) \text{ comme suit :}$$

- $Q \subseteq V \times \mathbb{R}^n$ *est l'ensemble des états admissibles ;*
- $I = \{(v_0, \mathsf{a}_0) \in Q \mid \mathsf{a}_0 \in \text{init}(v_0)\}$ *est l'ensemble des états initiaux ;*
- $Steps(v, \mathsf{a}) := Cts(v, \mathsf{a}) \cup Dis(v, \mathsf{a})$, *pour l'état* $(v, \mathsf{a}) \in Q$, *et :*
  - $Cts(v, \mathsf{a}) \subseteq \mathbb{R} \times Q$ *contient les transitions de délai : toutes les paires* $(d, (v, \mathsf{b}))$ *telles que* $d \in \mathbb{R}_{\geq 0}$, $\mathsf{b} \in \text{inv}(v)$, *et qu'il existe une fonction différentiable*

$f : [0, d] \to \mathbb{R}^n$ *avec* $\dot{f} : (0, d) \to \mathbb{R}^n$ *telle que* $f(0) = \mathsf{a}$, $f(d) = \mathsf{b}$, *et pour tout* $\epsilon \in (0, d)$, $\dot{f}(\epsilon) \in \mathit{flow}(v, f(\epsilon))$ *et* $f(\epsilon) \in \mathrm{inv}(v)$;

– $\mathit{Dis}(v, \mathsf{a}) \subseteq A \times \mathrm{Dist}(Q)$ *contient, pour chaque* $\mu \in \mathit{prob}(v, a)$ *avec* $\mathsf{a} \in \mathit{pre}(\mu)$, *pour toute combinaison* $\langle \mathsf{b} \rangle \in \mathit{Cible}(\mu)$, *tous les couples* $(a, \mu_{\langle \mathsf{b} \rangle})$, *où* $\mu_{\langle \mathsf{b} \rangle}$ *est définie comme :*

$$\mu_{\langle \mathsf{b} \rangle}(v', \mathsf{c}) := \begin{cases} \sum\limits_{\substack{i=1 \\ \mathsf{c}=\mathsf{b}_i, v_i=v'}}^{|\mathrm{supp}(\mu)|} \mu(v_i, \mathsf{post}_i, \mathcal{X}_i) & \textit{si } \mathsf{c} \in \mathrm{inv}(v') \\ 0 & \textit{sinon.} \end{cases}$$

Si $s := (v, \mathsf{a})$ est un état de Q, alors $\mathit{Steps}(s)$ est l'ensemble des transitions quittant s. Du fait que cet ensemble regroupe toutes les transitions continues qui sont encapsulées dans la fonction d'évolution $\mathit{flow}(v, \cdot)$, et les transitions discrètes, il peut être infini. De plus, l'espace d'état Q contient tous les états admissibles du PHA, et par conséquent, il est infini et non-dénombrable. Le système de transition d'un PHA ne peut donc être vérifié sans que l'on n'ait recours à des méthodes d'approximation.

Les résultats présentés, jusque là, s'appliquent aux systèmes hybrides probabilistes rectangulaires et aux différentes classes de systèmes qu'ils englobent. Dans la section suivante, nous présentons brièvement les résultats de nos travaux sur les systèmes hybrides probabilistes non rectangulaires.

4.5 Analyse des systèmes hybrides probabilistes non-linéaires

Dans cette section, nous présentons un résumé des contributions que nous avons faites dans le domaine de l'analyse des automates hybrides probabilistes et qui ont été publiées dans un article international [7]. Les travaux qui avaient été faits dans ce domaine étaient essentiellement ceux de Sproston [40] qui a étudié la décidabilité des automates hybrides rectangulaires probabilistes. Depuis lors, les automates hybrides rectangulaires probabilistes étaient la plus grande classe d'automates hybrides probabilistes vérifiables. Nos contributions ont permis d'élargir cette plus grande classe à celle des automates hybrides non linéaires probabilistes. La méthode utilisée repose sur des techniques d'approximation développées par Henzinger et al. [26, 27] pour la vérification des automates hybrides non linéaires et non probabilistes. Nous avons adapté ces techniques dans le contexte probabiliste, proposant ainsi deux nouvelles techniques

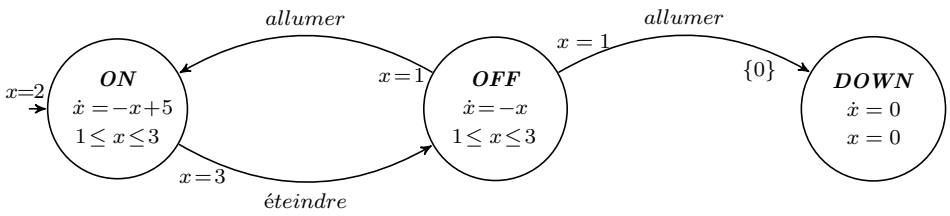

FIGURE 4.3 – La représentation graphique de l'automate non probabiliste du thermostat

pour la vérification des automates hybrides probabilistes. L'annexe C présente en détail ces techniques.

4.5.1 Les méthodes de Henzinger et al

Deux méthodes d'approximation ont été proposées par Henzinger et al. pour la vérification des automates hybrides non linéaires et non probabilistes. Voici un bref aperçu de ces méthodes.

Soit l'exemple suivant :

Exemple 4.5.1. *Considérons la version non probabiliste du thermostat 4.1. La figure 4.3 est la représentation graphique de l'automate obtenu.*

La substitution par horloge

La première méthode appelée *substitution par horloge*[1] (de l'anglais clock-translation), est une méthode basée sur la substitution des variables réelles représentant les paramètres physiques par des horloges. L'automate résultant de cette transformation est un automate temporisé qui a le même comportement que l'automate hybride. Toutefois, il faut noter que cette méthode ne peut s'appliquer sur un automate hybride que si celui-ci est *résoluble*. Les définitions suivantes permettent de définir un automate hybride *résoluble*.

Soit M un automate hybride non-linéaire.

Définition 4.5.2. *Soit x une variable et P_x un prédicat. On dit que P_x est un prédicat simple pour x si et seulement si il est une combinaison positive d'expressions booléennes de la forme $x \sim c$ où $c \in \mathbb{R}$ et $\sim \in \{<, \leq, =, \geq, >\}$.*

1. Il s'agit d'une traduction française non officielle.

Dans l'exemple 4.5.1, tous les prédicats des invariants sont simples sur $\{x\}$. Par contre, un prédicat comme $y = x + 2$ n'est pas simple sur $\{x, y\}$.

Une des conditions requise pour qu'une variable soit résoluble dans un M est que celle-ci n'apparaisse que dans des prédicats simples. Plus concrètement, on définit la résolubilité d'une variable dans M comme suit.

Définition 4.5.3. *On dit qu'une variable x is* résoluble *dans M si :*

- *chaque condition initiale, condition d'invariants, et précondition de M définit des prédicats simples pour x et chaque condition sur l'évolution de x dans* $\mathrm{Flow}(v)$ *est de la forme $(\dot{x} = f_x^v(x)) \wedge P_x$, où P_x est un prédicat simple pour x ; les évolutions des autres variables ne doivent pas dépendre de x ou de $\dot{x}$;*
- *le problème $\dot{y}(t) = f_x^v(y(t))$; $y(0) = c$ admet une solution unique, continue et strictement monotone g_c ;*
- *M est initialisé pour x.*

Si pour tout $x_i \in V$, x_i est résoluble dans M alors on dit que M est résoluble.

Exemple 4.5.4. *Considérons la version non probabiliste du thermostat de la figure 4.3. Cet automate est résoluble puisque l'équation décrivant l'évolution de la variable x admet une solution dans tous les modes : dans le mode ON, l'équation différentielle $\dot{x} = -x + 5$ avec la condition initiale $x(0) = 2$ a comme solution $x(t) = -3e^{-t} + 5$ où $t \in \mathbb{R}_+$.*

Supposons que M est résoluble. La substitution par horloge de M consiste à substituer toutes les occurrences de $x_i \in X$ dans les prédicats simples pour x_i qui définissent les conditions d'initialisation, les invariants, les préconditions et les évolutions par une variable t_i grâce à la transformation suivante.

Transformation de $x \sim l$ en $t_x \sim' g_c^{-1}(l)$. Soit $g_c(t)$ l'unique solution de l'équation $\dot{y}(t) = f_x^v(y(t)); y(0) = c$, où $c \in \mathbb{R}$. Puisque $g_c(t)$ est strictement monotone, il existe au plus une variable $t \in \mathbb{R}_+$ telle que $g_c(t) = l$ pour chaque $l \in \mathbb{R}$. Soit $g_c^{-1}(l) = t$ si $g_c(t) = l$ et $g_c^{-1}(l) = -$ si $g_c(t) \neq l$ pour tout $t \in \mathbb{R}_+$. Soit O l'ensemble des opérateurs relationnels $<, \leq, =, \geq,$ and $>$. La transformation d'un prédicat simple pour $\{x\}$ vers un prédicat simple pour $\{t_x\}$ est la fonction α_c définie en utilisant $\sim \in O$, $lt : O \to O$ et $gt : O \to O$, comme suit :

$$\alpha_c(x \sim l) = \begin{cases} \mathit{vrai} & \text{si } g_c^{-1}(l) = - \text{ et } c \sim l. \\ \mathit{faux} & \text{si } g_c^{-1}(l) = - \text{ et } c \nsim l. \\ t_x \ lt(\sim)\ g_c^{-1}(l) & \text{si } g_c^{-1}(l) \neq - \text{ et } c \sim l. \\ t_x \ gt(\sim)\ g_c^{-1}(l) & \text{si } g_c^{-1}(l) \neq - \text{ et } c \nsim l. \end{cases}$$

$\sim$	$lt(\sim)$	$gt(\sim)$
$<$	$<$	$>$
$\leq$	$\leq$	$\geq$
$=$	$=$	$=$
$\geq$	$\leq$	$\geq$
$>$	$<$	$>$

Cette transformation agit également sur les modes, et tout mode v de M donne un ensemble de modes (v, c_i) où $c_i \in \mathbb{R}$ est une des valeurs initiales de v choisie telle que s'il existe une transition de v vers v' avec l'action a et la postcondition k pour x, alors il existe une transition étiquetée a de tous les (v, c_i) vers (v', k) dans l'automate obtenu.

Pour chaque (v, c_i) de l'automate obtenu, tout prédicat $x \sim l$ est remplacé par le prédicat $\alpha_{c_i}(x \sim l)$, sauf le prédicat définissant l'invariant qui est remplacé par $\alpha_{c_i}(x \sim l)$ si $c_i \sim l$, et par *faux* sinon ((v, c_i) peut être enlevé dans le dernier cas).

Exemple 4.5.5. *L'automate temporisé de la figure 4.4 est obtenu en appliquant la substitution par horloge sur l'automate du thermostat. Chaque mode v de l'automate est divisé en n modes, où n représente le nombre de valeurs initiales dans v. Puisque ON admet $\{1, 2\}$ comme ensemble de valeurs initiales pour la variable x, nous obtenons les modes (ON, 1) et (ON,2). Pour ces deux modes, les équations différentielles sont respectivement «$\dot{x} = -x + 5$, où $x(0) = 2$» et «$\dot{x} = -x + 5$, où $x(0) = 1$», et donc nous avons les solutions «$x(t) = -3e^{-t} + 5$» et "$x(t) = -4e^{-t} + 5$" respectivement. Ensuite, nous substituons la variable x par l'horloge t_x dans les prédicats définissant les préconditions, et les invariants. Ainsi, la contrainte $x \leq 3$ devient $t \leq \ln(2)$ et $t \leq \ln(\frac{3}{2})$ respectivement.*

Le théorème suivant permet de déduire des résultats de vérification formelle d'un automate hybride résoluble à partir de sa substitution par horloge.

Theorème 4.5.6. *[26] M est bisimilaire à sa substitution par horloge T. La relation qu'on utilise est $\varphi : S_T \to S_H$, définie sur $\varphi((v, c), \mathsf{a}) := (v, \mathsf{a}')$, où a' satisfait $\mathsf{a}'|_X = \mathsf{a}|_X$ et $\mathsf{a}'(x) = g_c(\mathsf{a}(t_x))$ où g_c est la solution de l'équation $[\dot{y}(t) = f_x^v(y(t)); y(0) = c]$.*

Comme corollaire, M et T satisfont les mêmes propriétés de la logique temporelle.

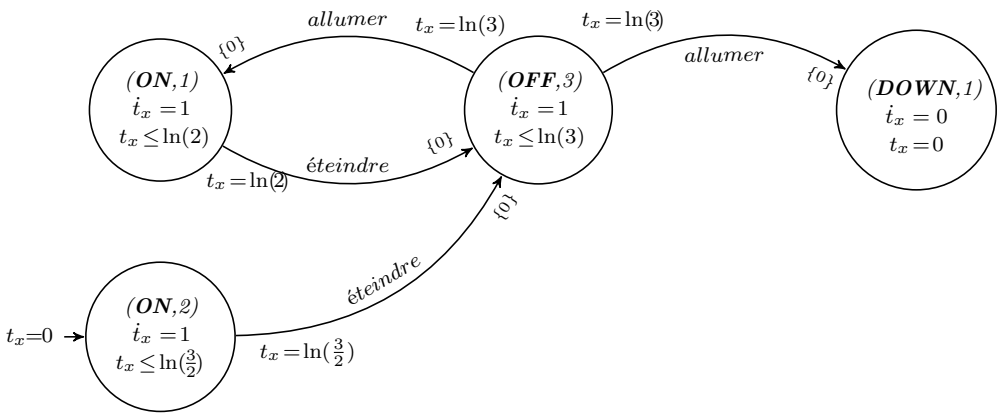

FIGURE 4.4 – La substitution par horloges de l'automate du thermostat

L'approximation linéaire

La seconde technique que nous désignons, ici, par *approximation linéaire* (de son nom original *phase-portrait approximation*) transforme, sans restriction aucune, tout automate hybride en un automate rectangulaire qui sera approximé jusqu'à l'obtention d'un automate pour lequel la vérification est décidable. L'automate hybride original est simulé par l'automate obtenu (voir la définition 3.2.2). Par conséquent, l'automate hybride obtenu possède les comportements de l'automate hybride original sans lui être équivalent.

D'une manière générale, l'approximation linéaire d'un automate hybride non probabiliste se fait en deux étapes : le *partitionnement* et l'*approximation*. Soit M un automate hybride quelconque.

Le partionnement de M est consiste à diviser l'ensemble des invariants en plusieurs parties. Ainsi, si v est un mode de M, alors le partionnement de v conduit à l'obtention d'un ensemble $D := \{v_0, v_1, ..., v_n\}$ et pour chaque $v_i \in D$, on associe un ensemble $\text{inv}_i(v_i)$ tel que $\text{inv}_i(v_i) \subseteq \text{inv}(v)$ et $\text{inv}(v) = \bigcup_{0 \leq i \leq n} \text{inv}_i(v_i)$. De plus, pour reprendre le comportement de l'automate hybride original, on relie les modes v_i par des transitions silencieuses. Soit H_p l'automate obtenu. M et H_p sont bisimilaires et on dit que H_p est une *partition* de M.

Exemple 4.5.7. *Considérons l'exemple non probabiliste du thermostat dont l'automate est représenté à la figure 4.3. L'automate représenté à la figure est une représentation de la partition de l'automate non probabiliste du thermostat. Le mode ON est partitionné en* $(ON, 1)$ *et* $(ON, 2)$ *avec* $\text{inv}((ON, 1)) = \{x \mid 1 \leq x \leq 2\}$ *et* $\text{inv}((ON, 2)) = \{x \mid 2 \leq x \leq 3\}$*, tandis que le mode DOWN n'a pas été partitionné.*

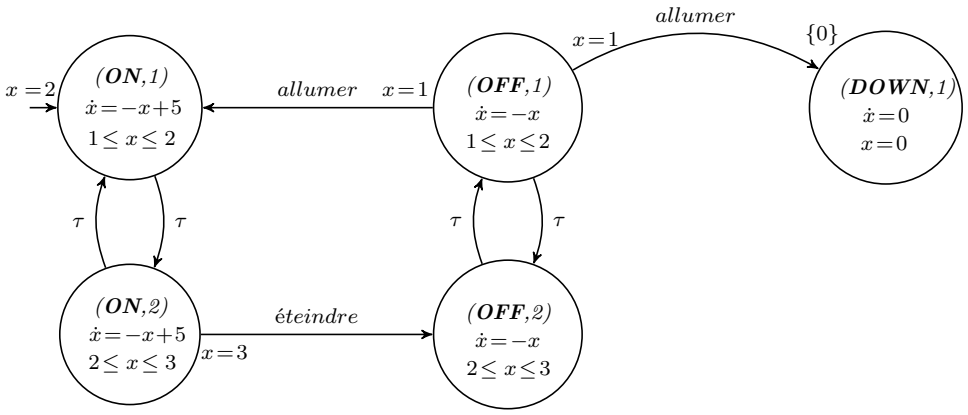

FIGURE 4.5 – Une partition du thermostat

On peut remarquer les transitions silencieuses entre les états $((ON, i), 2)$*, avec* $i = 1, 2$*, pour représenter la transition de* $(ON, \{x \mapsto 2\})$*, et ainsi préserver l'évolution dans le mode ON.*

Une fois la partition de l'automate obtenue, on l'approxime. L'approximation consiste à transformer la partition M_p de M pour obtenir un automate hybride rectangulaire. Cette transformation se fait en remplaçant toutes les équations d'évolution de la forme $\dot{x} = ax + b$ par des combinaisons de $\bigwedge \dot{x} \sim c$ où $\sim \in \{<, \leq, =, \geq, >\}$ en respect avec les invariants. Autrement dit, par cette transformation, on fait disparaître toutes les occurrences de x_i dans $\dot{x}_i$. Soit T l'automate hybride obtenu. On dit que T est une *approximation linéaire* de M.

Exemple 4.5.8. *L'automate de la figure 4.6 est une approximation du thermostat (avec la partition de la figure 4.5). Tout prédicat sur* $\dot{x}$ *est remplacé par un prédicat qui spécifie sa borne supérieure et inférieure. Par exemple, l'approximation de la variable* $\dot{x}$ *dans le mode (ON,1) permet d'obtenir l'ensemble* $\{\dot{x} \mid 3 \leq \dot{x} \leq 4\}$.

L'approximation linéaire T obtenue est utilisée pour vérifier M grâce au théorème suivant.

Theorème 4.5.9. *[26] Si* T *est une approximation de* H*, alors* T *simule* M.

L'automate de la figure 4.6 simule la partition de la figure 4.5, et par transitivité, simule l'automate hybride original de la figure 4.3. La conséquence du théorème 4.5.9 est que si une propriété est satisfaite par T, alors elle l'est également par M. Par contre, lorsque T ne satisfait pas une propriété, on ne peut pas directement conclure que M ne la satisfait pas. Dans ce dernier cas, on doit raffiner l'approximation en partitionnant davantage l'automate M afin d'obtenir une meilleure précision lors de la vérification.

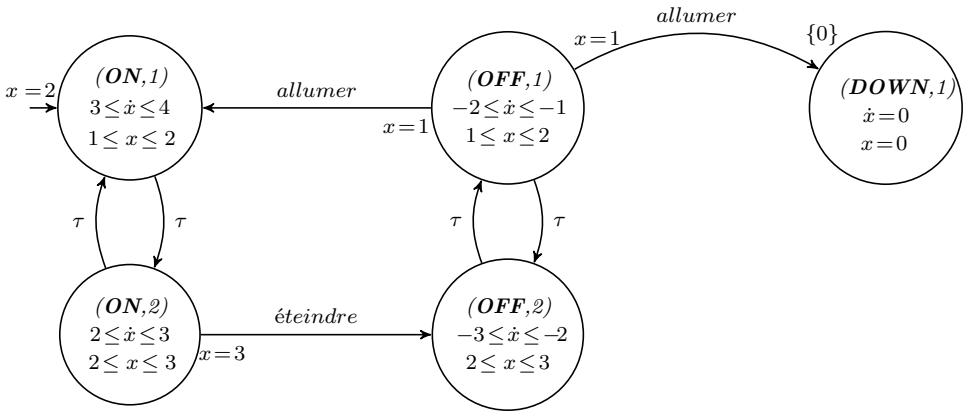

FIGURE 4.6 – Une approximation linéaire du thermostat

Les deux techniques d'approximation de Henzinger et al. étant définies, nous présentons, dans la prochaine section, leur application sur des automates hybrides probabilistes.

4.5.2 Application aux automates hybrides probabilistes

Nous présentons, ici, les méthodes d'approximation de Henzinger et al. appliquées aux automates hybrides probabilistes.

Substitution par horloge probabiliste

D'une manière générale, la méthode consiste à d'abord construire un automate hybride non probabiliste à partir du PHA. Ensuite, on applique la substitution par horloge telle que définie par Henzinger et al. pour obtenir un automate temporisé. Enfin, on ajoute les valeurs de probabilité sur les transitions de l'automate temporisé pour obtenir la *susbtitution par horloge probabiliste*. Aucune condition n'est requise pour dériver l'automate hybride non probabiliste à partir d'un probabiliste, mais il faut que l'automate dérivé respecte les conditions requises pour l'application de la substitution par horloge.

Dans le cas non probabiliste, la substitution de la variable x par l'horloge t_x est possible si et seulement si x est résoluble, et la méthode de substitution ne peut s'appliquer sur un automate que s'il est résoluble. Pour les automates hybrides probabilistes, la résolubilité d'une variable requiert une condition supplémentaire s'appliquant aux transitions discrètes probabilistes. Ainsi, si H est un automate hybride probabiliste, la

résolubilité de x se définit comme suit :

Définition 4.5.10 ([7]). *Une variable x d'un automate hybride probabiliste H est dite* résoluble *si :*

- *les deux premières conditions de résolubilité pour les systèmes hybrides non probabilistes sont satisfaites ;*
- *pour tout état admissible $s = (v, \mathsf{a})$ de H, si $s \xrightarrow{a} \mu$, et qu'il existe w tel que $\mu(w, \mathsf{post}, \cdot) > 0$, alors $\mathsf{post}(x) = \{r\}$ pour $r \in \mathbb{R}(X) \cup \{*\}$. De plus, si on a $\mathsf{post}(x) = \{*\}$, alors nous devons avoir $flow(v, \cdot)(x) = flow(w, \cdot)(x)$.*

On dit que H est résoluble si toutes ses variables sont résolubles.

La restriction sur les transitions probabilistes, dans la définition 4.5.10, permet de s'assurer que H est initialisé pour la variable x, et par conséquent, après toute transition discrète, la valeur de x ne peut être réinitialisée que si l'automate change de location.

Si H est résoluble, alors la substitution de ses variables conduit à l'obtention d'un automate temporisé probabiliste. La transformation que nous proposons se fait essentiellement en trois étapes. Ainsi, elle consiste à :

1. construire l'automate hybride non probabiliste correspondant à H en retirant toutes les valeurs de probabilité sur les transitions. Soit H_n l'automate hybride obtenu. H_n a les mêmes ensembles d'états, d'états initiaux, d'invariants, d'actions, de préconditions et de post-conditions que H. De plus, les transitions sont les mêmes dans les deux automates sauf qu'elles ne sont pas accompagnées de leurs valeurs de probabilité dans H_n.
2. substituer les variables de H_p par des horloges en utilisant la transformation non-probabiliste telle que décrite à la section 4.5.1. Désignons T_n l'automate temporisé obtenu suite à cette transformation. Par le théorème 4.5.6, T_n et H_n sont bisimilaires.
3. construire l'automate temporisé probabiliste T en ajoutant les valeurs de probabilité des transitions de H aux transitions en relation dans T_n. T est la substitution probabiliste de H.

Les détails de la transformation et la preuve que la construction de la substitution par horloge probabiliste conduit à l'obtention d'un automate temporisé probabiliste valide sont donnés dans l'annexe C.

Exemple 4.5.11. *La substitution par horloge de l'automate hybride probabiliste du thermostat est représentée à la figure 4.7 ; toutes les transitions obtiennent la probabilité*

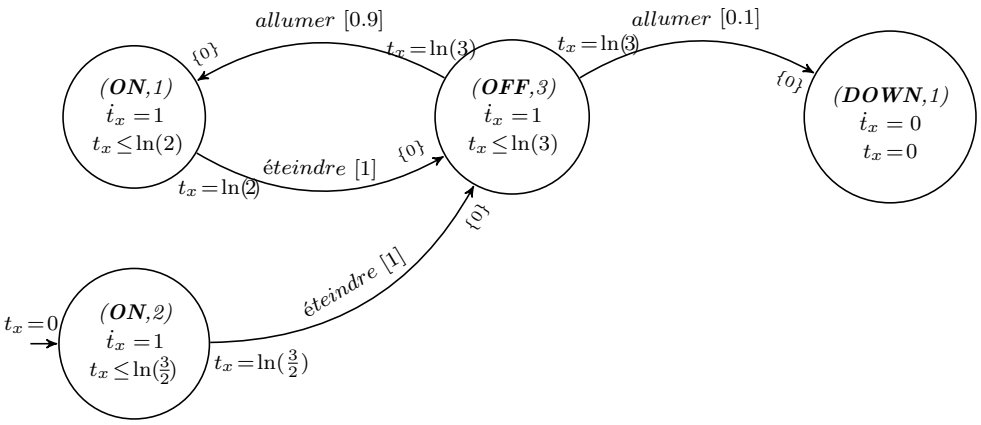

FIGURE 4.7 – La substitution probabiliste du thermostat

1 *sauf la transition on quittant l'état* $(OFF, 3)$ *qui a une probabilité de* 0.9 *vers* $(ON, 1)$ *et* 0.1 *vers* $(DOWN, 1)$.

La preuve de la validité de la substitution probabiliste avec horloges nous a permis de formuler le théorème suivant.

Theorème 4.5.12 ([7])**.** *Si T est la substitution par horloge de H, alors H et T sont bisimilaires.*

Comme corollaire, T et H satisfont les mêmes propriétés de la logique temporelle. La preuve du théorème 4.5.12 est donnée dans l'annexe C.

Dans la section suivante, nous présentons l'approximation linéaire que nous avons proposée pour les PHA.

L'approximation linéaire probabiliste

La méthode de l'approximation dans un contexte probabiliste suit des étapes semblables à celles de la substitution par horloge probabiliste, en passant également par la construction d'un automate hybride non probabiliste. Toutefois, cette transformation est plus simple, voire plus rapide, puisqu'aucune condition ne doit être satisfaite par l'automate non probabiliste.

Soit H un automate hybride probabiliste. L'approximation de H consiste à :

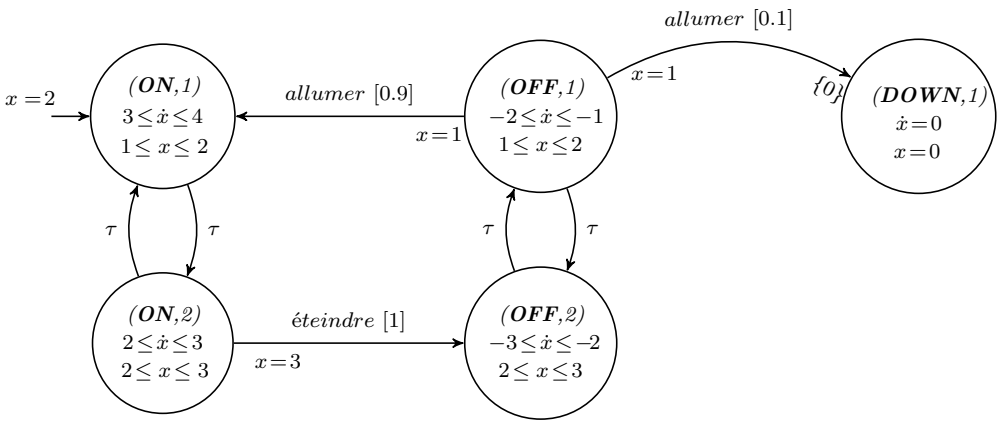

FIGURE 4.8 – Une approximation linéaire du thermostat

1. construire l'automate hybride non probabiliste dérivé de H. Cet automate est obtenu en retirant les valeurs de probabilité des transitions de H. Soit H_n l'automate hybride obtenu.
2. approximer H_n en appliquant l'approximation linéaire telle que décrite à la section 4.5.1. Soit T_n l'approximation obtenue.
3. ajouter les valeurs de probabilité de H sur les transitions en relation dans T_n. Chaque transition silencieuse dans T_n a pour probabilité 1 afin de préserver l'évolution au sein des modes originaux. Soit T l'automate obtenu. On dit que T est une approximation linéaire probabiliste de H.

La preuve de la validité de cette méthode est donnée dans l'annexe C.

Exemple 4.5.13. *Une approximation linéaire du thermostat probabiliste est obtenue en modifiant l'automate hybride de la figure 4.5 et est illustrée dans la figure 4.8. Pour chaque transition discrète d'un mode v vers v', la valeur de la probabilité a été reportée sur les transitions des modes (v, i) et (v, j). C'est le cas de la transition allumer à partir du mode $(OFF, 1)$ qui a une probabilité 0.9 vers $(ON, 1)$ et 0.1 vers $(DOWN, 1)$ comme dans l'automate la figure 4.5.1.*

La preuve de la construction conduit à un PHA valide : comme conséquence, on obtient le théorème suivant.

Theorème 4.5.14. *Toute approximation linéaire T d'un PHA H le simule : c'est-à-dire, $H \preceq T$. Une partition d'un PHA lui est bisimilaire.*

Comme corollaire de ce théorème, si l'approximation linéaire T satisfait une propriété $\forall$-PBTL, alors H la satisfait. Par contre, lorsque T ne satisfait pas une propriété $\forall$-PBTL,

on ne pourrait pas conclure directement que H ne la satisfait pas. On pourrait, comme dans le cas de l'approximation linéaire non-probabiliste, partitionner davantage en vue d'obtenir une approximation plus précise.

La preuve du théorème 4.5.14 est donnée dans l'annexe C.

Dans ce chapitre, nous avons présenté les systèmes hybrides probabilistes comme systèmes dynamiques, et nous avons défini les PHA comme formalisme permettant de les représenter. Nous avons également présenté la sémantique des PHA qui permet de dériver de tout PHA un LMP qui ne peut être utilisé à des fins pratiques. Enfin, nous avons présenté des techniques pour la vérification des PHA.

De par leur sémantique, un LMP et un PHA semblent avoir des points communs, puisqu'ils ont tous les deux un lien avec les systèmes de transitions probabilistes à espace d'états continu. Toutefois, il reste qu'ils n'appartiennent pas à la même classe de systèmes étant donné qu'un PHA, contrairement à un LMP, supporte le temps-continu. Le chapitre suivant sera consacré à la comparaison entre LMP et PHA.

Deuxième partie

Comparaison entre LMP et PHS

Chapitre 5

Processus de Markov Étiquetés VS Systèmes Hybrides Probabilistes

5.1 Introduction

Comme nous l'avons vu dans les chapitres précédents, les LMP et les PHA modélisent tous les deux des systèmes probabilistes à espace d'états continu, mais de manière différente. Notre but, dans ce chapitre, est de faire ressortir leurs différences et limites sur la base de deux études de cas, le système d'un avion dans les airs et celui d'un bateau en mouvement dans un plan, que nous décrivons dans la section suivante. Nous essayerons de modéliser ces systèmes, dans chacun des formalismes, en vue de trouver celui qui répond le mieux aux exigences dans chacun des cas.

5.2 Études de cas

Dans cette section, nous présentons deux systèmes que nous analyserons en utilisant le formalisme des LMP et des PHA.

5.2.1 Un avion

Le premier système qui est soumis à notre analyse est un avion qui se déplace dans les airs. Voici, plus en détail, la description de ce système.

Description du système

Initialement, l'avion est au sol et une fois démarré, il s'elève dans les airs tant que l'altitude maximum autorisée n'est pas atteinte. La vitesse d'élevation est comprise entre 20 et 25 mètres par unité de temps. À tout moment, le pilote peut faire pivoter l'avion à droite ou gauche ; au cours de la rotation, l'avion, sous certaines forces de frottement (résistance de l'air, etc.), peut perdre de l'altitude suivant une loi de probabilité. Si l'avion atteint l'altitude zéro après une rotation, nous supposerons qu'il s'est écrasé. De plus, si le pilote tente de le pivoter lorsque son altitude est en dessous de la valeur minimale autorisée H_{min}, alors l'avion s'écrasera. Nous prévoyons qu'après une rotation, il y a de forte chance que l'avion se retrouve à une hauteur plus proche que loin de sa précedente hauteur. Ainsi, la probabilité que d'une altitude s, l'avion perde environ 20 mètres de sa hauteur doit être supérieure à sa probabilité de perdre 50 mètres. Supposons que les valeurs possibles d'altitude sont dans $E := [0, H_{\max}] \cup \{\mathit{Ecrasé}\}$ où $H_{\max}$ est l'altitude maximum. Nous désignerons par $p(s, [a, b])$ la probabilité que d'une altitude s, l'altitude de l'avion atteigne soit dans $[a, b]$ après une rotation. Nous choisissons une distribution exponentielle sur $s - x$ où s est l'altitude de l'avion lorsque le pilote effectue une rotation afin de garantir que les intervalles de valeurs d'altitude qui sont proches de s aient une probabilité plus grande. Nous définissons p comme suit, où s est en unité de hauteur (pas nécessairement en mètres) :

- si $0 \leq s < H_{\min}$, $p(s, \{\mathit{Ecrasé}\}) := 1$
- si $H_{\min} \leq s \leq H_{\max}$, $p(s, \cdot)$ est l'unique extension de mesure de probabilité définie par l'ensemble des fonctions suivantes : (le cas $a \leq s < b$ peut être déduit)
  - $p(s, [a, b]) := \begin{cases} \int_a^b e^{-(s-x)} dx & \text{si } 0 < a \leq b \leq s \\ p(s, [a, s]) & \text{si } s < b \quad \text{ainsi 0 si } s < a \end{cases}$
  - $p(s, \{\mathit{Ecrasé}\}) := \int_{-\infty}^{0} e^{-(s-x)} dx = e^{-s}$

Remarquons que le choix de cette mesure de probabilité pour modéliser l'avion ne tient pas compte de l'instabilité qui pourrait être engendrée par une rotation : nous supposons que l'altitude perdue est mesurée une fois que l'avion est stabilisé. Nous supposons aussi que l'avion ne peut que perdre en altitude (c'est-à-dire si $s \leq a \leq b$, $p(s, [a, b]) = 0$). Cette simplification n'aura aucune influence dans notre modélisation.

Dans nos modélisations du système, nous n'aurons besoin que de valeurs numériques pour exprimer des propriétés telles que "à une certaine altitude supérieure à $H_{\min}$, la probabilité que l'avion perde 100 mètres ou plus après une rotation est entre 25% et 50%" : dans ces cas, l'unité de la hauteur sera 100 mètres et $H_{\min} := 5$.

Analyse du système

Le système de l'avion, tel que nous l'avons décrit, a plusieurs comportements dans son fonctionnement. On peut observer un comportement continu caractérisé par le déplacement de l'avion dans les airs, et un comportement discret qui correspond à la rotation déclenchée par le pilote. À chacun de ces comportements, sont liées des caractéristiques qui s'appliquent au système. Par exemple, lorsqu'il est dans les airs, l'avion s'élève à une vitesse comprise entre 20 et 25 mètres par unité de temps. Il existe donc une incertitude dans le choix d'un prochain état puisque les transitions de temps se font de manière non-déterministe suivant des distributions non définies. Par contre, lorsque le pilote fait pivoter l'avion, l'incertitude réside dans la perte d'altitude, et a pour valeur $p(s, [a, b])$ où s est l'altitude courante de l'avion et $[a, b]$ l'intervalle cible. Deux actions sont possibles dès lors : τ pour les transitions de délai et *tourner* pour représenter l'action par laquelle le pilote fait pivoter l'avion.

Dans ce qui suit, nous modélisons l'avion comme un LMP et un PHA.

L'avion comme un LMP

Dans cette partie, nous tenterons de modéliser l'avion à l'aide d'un LMP.

L'avion étant un système à temps continu, cette modèlisation ne pourrait se faire fidèlement. Nous aurons besoin de discrétiser le temps. Soit t une unité de temps. Nous utilisons t comme délai de base pour nos transitions. L'ensemble des états sera composé des altitudes possibles de l'avion ajoutées à l'état de l'écrasement : $[0, H_{\max}] \cup \{\mathit{Ecrasé}\}$, avec comme état initial 0. La σ-algebra est celle générée par l'union de Σ et de l'état supplémentaire : nous la désignons par $\Sigma + \{\mathit{Ecrasé}\}$. La fonction de propositions atomiques n'est pas pertinente dans ce cas : on pourrait choisir d'étiqueter tous les états dans lesquels l'avion ne s'est pas écrasé par *Air*, ou ceux qui sont entre $H_{\min}$ et $H_{\max}$ comme *Sécuritaire* et les plus petits états comme *Bas* : ce choix dépend des propriétés qu'on a besoin de vérifier. L'ensemble des actions contiendra *tourner*, mais également une action τ qui représentera les changements qui ont lieu à chaque unité de temps t.

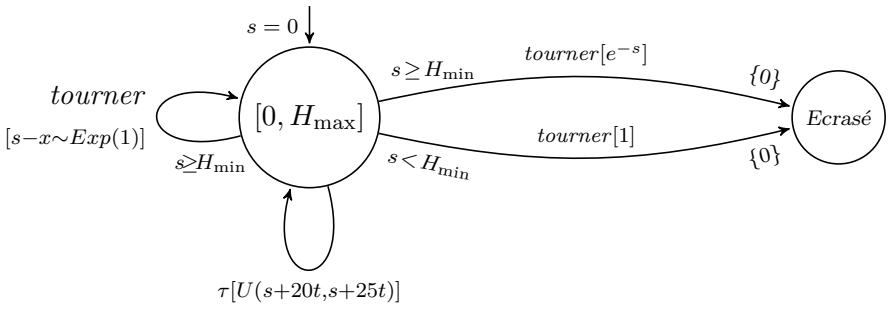

FIGURE 5.1 – L'avion comme un LMP

Nous définissons le LMP sur Act := $\{tourner, \tau\}$ comme :

$$G^t_{\text{LMP}} = ([0, H_{\max}] \cup \{Ecrasé\}, \Sigma + \{Ecrasé\}, 0, \{\mu_{tourner}, \mu_\tau\}, L).$$

Deux sortes de transitions probabilistes discrètes sont définies. Les transitions *tourner* sont définies exactement comme dans la spécification.

- Si $0 \leq s < H_{\min}$, $\mu_{tourner}(s, \{Ecrasé\}) = 1$.
- Si $H_{\min} \leq s \leq H_{\max}$, $\mu_{tourner}(s, \cdot)$ est l'unique extension de mesure de probabilité de la forme :

$$\mu_{tourner}(s, [a, b]) := \begin{cases} \int_a^b e^{-(s-x)} dx & \text{si } 0 < a \leq b < s \quad \text{i.e. } s - x \sim \text{Exp}(1) \\ 0 & \text{si } s \leq a \leq b \end{cases}$$

$$\mu_{tourner}(s, \{Ecrasé\}) := \int_{-\infty}^0 e^{-(s-x)} dx = e^{-s}.$$

Les transitions de temps ont lieu pendant que l'avion s'élève vers son altitude maximum. D'après la spécification, cela se produit à une vitesse comprise entre 20 and 25. Ainsi, si l'altitude est $0 \leq s \leq H_{\max} - 25$, l'avion devrait se retrouver à une altitude comprise entre $s + 20t$ et $s + 25t$ après que l'unité de temps t soit écoulé. Cependant, cette incertitude n'est pas quantifiée, ce qui fait qu'il est impossible de la modéliser telle quelle dans la structure du LMP. Dès lors, nous devons choisir comment nous elle aura lieu. Nous choisissons, une fois de plus, la distribution uniforme. Ainsi,

$$\text{si } 0 \leq s \leq H_{\max} - 25t, \quad \mu_\tau(s, \cdot) := U(s + 20t, s + 25t)$$

Pour s entre $H_{\max} - 25t$ et $H_{\max}$, μ_τ est définie de manière évidente, dans le but de ne pas avoir à dépasser la hauteur maximale $H_{\max}$. Le LMP obtenu G^t_{LMP} est représenté à la figure 5.1. Un exemple de propriété qui peut être vérifiée sur G^t_{LMP} est **P1** :

«si $s > H_{\min}$, la probabilité que l'avion perde 100 mètres ou plus après une rotation est entre 25% et 50%».

La distribution associée à l'action *tourner* si $s > H_{\min}$ donne une probabilité de $p(s, [0, s-100] \cup \{Ecrasé\}) = e^{-1} - e^{-s} + e^{-s} = e^{-1}$. Dès lors, nous concluons que G^t_{LMP} satisfait la propriété.

Toutefois, toute propriété qui fait appel à une précision supérieure à celle que nous avons choisi avec t ne pourrait être vérifiée de manière fiable simplement parce que dans le système temps-discret, les informations entre deux unités de temps ne sont pas prises en compte. Comme exemple de propriété, considérons **P2** :

«si l'avion se trouve au dessus de $H_{\min}$, il lui prend moins d'une seconde pour gagner 10 mètres d'altitude».

La vérification de cette propriété requiert les informations sur les états atteignables en moins d'une seconde. Si l'unité de temps définie pour la discrétisation est par exemple supérieure ou égale à une seconde, alors soit cette propriété ne pourrait être vérifiée, soit elle sera réfutée. En effet, les informations sur l'état du système ne seraient pas représentées dans le LMP si bien que l'on serait dans l'incapacité d'affirmer si le système satisfait ou non la propriété. Une solution serait de déterminer l'unité de temps adéquate pouvant permettre d'obtenir un LMP suffisamment précis pour que la vérification soit faite. Il s'agit toutefois d'une technique ardue qui pourrait faire l'objet d'autres travaux de recherche.

L'avion comme un PHA

Nous désirons à présent modéliser l'avion comme un PHA.

Puisque l'avion est un système à temps-continu, les systèmes hybrides conviennent parfaitement pour modéliser les transitions de délai. D'un autre coté, les PHA ne supportent que des distributions de probabilité discrètes tandis que les distributions de probabilité qui s'appliquent à l'avion sont continues. Dès lors, le système ne pourrait être fidèlement modélisé une fois de plus. Dans le PHA, on n'aura qu'une seule variable, l'altitude, et donc des valeurs réelles comme valuations. Il y aura deux modes : *Air* représentant la situation dans laquelle l'avion peut voler et *Ecrasé* où il peut s'écraser . La seule action discrète est *tourner*. Nous définissons maintenant les autres paramètres du PHA, que nous appelons :

$$G_{\text{PHA}} = (\{x\}, \{Air, Crashed\}, \text{init}, \{rotate\}, \text{inv}, \mathit{flow}, \mathit{prob}, \langle \mathit{pre}_{Air}, \mathit{pre}_{Crashed} \rangle)$$

- $\text{init}(Air) = \{0\}$.
- $\text{inv}(Air) = [0, H_{\max}]$ et $\text{inv}(Ecrasé) = \{0\}$.

- $\mathit{flow}(\mathit{Air}) = [0.2, 0.25]$ et $\mathit{flow}(\mathit{Ecrasé}) = \{0\}$.
- $\mathit{prob}(\mathit{Ecrasé}, \mathit{tourner}) = \emptyset$, puisqu'aucune action ne peut être exécutée.
- $\mathit{prob}(\mathit{Air}, \mathit{tourner})$ est séparée en deux distributions :
  - μ_1 avec $\mathit{pre}(\mu_1) = [0, H_{\min}]$: pour tout $s \in [0, H_{\min}]$, $\mu_1(\mathit{tourner}, \{0\}, \{s\}) = 1$, autrement dit, l'avion se retrouve dans le mode *Ecrasé* et s est réinitialisé à 0.
  - μ_2 avec $\mathit{pre}(\mu_2) =]H_{\min}, H_{\max}]$: pour tout $s \in \mathit{pre}_{\mathit{Air}}(\mu_2)$, nous discutons de la définition de $\mu_2((\mathit{Air}, \mathsf{post}, Y))$ ci-dessous.

Pour définir la distribution μ_2, nous avons un choix difficile à faire. En effet, pour être fidèle au modèle, nous devons spécifier une précondition pour toute valeur de x, et nous aurions besoin que μ_2 soit une distribution non discrète. Ceci n'est pas possible puisque nous ne pouvons avoir qu'un nombre fini de distributions discrètes. Une solution est de :

1. partitionner l'intervalle $]H_{\min}, H_{\max}]$ en autant d'intervalles $I_1, I_2, \ldots, I_n$ que possible ;
2. partitionner le mode *Air* en fonction de ces intervalles la transition de I_j vers I_k, notée $\mu_j(\mathit{Air}, I_k, \emptyset)$;
3. choisir une valeur qui représenterait l'ensemble $\{p(i, I_k) \mid i \in I_j\}$ (où p est la probabilité définie pour l'avion dans la section 5.2.1).

Pour simplifier, nous choisissons, ici, de ne pas partitionner le mode pour les préconditions, mais nous considérons plutôt séparement les deux postconditions $I_1 := [0, H_{\min}]$ et $I_2 :=]H_{\min}, H_{\max}]$; nous devons choisir les valeurs qui représentent les ensembles $P_1 := \{p(x, [0, H_{\min}]) \mid x \in [H_{\min}, H_{\max}]\}$ et $P_2 := \{p(x,]H_{\min}, H_{\max}]) \mid x \in [H_{\min}, H_{\max}]\}$. Nous donnons, ici, les valeurs des paramètres q et q' pour simplifier, tout en ne faisant aucune hypothèse (on aurait pu choisir le plus petit ou la moyenne des valeurs des ensembles P_1 et P_2). De manière analogique, nous donnons la valeur q'' à $\mu_2(\mathit{Ecrasé}, \{0\}, \{x\})$. La figure 5.2 est une représentation du PHA obtenu.

Une fois de plus, on est loin de la distribution exponentielle qui est utilisée dans la spécification du sytème. Toutefois, ce modèle peut être utilisé pour vérifier quelques propriétés. Discutons de la manière dont on pourrait vérifier les propriétés **P1** et **P2** sur G_{PHA}. Rappelons que **P1** est la propriété suivante :

«si $s > H_{\min}$, la probabilité que l'avion perde 100 mètres ou plus après une rotation est entre 25% et 50%».

Cette propriété pourrait être vérifiée dans la version LMP de l'avion. Dans ce modèle, par contre, tout ce que l'on peut affirmer sur la probabilité d'atteindre $[0, s - 100] \cup \{\mathit{Ecrasé}\}$ est qu'elle est supérieure à la probabilité d'atteindre $[0, H_{\min}] \cup \{\mathit{Ecrasé}\} =$

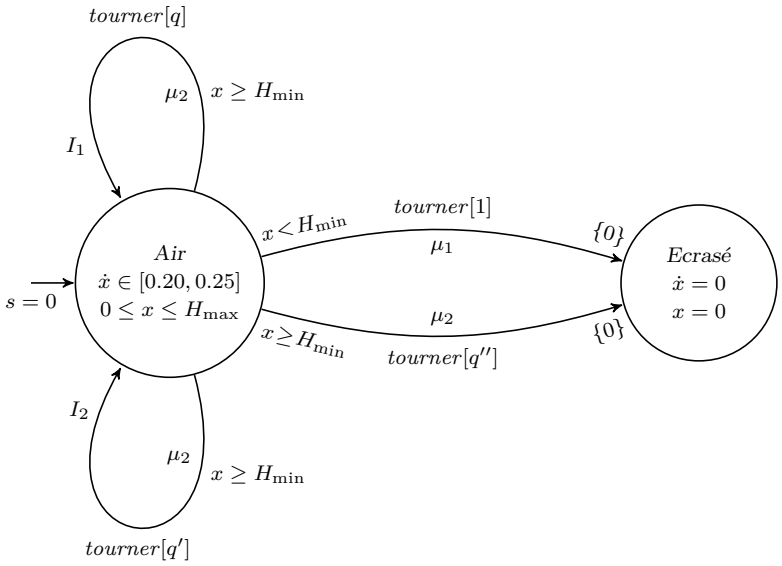

FIGURE 5.2 – L'avion comme un PHA : ici $I_1 = [0, H_{\min}]$ et $I_2 =]H_{\min}, H_{\max}]$

$q' + q''$. Ainsi, nous serons en mesure de réfuter la formule si $q' + q''$ est supérieure à 50%, sinon nous ne pouvons rien affirmer. Quant à la propriété **P2** :

«si l'avion se trouve au dessus de $H_{\min}$, il lui prend moins d'une seconde pour gagner 10 mètres d'altitude»,

elle peut être vérifiée sur G_{PHA} grâce à l'algorithme proposé par Sproston pour les propriétés de la logique $\mathcal{T}_0$ définie dans la section 4.3.

5.2.2 Un bateau

Il s'agit d'un exemple, tiré de [20], que nous avons modifié afin de l'utiliser dans notre contexte. Dans ce qui suit, nous donnons une description du système.

Description

Le système, d'une façon sommaire, simule le mouvement, dans un plan euclidien, d'un objet soumis par endroit à des forces de frottement. Ce plan est divisé en $n \times m$ régions, chacune caractérisée par :

– une étiquette $r \in \{0, ..., 7\} \cup \{A, B\}$ qui est associée à chaque région du plan. Dans une région, l'étiquette détermine la vitesse de l'objet. La valeur de cette vitesse est v_d :

$$v_d = (v_{d_{x(r)}}, v_{d_{x(r)}}) \;=\; (\sin(r \times \pi/4), \cos(r \times \pi/4)) \; avec \; r = 0, ..., 7$$

– une équation qui définit pour tout objet de vitesse v, une évolution sous la forme $\dot{v} = A(v - v_d)$ où $A \in \mathbb{R}^2$ est une matrice dont la diagonale est pourvue de valeurs strictement négatives. v_x et v_y représentent respectivement les vitesses horizontale et verticale à l'intérieur d'une région.

Le but de l'analyse sera de nous assurer que l'objet, dans son déplacement sur la grille, passera à un moment donné par l'une des régions étiquetées **A**, et ne passera jamais par les régions **B**. On suppose que la longueur et la largeur d'une cellule sont 1, et que le plus bas coin gauche de la grille est l'origine. La position de l'objet sur la grille est donnée par les variables x et y, et en tout moment, la vitesse v de l'objet aura pour coordonnées (v_x, v_y), où v_x et v_y représentent respectivement la vitesse horizontale et verticale de l'objet. La vitesse initiale de l'objet est v_0. Sa vitesse sur la grille ne devrait, toutefois, pas dépasser une certaine valeur que nous noterons V_{max}. Lorsque, dans une région, la vitesse v_d est supérieure à celle de l'objet, celui-ci est emporté par le courant. Dans le cas contraire, le courant a peu d'influence sur l'objet.

L'équation d'évolution assure que l'objet, une fois dans une région, voit sa vitesse évoluer vers v_d. De plus, nous supposons qu'aux extrémités de deux régions, l'objet n'est soumis à aucune force et par conséquent v_d a la valeur $(0, 0)^T$.

Dans ce qui suit, on considère que l'objet est un bateau, et que la grille est un ensemble de régions, chacune étant caractérisée par un courant donné. On suppose que les régions portant la même étiquette ont des propriétés communes.

La figure 5.3a montre une grille de dimension 3×3 où chacune des étiquettes i permet de calculer la vitesse v_d au sein de la région portant l'étiquette r. La figure 5.3b, quant à elle, montre deux trajectoires d'objets, avec $A = \begin{pmatrix} -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix}$. Chacune d'elles satisfait la propriété que l'objet passera par la région **A** et que la région **B** ne serait jamais atteinte.

Un état du bateau est, par conséquent, représenté par le vecteur à 4 dimensions $s = (x, y, v_x, v_y)^T$. Pour l'exemple 5.3, son évolution sur la grille est donnée par l'équation différentielle suivante :

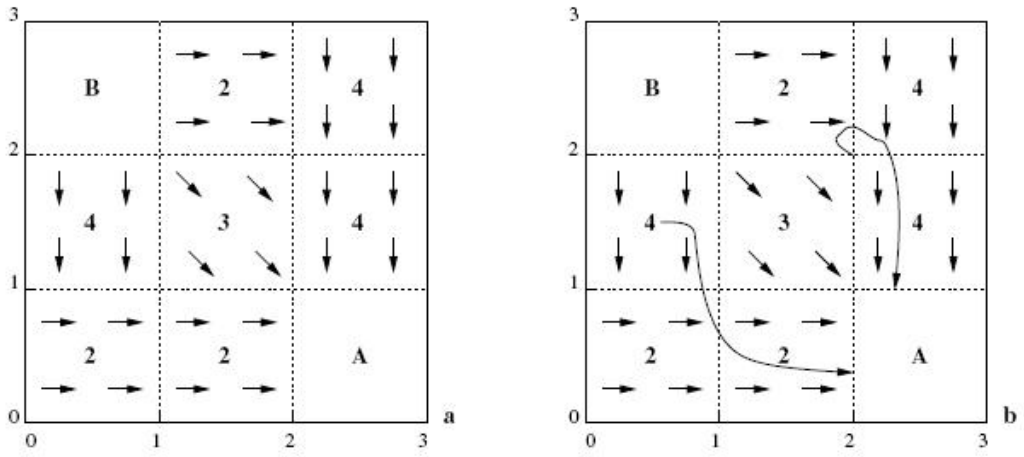

FIGURE 5.3 – a : Grille de dimension 3×3. b : Deux trajectoires d'objets se déplaçant sur la grille

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v_x} \\ \dot{v_y} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1.2 & 0.1 \\ 0 & 0 & 0.1 & -1.2 \end{pmatrix} \begin{pmatrix} x \\ y \\ v_x \\ v_y \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix} \begin{pmatrix} v_{d_{x(i)}} \\ v_{d_{x(i)}} \end{pmatrix}.$$

Nous modifions le système en y introduisant une incertitude de la manière suivante. Le capitaine peut faire descendre les voiles et ainsi faire diminuer la vitesse du bateau afin qu'elle puisse atteindre plus rapidement la vitesse désirée v_d. Pour empêcher que cette action soit exécutée de manière infinie, nous supposons qu'elle ne peut être exécutée que si la vitesse du bateau est supérieure à une certaine valeur $V_0 \leq V_{max}$ qu'on désignera par *vitesse permise*. Soient $v = (v_x, v_y)$ et $v' = (v'_x, v'_y)$ les vitesses du bateau respectivement avant et après la descente des voiles. Pour chacune des vitesses (horizontale et verticale) v_k, $k \in \{x, y\}$, on désigne par $P(v_k, [a, b])$, la probabilité que la vitesse, après la descente des voiles, soit entre $[a, b]$, où $0 \leq a \leq b \leq v_k \leq V_{max}$. Ainsi, $P(v_x, [a, b])$ est la probabilité que la vitesse horizontale soit dans $[a, b]$ et $P(v_y, [a, b])$, la probabilité que la vitesse verticale soit dans $[a, b]$. Pour simplifier, nous avons choisi d'exprimer la valeur de la probabilité en fonction des vitesses. En effet, nous aurions pu l'exprimer en fonction de la position du bateau, mais cela aurait nécessité, à chaque étape, des calculs supplémentaires (calcul des fonctions v_x et v_y, et intégration pour dériver x et y) qui ne font pas l'objet de notre illustration, et ne changent rien à nos concepts. Pour chacune de ces vitesses, la probabilité est donnée par la formule :

$$P(v_k, [a, b]) = \frac{(b-a)b}{{v_k}^2} \text{ et } P(0, [a, b]) = 0.$$

Avec les restrictions sur a et b, on s'assure que les vitesses du bateau après la descente des voiles soient inférieures à celles avant. Autrement dit, nous voulons que le bateau n'ait aucune chance de gagner en vitesse suite à l'action du capitaine. De plus, nous fixons la vitesse permise V_0 à v_d pour toutes les régions. Ainsi, on pourra toujours diminuer la vitesse du bateau tant que celle-ci est supérieure à v_d.

En résumé, une instance de ce modèle est caractérisée par :
- la position initiale (x_0, y_0) de l'objet sur la grille
- la vitesse initiale $v_0 = (v_{x_0}, v_{y_0})$ de l'objet
- la fonction de transition probabiliste P
- la matrice A de l'équation différentielle caractérisant l'évolution de la vitesse v
- la grille pouvant être représentée comme une matrice de dimension $m \times n$ dont les éléments appartiennent à l'ensemble $\{0, ..., 7\} \cup \{\mathbf{A}, \mathbf{B}\}$. Ainsi, la grille de la figure 5.3 serait représentée par $\begin{pmatrix} B & 2 & 4 \\ 4 & 3 & 4 \\ 2 & 2 & A \end{pmatrix}$.

Analyse du système

Étant donné que les paramètres physiques du système qui sont représentés par les coordonnées et la vitesse sur la grille évoluent dans le temps, le bateau est un système continu. Les paramètres physiques sont :
- les variables x et y représentant respectivement l'abscisse et l'ordonnée du bateau sur la grille, telles que $x \in [0, n]$ et $y \in [0, m]$,
- les variables v_x et v_y qui donnent les coordonnées du vecteur vitesse au point (x, y) telles que $v_x = \dot{x}$ et $v_y = \dot{y}$.

Une instance de la grille est une matrice d'étiquettes r caractérisant chacune des régions. Une région de la grille est représentée par :
- sa position (i, j) sur la grille où $i \in \{0, 1, ..., n\}$ et $j \in \{0, 1, ..., m\}$; la position servira comme identifiant des régions dans la suite ;
- l'étiquette $r \in \{0, 1, ..., 7\} \cup \{\mathbf{A}, \mathbf{B}\}$ qui permet de définir la valeur de la vitesse v_d de la région ;
- les invariants sur les coordonnées du bateau dans la région : $x \in [i, i + 1]$ et $y \in [j, j + 1]$;
- l'évolution du vecteur $(x, y, v_x, v_y)^T$ donnée par le vecteur $(\dot{x}, \dot{y}, \dot{v}_x, \dot{v}_y)$; cette évolution se fait suivant l'équation $\dot{v} = A(v - v_d)$ ou A est une matrice et v_d la

vitesse à atteindre dans cette région.

D'après la description, on distingue une seule action externe sur le système entier. Il s'agit de l'action par laquelle l'on fait descendre les voiles. Puisque cette action a pour effet de ralentir le bateau, on la nommera *ralentir*. Si on suppose que la vitesse du bateau est supérieure à V_0, alors une transition étiquetée *ralentir* aura lieu. Les probabilités que les vitesses v'_x et v'_y soient dans un intervalle $[a, b]$ après la descente des voiles sont $P(v_x, [a, b])$ et $P(v_y, [a, b])$.

Considérons l'instance $\begin{pmatrix} B & 2 & 4 \\ 4 & 3 & 4 \\ 2 & 2 & A \end{pmatrix}$ du modèle représentée sur la figure 5.3 avec $A = \begin{pmatrix} -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix}$. Dans le but de décrire le changement de régions lors de l'évolution du système, nous allons définir un système de transitions. Ainsi, on distingue les ensembles l_i d'états suivants :

1. $l_{\mathbf{B}}$ l'ensemble des états appartenant à la région qui doit être évitée ;
2. l_2 correspond à l'ensemble des états de régions ayant pour étiquette $r = 2$ ou $v_d = (1, 0)$; par conséquent, le déplacement du bateau se fait dans ces états suivant le système d'équations différentielles ci-après :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \\ -1.2v_x + 0.1v_y + 1.2 \\ 0.1v_x - 1.2v_y - 0.1 \end{pmatrix};$$

3. l_3 correspond à $r = 3$ ou $v_d = (+0.707, -0.707)$; dans ces états, le mouvement du bateau se fait suivant le système d'équations différentielles :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \\ -1.2v_x + 0.1v_y + 0.919 \\ 0.1v_x - 1.2v_y - 0.919 \end{pmatrix};$$

4. l_4 correspond à $r = 4$ ou $v_d = (0, -1)$; dans ces états, le mouvement du bateau peut être décrit avec le système d'équations différentielles :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \\ -1.2v_x + 0.1v_y + 0.1 \\ 0.1v_x - 1.2v_y - 1.2 \end{pmatrix};$$

5. $l_{\mathbf{A}}$ correspond aux états de la région **A** qui doit être atteinte.

Les invariants dans chacun de ces ensembles sont définis sur les valeurs des variables x et y. Ainsi, dans l'exemple de la figure 5.3, le bateau restera dans l'ensemble de régions l_2 s'il est dans l'une des régions $(0,0)$, $(1,0)$ ou $R^{(1,2)}$. Par conséquent, le bateau restera dans l_2 si :

- $x \in [0,2]$ et $y \in [0,1]$ ou
- $x \in [1,2]$ et $y \in [2,3]$.

Au niveau des transitions, une transition *ralentir* est exécutée pour la variable v_k si $v_k \geq v_{d_k}$. Puisque l'on considère qu'aucune action n'a lieu sur les frontières des régions, toute transition *ralentir* aura pour cible un état de la région courante.

À présent, supposons que le bateau se trouve à l'état initial s_0, ce qui veut dire que l'on se trouve initialement en l_3 et les équations différentielles prédominant le système sont celles décrites au point 4 ci-haut. Grâce à la méthode de transformation de Laplace, nous pouvons résoudre le système d'équations différentielles 4 avec l'outil Maple pour obtenir les solutions suivantes pour x et y :

$$\begin{aligned} x &= -0.5e^{-1.1t} + 0.346e^{-1.3t} + 0.654 \\ y &= -0.5e^{-1.1t} - 0.346e^{-1.3t} - t + 2.35 \\ v_x &= 0,5e^{-1.1t} - 0.4498e^{-1.3t} \\ v_y &= 0,5e^{-1.1t} + 0.4498e^{-1.3t} - 1. \end{aligned}$$

Ces polynômes permettront de trouver la position et la vitesse du bateau en tout instant t.

L'exemple du bateau étant présenté, nous nous attellerons, dans les sections suivantes, à modéliser ce système dans le formalisme des LMP et des PHA.

Le bateau comme un LMP

Nous modélisons d'abord le bateau comme un LMP.

Comme dans le cas de l'avion, nous ne pouvons obtenir un modèle fidèle du bateau avec un LMP. En effet, comme l'avion, le bateau est un système à temps continu puisque les variables x, y, v_x et v_y évoluent continuellement dans le temps. Nous aurons donc recours à une unité de temps pour pouvoir obtenir une représentation approximative du bateau avec un LMP. Choisissons u comme délai de base entre deux transitions. L'ensemble des états sera bien sûr l'ensemble $S := \{(x, y, v_x, v_y)^T \mid x \in [0,n], y \in [0,m]\}$ avec comme état initial $s_0 := (0.5, 1.5, 0.1, 0)^T$. La σ-algèbre est celle engendrée par

l'union de S et des autres états : nous la dénotons Σ. Pour la fonction de propositions atomiques, on choisit d'associer un état à la région à laquelle il appartient. Ainsi, l'état $(1.434, 0, \cdot, \cdot)^T$ aura comme étiquette $(1, 0)$. Quant aux états appartenant aux régions à atteindre ou à ne pas atteindre, on leur affectera respectivement les étiquettes **A** et **B** respectivement. L'ensemble des actions contiendra *ralentir*, mais aussi l'action silencieuse τ qui représentera les changements internes qui ont lieu à chaque unité de temps u. Nous définissons le LMP sur $\mathcal{A} := \{ralentir, \tau\}$ comme :

$$G^u_{\text{LMP}} = (S, \Sigma, s_0, \{\mu_{ralentir}, \mu_\tau\}, L).$$

Deux différentes sortes de transitions discrètes probabilistes sont définies sur G^u_{LMP}. Les transitions étiquetées *ralentir* sont définies conformément à la spécification. Pour tout $k \in \{x, y\}$, $\mu_{ralentir}$ est l'unique mesure de probabilité définie comme suit :

- Si $0 \leq a \leq v_k \leq V_{max}, \mu_{ralentir}(v_k, [a, b]) = \frac{(b-a)b}{{v_k}^2}$
- Si $v_k = 0$, $\mu_{ralentir}(0, [a, b]) = 0$.

Les transitions de temps ont lieu lorsque le bateau est en mouvement à l'intérieur d'une région. D'après la spécification, le déplacement se fait suivant un rythme qui s'applique aux quatre paramètres physiques et qui est propre à chaque région. Ainsi, dans une région dont la vitesse cible est v_d, la variable vitesse v évolue suivant l'équation $\dot{v} = A(v - v_d)$ où $A \in \mathbb{R}^2$. Pour représenter cette évolution, nous choisissons de définir une unité de temps fixe. L'information sur l'état du système sera prise à chaque intervalle de temps et sera déterminée suivant les solutions des systèmes d'équations différentielles caractérisant chaque région. Ainsi, si $f : \mathbb{R}_+ \to \mathbb{R}$ est la solution caractérisant les abscisses au sein d'une région, alors on pourra calculer les abscisses $f(u), f(2u), ...$ après u unités de temps. Il en est de même pour les ordonnées et les vitesses.

Plus concrètement, considérons l'instance de la grille présentée à la figure 5.3 avec $A = \begin{pmatrix} -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix}$. Tel que présenté dans 5.2.2, de l'état initial s_0, l'évolution dans les régions portant l'indice 4, c'est-à-dire $(0, 1)$, $(1, 2)$, $(2, 2)$, se fait suivant :

$$\begin{aligned} x &= -0.5e^{-1.1t} + 0.346e^{-1.3t} + 0.654 \\ y &= -0.5e^{-1.1t} - 0.346e^{-1.3t} - t + 2.35 \\ v_x &= 0,5e^{-1.1t} - 0.4498e^{-1.3t} \\ v_y &= 0,5e^{-1.1t} + 0.4498e^{-1.3t} - 1. \end{aligned}$$

Simulons une évolution du bateau sur la grille. Supposons que le bateau se trouve

dans la région $(0,1)$ à l'état $s_0 = (x_0, y_0, \cdot, \cdot)$ avec $0 \leq x_0 \leq 1$ et $1 \leq y_0 \leq 2$ à l'instant t. On a donc $x_0 = x(t := t_0)$ et $y_0 = y(t := t_0)$. Après une unité de temps (de durée u) dans la région $(0,1)$, le bateau se retrouvera dans l'état $s_1 = (x_1, y_1, v_x^u, v_y^u)$ avec $0 \leq x_1 \leq 1$ et $1 \leq y_1 \leq 2$ tels que $x_1 = x(t := t_0 + u)$ et $y_1 = y(t := t_0 + u)$. De même, après $2u$ unités de temps, il se retrouvera en $s_2 = (x_2, y_2, v_x^{2u}, v_y^{2u})$ avec $0 \leq x_2 \leq 1$ et $1 \leq y_2 \leq 2$ tels que $x_2 = x(t := t_0 + 2u)$ et $y_2 = x(t := t_0 + 2u)$. On affecte donc la valeur 1 à la probabilité de se rendre de s_1 vers s_2. L'état du système étant pris à chaque unité de temps, les informations sur l'état du bateau entre u et $2u$ ne sont, par conséquent, pas connues.

Discutons, à présent, du comportement probabiliste du système. Puisque les informations sur les états parcourus entre u et $2u$ ne sont pas représentées, on est donc pas en mesure de modéliser fidèlement les transitions vers ces états. Nous sommes tenus de faire un choix de modélisation qui, même s'il n'est pas conforme au comportement du système original, nous permettra d'assurer la continuité entre les transitions. Désignons par Q l'ensemble des états visités par le système entre les instants u et $2u$, et soit $s = (x, y, v_x, v_y)$ l'état courant du système. Soit V_k^Q l'ensemble des valeurs de vitesses v_k des états de Q avec $k \in \{x, y\}$. Plusieurs alternatives s'offrent pour modéliser une transition de s vers vers Q. Entre autres, on peut choisir de :

1. donner une probabilité nulle aux transitions de s vers Q avec la distribution de probabilité μ_k^τ définie comme suit :

   $$\mu_k^\tau(v_k, \cdot) := 0 \quad \text{pour } k \in \{x, y\}.$$

   Dans ce cas, les transitions de s vers les états de Q sont ignorées dans notre LMP, ce qui est loin de traduire la réalité puisqu'il existe des états de Q par lequel le bateau passe entre u et $2u$.

2. donner 1 à la probabilité de se rendre de s vers Q telle que

   $$\text{pour } k \in \{x, y\}, \mu_k^\tau(v_k, V_k^Q) := 1.$$

   Ici, on considère uniquement la transition vers l'ensemble Q, et on suppose que toute transition vers un état de l'ensemble a la probabilité 0. De ce fait, on sous-entend que l'ensemble Q est atteignable à partir de s.

3. définir μ_k comme une loi de probabilité Uniforme sur V_k^Q :

   $$\text{si } \forall v_k^i \in V_k^Q,\ v_k^i \leq V_{max},\ \text{alors } \mu_k^\tau(v_k, \cdot) := U(v_{1k}, v_{2k}) \quad \text{pour } k \in \{x, y\}.$$

Aucune des options présentées ne permet de représenter fidèlement l'incertitude liée aux transitions de s vers les états Q tel que spécifié dans le système original. Pour la suite

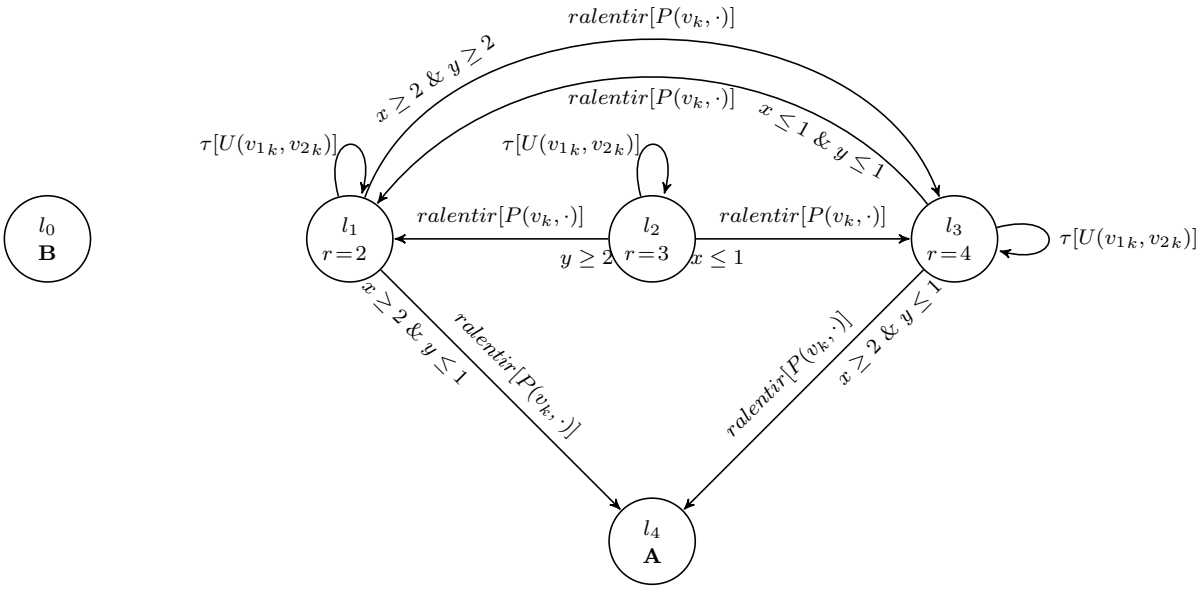

FIGURE 5.4 – Le bateau comme un LMP : l'instance de la figure 5.3

de notre tentative de modélisation, nous utiliserons, par pur choix, la loi Uniforme (option 3) pour représenter cette incertitude. La figure 5.4 est une représentation graphique du modèle G^u_{LMP} obtenu avec l'instance de la figure 5.3.

Nombreuses sont les propriétés pouvant être vérifiées sur G^u_{LMP}. Ainsi, on pourrait s'intéresser à déterminer les valeurs de vitesse à partir desquelles le capitaine peut faire descendre les voiles sans passer en dessous d'un certain seuil de vitesse, ou celles à partir desquelles le bateau a une certaine chance de passer dans une autre région avec la vitesse désirée de celle-ci. Soit le problème suivant :

P3 : «Supposons que le capitaine, dans le souci de maintenir une stabilité dans la navigation, désire passer de la région $(0,1)$ vers la région $(0,0)$ avec une vitesse v_x comprise entre $I := [v_{dx} - 0.1, v_{dx}]$. On cherche à savoir les valeurs de v_x à partir desquelles il peut faire descendre les voiles et avoir plus de 80% de chance de réussir son coup».

Soit K l'ensemble des v_x recherché. La détermination de K passe par le calcul de $[\![\langle ralentir \rangle_{0.8} I]\!]$. Puisque que la région $(0,0)$ a pour indice $r = 2$, sa vitesse v_d est le couple $(1,0)$. D'après la sémantique de $\mathcal{L}_0$ présentée à la section 3.2, on a :

$$\begin{aligned} s \vDash \langle ralentir \rangle_{0.8} I &\Leftrightarrow \mu_{ralentir}(v_x, I) > 0.8 \\ &\Leftrightarrow \mu_{ralentir}(v_x, [0.9, 1]) > 0.8 \\ &\Leftrightarrow \frac{(1-0.9) \times 0.9}{{v_x}^2} > 0.8 \end{aligned}$$

On obtient donc :

$$s \vDash \langle ralentir \rangle_{0.8} I \Leftrightarrow v_x \in]-0.335, 0, 335[$$

Par conséquent, l'ensemble des valeurs de v_x qui sont solutions de **P3** est $K =]-0.335, 0, 335[$.

Considérons à présent le problème **P4** suivant :

P4 : «On cherche à savoir si dans plus de 50% des cas, le bateau peut atteindre, en moins de deux secondes, la région $R^{(1,0)}$ à partir de l'état initial».

La vérification d'une telle propriété requiert une connaissance des états du système entre 0 et 2 secondes. Puisque notre modèle ne nous permet pas d'avoir ces informations, on ne pourrait donc pas procéder convenablement à la vérification d'une telle formule. D'une manière générale, toute propriété qui fait appel à une occurrence plus grande que celle que nous avons choisie avec u ne pourra être vérifiée fidèlement. Dans certains cas, on pourra, toutefois, choisir u suffisamment petit pour minimiser les délais entre les transitions et disposer d'un maximum d'informations sur les états de G^u_{LMP}. Même si cette dernière solution pourrait être efficace dans bien des cas, il n'en reste pas moins que son application implique une granularité plus fine de l'espace d'états et une explosion du nombre de transitions.

Le bateau comme un PHA

Dans cette partie, nous essayons de modéliser le bateau en utilisant le formalisme des PHA.

D'après l'analyse faite dans la section 5.2.2, le système est temps-continu, et par conséquent, les systèmes hybrides sont donc parfaits pour modéliser les transitions de délai qui ont lieu à l'intérieur des régions.

Toutefois, les PHA ne supportent que des distributions discrètes de probabilité tandis que les distributions de probabilité qui accompagnent les transitions discrètes du bateau sont continues. Par conséquent, le système du bateau ne peut être modélisé fidèlement. Néanmoins, des tentatives de modélisation peuvent être faites avec des distributions discrètes que nous choisirons.

Considérons l'instance du bateau de la figure 5.3. Une tentative de modélisation conduirait au PHA, noté G_{PHA}, défini comme suit :

$$G_{\text{PHA}} = (X, V, \text{init}, \text{Act}, \text{inv}, \mathit{flow}, \mathit{prob}, \langle \mathit{pre}_{v \in V} \rangle)$$

- X contient les paramètres physiques, donc $X := \{x, y, v_x, v_y\}$;
- V, l'ensemble des locations, contient les coordonnées des régions. Par conséquent, on a $V := \{(i,j) | i \in \{0, ..., n\}, j \in \{0, ..., m\}\} \cup \{\mathbf{A}, \mathbf{B}\}$;
- init est telle que pour $v = (0,1)$, $\text{init}(v) := \{\mathsf{a}_0\}$ avec $\mathsf{a}_0 := (0.5, 1.5, 0.1, 0)^T$, et pour tous les autres $v \in V$, on a $\text{init}(v) = \emptyset$;
- inv est telle que :
  - $\text{inv}((0,0)) = \{(x, y, v_x, v_y) \in \mathbb{R}^4 \mid (x,y) \in [0,1] \times [0,1]\}$
  - $\text{inv}((0,1)) = \{(x, y, v_x, v_y) \in \mathbb{R}^4 \mid (x,y) \in [0,1] \times [1,2]\}$
  - ...
  - $\text{inv}(\mathbf{A}) = \{(x, y, v_x, v_y) \in \mathbb{R}^4 \mid (x,y) \in [2,3] \times [0,1]\}$
  - $\text{inv}(\mathbf{B}) = \{(x, y, v_x, v_y) \in \mathbb{R}^4 \mid (x,y) \in [0,1] \times [2,3]\}$;
- pour chacun des modes v, $\mathit{flow}(v, \mathsf{a})$ est l'ensemble des solutions des systèmes d'équations différentielles caractérisant v. Ainsi, pour $v = (1,1)$, l'évolution $\mathit{flow}(v, \cdot)$ est déterminée par le système d'équations suivant :

$$\begin{aligned} x &= -0.5e^{-1.1t} + 0.346e^{-1.3t} + 0.654 \\ y &= -0.5e^{-1.1t} - 0.346e^{-1.3t} - t + 2.35 \\ v_x &= 0,5e^{-1.1t} - 0.4498e^{-1.3t} \\ v_y &= 0,5e^{-1.1t} + 0.4498e^{-1.3t} - 1; \end{aligned}$$

- pour tout mode $v \in V$, $\mathit{prob}(v, \mathit{ralentir}) = \{\mu_k\}$ tel que $\mathit{pre}(\mu_k)$ représente l'ensemble des préconditions de la transition *ralentir* définies pour la variable v_k. Puisque cette transition ne peut être exécutée que si v_k est supérieure à v_{dk}, alors $\mathit{pre}(\mu_k) := [v_{dk}, V_{max}]$. Pour tout $s \in \mathit{pre}(\mu_k)$ avec $\text{supp}(\mu_k) \neq \emptyset$, la définition de $\mu_k(v', \mathsf{post}, Y)$ est discutée ci-dessous.

En définissant μ_k, nous avons un choix à faire. En effet, afin d'être fidèle au modèle, d'une part, nous devons associer une précondition à toute valeur de x et d'autre part, nous avons besoin que μ_k soit une distribution continue. Dans les deux cas, on fait face

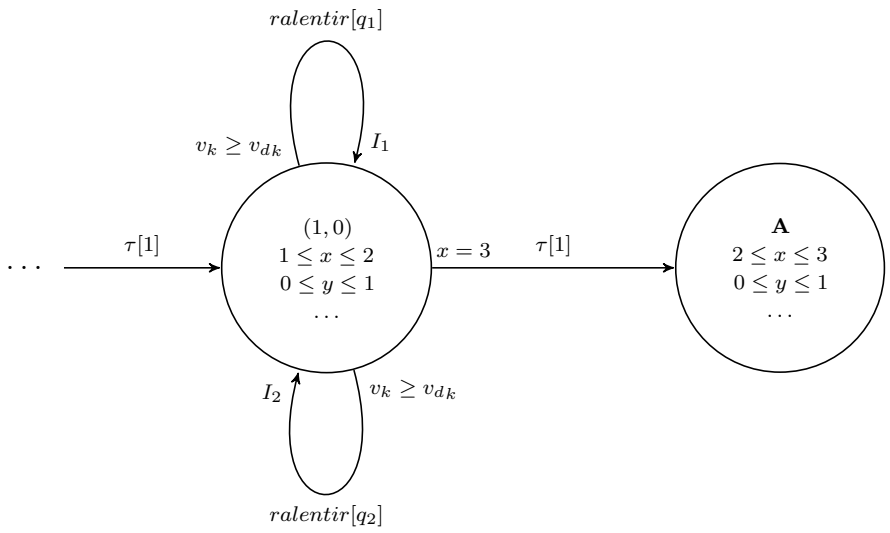

FIGURE 5.5 – Représentation partielle de G_{PHA} : régions $(1, 0)$ et **A**

à une impossibilité, puisque nous ne pouvons définir qu'un nombre fini de distributions discrètes. Une solution consisterait à :

1. partitionner l'intervalle $pre(\mu_k)$ en le plus d'intervalles $I_1, I_2, \ldots, I_p$ possibles ;
2. partitionner les modes v' pour lesquels on a $\mu_k(v', \mathsf{post}, Y) > 0$ suivant les intervalles $I_1, I_2, \ldots, I_p$ tel que la transition de I_i vers I_j, notée $\mu_k(v', I_j, \emptyset)$, ait une valeur qui représente l'ensemble $\{P(c, I_j) \mid c \in I_i\}$ (où p est la probabilité définie pour le bateau dans la description).

Pour simplifier, nous choisissons de ne pas partitionner le mode pour les préconditions, en préférant plutôt partitionner les postconditions en des intervalles distincts $I_1, I_2, \ldots, I_p$ tels que $I_i \cap I_j = \emptyset$. À chaque I_j, nous choisissons d'associer la valeur du paramètre q_j pour représenter $\{P(c, I_j) \mid c \in I_i\}$ qui est l'ensemble des probabilités vers I_j à partir d'un état de I_i, tout en ne faisant aucune hypothèse (on pourrait choisir le plus petit ou la moyenne des valeurs des ensembles). On obtient donc un ensemble fini de valeurs $q_1, q_2, \ldots, q_p$ pour les transitions vers les ensembles $I_1, I_2, \ldots, I_p$. La figure 5.5 est une représentation des régions $(1, 0)$ et **A** du PHA obtenu.

Il est évident qu'avec cette approche, on est encore loin de la distribution exprimée par la fonction de transition utilisée dans la spécification du système. Pour preuve, utilisons le PHA G_{PHA} pour vérifier les propriétés **P1** et **P2** sur le modèle du bateau. Rappelons **P1** :

P3 : «Supposons que le capitaine, dans le souci de maintenir une stabilité dans la navigation, désire passer de la région $R^{(0,1)}$ vers la région $R^{(0,0)}$ avec une vitesse

v_x comprise entre $I := [v_{dx} - 0.1, v_{dx}]$. On cherche à savoir les valeurs de v_x à partir desquelles il peut faire descendre les voiles et avoir plus de 80% de chance de réussir son coup».

Puisque v_x est la seule variable concernée dans **P3**, nous ferons fi des autres variables dans ce qui suit, en faisant l'hypothèse que les postconditions dans les transitions sont incluses dans les invariants des modes cibles. Soit K l'ensemble des états recherchés. La résolution de ce problème ne pourrait se faire de façon efficace avec G_{PHA}. En effet, la résolution de **P3** consiste en la détermination des états $s := (v', \mathsf{a})$ tels que $s \in pre(\mu_{ralentir})$ et $\mu_{ralentir}(v', I, \{v_x\}) \geq 0.8$ où I contient un ensemble des nouvelles valeurs a' de v_x dans v'. Or, l'ensemble des postconditions I étant continu, son partitionnement en $I_1, I_2, \ldots, I_p$ conduirait à la génération des valeurs de probabilité $q_1, q_2, \ldots, q_p$. Deux cas de figures se présentent.

- S'il existe $q_j := \mu_{ralentir}(v', I_j, \{v_x\}) \geq 0.8$ alors nous pouvons conclure que s appartient à l'ensemble recherché.
- Si par contre, il existe $q_j := \mu_{ralentir}(v', I_j, \{v_x\}) < 0.8$, alors on ne peut dire avec certitude que s n'appartient pas à K. Il se peut que I_j soit un sous ensemble d'un ensemble I' pour lequel $q' := \mu_{ralentir}(v', I', \{v_x\}) \geq 0.8$.
- S'il existe q_i et q_j pour I_i et I_j respectivement tels que $q_i + q_j >= 0.8$, on ne pourrait pas non plus conclure sans réserve que s appartient à K. Il se peut que pour des valeurs c' appartenant à I_i (ou I_j), la valeur $P(\mathsf{c}, \{\mathsf{c}'\})$ ne soit pas supérieure à 0.8.

À cause de cette ambiguité, le modèle G_{PHA} ne garantit pas dans tous les cas l'exactitude de la solution pour **P3**.

Considérons maintenant la propriété :

P4 : «On cherche à savoir si dans plus de 50% des cas, le bateau peut atteindre, en moins de deux secondes, la région $R^{(1,0)}$ à partir de l'état initial».

Contrairement à **P3**, la propriété **P4** peut être aisément vérifiée sur G_{PHA}. En considérant la logique $\mathcal{T}_0$, vérifier **P4** sur G_{PHA} revient à vérifier si $s_0 := ((0,1), \mathsf{a}_0)$ satisfait la propriété $[\top \forall \mathcal{U}(1,0)]_{\geq 0.5}$. Des algorithmes pour vérifier de telles propriétés sur les PHA à distributions discrètes existent [42, 23], et des outils ont été récemment développés [21, 23]. Au nombre de ces ceux-ci, on distingue SSMT (de l'anglais Stochastic Satisfiability Modulo Theory) développé par le groupe de recherche de Fränzle et al. [21] qui prend en entrée un PHA et le transforme en une formule de la logique nommée SSMT. Cette formule SSMT est ensuite utilisée pour déduire les résultats de vérification.

Les méthodes d'approximation sur les PHA que nous avons présentées à la section 4.5.2 peuvent également être utilisées pour effectuer cette vérification.

5.2.3 Conclusion

À travers les études de cas présentées dans ce chapitre, nous avons fait ressortir les caractéristiques propres aux LMP et aux PHA, ainsi que leurs limites respectives.

La première étude porte sur un avion en déplacement qui perd une certaine altitude, déterminée par une fonction de probabilité, lorsque le pilote le fait pivoter. Quant à la seconde, elle concerne un bateau en mouvement sur une grille et qui, suivant une fonction de probabilité, perd de la vitesse lorsque le capitaine décide de ralentir.

Pour chacun de ces systèmes, nous avons procédé à la modélisation comme un LMP puis comme un PHA. Nous pouvons conclure qu'aucun des modèles obtenus ne peut permettre d'être fidèle à ces systèmes du point de vue de leurs spécifications. Ainsi, il existe des systèmes ne pouvant être modélisés ni par les LMP, et ni par les PHA. De plus, notre analyse a permis de remarquer qu'il existe principalement deux différences entre les LMP et les PHA ; celles-ci sont liées à la nature des distributions et à la manière dont le temps est représenté dans ces modèles (discret ou continu).

Il est important de noter que l'échec de la modélisation fidèle de la distribution exponentielle sur nos études de cas n'est pas la conséquence des limites des PHA rectangulaires. Même avec un PHA plus général, la *structure* de la définition n'est pas adéquate pour modéliser des distributions. Cela nous conduit à souligner une observation importante sur les LMP : la continuité de l'espace d'états dans les LMP dépend fondamentalement de la nature continue de la distribution. Se limiter à la distribution discrète sur tout espace conduit à l'obtention d'un processus discret, et dès lors, à la définition de relations de bisimulation.

Compte tenu de ces divergences, nous proposons, dans le chapitre suivant, une généralisation qui reflète les caractéristiques des LMP et des PHA.

Chapitre 6

Les LMP Hybrides (HLMP)

6.1 Introduction

L'étude comparative faite dans le chapitre 5 a permis de relever des insuffisances dans le comportement des LMP et des PHA. Deux points principaux sont ressortis : d'une part, il existe un système à espace d'états continu pouvant être modélisé par un PHA et ne pouvant pas l'être par aucun LMP (et vice versa), et d'autre part il existe une classe de systèmes à espace d'états continus pour laquelle aucune modèlisation fidèle n'est possible ni avec les LMP ni avec les PHA.

Ce chapitre est consacré à l'introduction d'une nouvelle plateforme de modèle pour les systèmes probabilistes à espace d'états continu et à temps continu qui regroupe les caractéristiques des LMP et des PHA : les Processus de Markov Étiquetés Hybrides. Avec cette nouvelle classe de modèle, nous serons en mesure de générer un langage suffisamment expressif pour englober celui généré par les LMP de Desharnais [12], et celui généré par les PHA de Sproston [40].

Nous commencerons d'abord par définir le formalisme des Processus de Markov Étiquetés Hybrides. Ensuite, nous discuterons du langage qu'ils génèrent et des types de systèmes qu'ils supportent. Enfin, nous nous concentrerons sur leur vérification formelle.

6.2 Définition

Comme mentionné dans le chapitre 5, les LMP et les PHA ont des particularités qui leur permettent de mieux modéliser les comportements de certains systèmes que d'autres. Dans le but de disposer d'une plateforme suffisamment générale pour modéliser des systèmes complexes tels que l'avion et le bateau, l'idée est de créer une nouvelle classe de modèles qui non seulement aura les caractéristiques des LMP et des PHA mais aussi répondra aux insuffisances de ceux-ci. Nous désignons cette nouvelle plateforme de modèle par Processus de Markov Étiquetés Hybrides (HLMP, de l'anglais Hybrid Labelled Markov Processes). Ces derniers devront donc combiner les comportements des LMP et des PHA, autrement dit, un HLMP aura des transitions de délai à valeurs réelles et des distributions continues.

Discutons maintenant des fondements de la théorie des HLMP. À cette fin, rappelons la définition suivante.

Définition 6.2.1. *Soit $\mathcal{B}$ la σ-algèbre Borel générée par $[0,1]$. Nous écrivons $\mathcal{B}_n$ pour désigner la σ-algèbre Borel sur $\mathbb{R}^n$.*

Afin de construire une plateforme de modèles valide et répondant à nos attentes, nous devons considérer les particularités de chacune des plateformes à combiner. Nous utiliserons comme point de départ les PHA. Nous choisissons de garder l'ensemble des locations comme fini au lieu de définir un espace de mesure arbitraire. Ceci est purement une restriction théorique qui ne doit pas empêcher des applications futures et qui aide à maintenir le modèle souple.

La principale modification des PHA se fera dans la façon dont nous intégrerons les distributions continues. Soient H un PHA, v un mode et a une action de H. Pour modifier H, il ne suffit pas seulement de changer la définition de *prob* pour que $prob(v,a)$ puisse contenir des distributions continues. À première vue, cela constitue un moyen pour supporter la continuité au niveau des distributions de H. Cependant, du fait que cet ensemble dépendrait uniquement de v et des préconditions qui seraient plus tard attachées à ces distributions, nous serions obligé de forcer un nombre continu de valuations, et ainsi nombre continu d'états, à se comporter comme v. En faisant cela, tous ces états pourraient finalement s'avérer bisimilaires dans la sémantique sous-jacente et ainsi nous reviendrions à notre point de départ : une distribution discrète. En particulier, remarquons que la distribution exponentielle dans l'exemple de l'avion ne pourrait pas être modélisée, puisqu'elle dépend de la valeur courante de l'altitude et non du fait que l'avion vole ou s'est écrasé.

Par conséquent, il y a deux caractéristiques du PHA qui doivent être modifiées pour pouvoir insérer des distributions continues au modèle :

1. la fonction donnant l'ensemble des distributions *prob* ;
2. la fonction déterminant les préconditions *pre*.

La modification de *pre* se fera comme suit : ce qui est encodé dans la fonction de préconditions deviendra un paramètre de la fonction de transition probabiliste. Ainsi, les mesures de probabilité *prob* qui seront retournées pourront utiliser ce paramètre.

Nous avons également besoin de définir une autre sorte d'ensemble de mesures de probabilité comme suit :

Définition 6.2.2. *Étant donnés V un ensemble fini et (S, Σ) un espace mesurable, l'ensemble des distributions de probabilité sur $(S, V \times \Sigma)$, noté $\overline{\mathsf{Sub}}(S, V \times \Sigma)$, est tel que :*

$$\overline{\mathsf{Sub}}(S, V \times \Sigma) := \{\mu \in \mathrm{Dist}(S, V \times \Lambda) \mid \Lambda \textit{ est une } \sigma\textit{-algèbre } \Lambda \subseteq \Sigma\}.$$

Nous définissons un processus de Markov étiqueté hybride comme suit :

Définition 6.2.3. *Un processus de Markov étiqueté hybride (HLMP pour l'anglais Hybrid Labelled Markov Process) est une structure $C = (X, V, \mathrm{init}, A, \mathrm{inv}, \mathit{flow}, \mathit{prob}, \mathit{Label})$ définie comme suit :*

- *X, V, A sont définis comme dans la définition 4.2.1 ;*
- *$\mathrm{init} : V \to \mathcal{P}(\mathbb{R}^n)$ définit un ensemble initial*[1] *;*
- *$\mathrm{inv} : V \to \mathcal{P}(\mathbb{R}^n)$ définit les invariants pour les variables de chaque location ;*
- *$\mathit{flow} : V \times \mathbb{R}^n \to \mathcal{P}(\mathbb{R}^n)$ est une condition d'évolution ;*
- *$\mathit{prob} : V \times \mathbb{R}^n \times A \to \mathcal{P}_{\mathit{fin}}(\overline{\mathsf{Sub}}(V \times \mathbb{R}^n, V \times \mathcal{B}_n))$ représente des transitions probabilistes ;*
- *$\mathit{Label} : V \times \mathbb{R}^n \to \mathcal{P}(\mathrm{AP})$ est une fonction mesurable utilisée pour décrire des états.*

La définition d'un HLMP ressemble beaucoup à celle des PHA telle que nous les avons définis dans la définition 4.2.1. En effet, les notions de modes, d'ensembles de variables et d'actions sont maintenues. En dehors de ces notions, les fonctions init, inv et *flow* sont également représentées et sont définies comme l'a fait Sproston [40], à la différence que les ensembles d'arrivée ne sont plus simplement des sous-ensembles de $\mathbb{R}^n$ mais plutôt de $\mathcal{B}_n$. Il faut dire que ce dernier aspect, même s'il n'apporte pas une grande

1. Nous l'écrivons de cette manière pour simplifier, mais l'image est plutôt $\mathcal{B}_n$.

amélioration dans la pratique, permet de maintenir un certain niveau d'abstraction et de généralité dans la définition de nos modèles.

Par contre, nous pouvons remarquer d'une part que la fonction *pre* n'est pas définie et d'autre part, que la fonction *prob* est définie, mais différemment par rapport à la définition 4.2.1 d'un automate hybride probabiliste. Autrement dit, la définition des transitions probabilistes dans un HLMP n'est pas la même que dans un PHA.

Discutons à présent des modifications apportées aux fonctions *pre* et *prob*. Commençons par *prob*. Observons d'abord que dans les PHA, une distribution μ exprime du non-déterminisme s'il existe un ensemble post de plus d'une valuation tel que $\mu(v, \mathsf{post}, Y) > 0$. Cette situation correspond aux cas où les variables dans Y peuvent être réinitialisées à plus d'une valuations dans le mode cible. Ainsi, si on a $\mathsf{post} := \{-ln(2), 1, 2, \sqrt{3}\}$ et $Y := \{x\}$ tels que $\mu(v, \mathsf{post}, Y) := 0.9$ alors μ est non-déterministe. Par conséquent, le fait de ne pas disposer de valeurs de probabilités spécifiques pour les sous-ensembles de post donne lieu à des transitions non-déterministes. De plus, il faut noter que les ensembles de ce genre peuvent se chevaucher donnant ainsi naissance à de nombreuses copies. Par conséquent, nous devons tenir compte des nombreuses copies produites lors de la définition de la sémantique en vue d'être fidèle au modèle et de respecter les contraintes liées aux mesures de probabilité. Dépendamment de la nature des distributions, nous proposons un moyen pour résoudre le problème lié au non-déterminisme des transitions.

- Lorsque les distributions sont discrètes, comme dans le cas des PHA, ceci peut être fait assez aisément puisque les valeurs de probabilité sont complètement indépendantes des valuations dans post.
- Avec des distributions continues, si les sous-ensembles post sont des ensembles mesurables, alors la probabilité de chacun d'eux, de même que celle de l'union et intersection sera connue. Dès lors, si pour un état s et une action a, la σ-algèbre Σ sur laquelle $\mu \in \mathit{prob}(s, a)$ est définie est $\mathcal{P}(V) \times \mathcal{B}_n$, alors il n'y a aucun non-déterminisme dans le comportement de μ : tout ensemble mesurable de cette σ-algèbre peut être partitionné en des sous-ensembles (à moins qu'il soit l'ensemble singleton) dont les valeurs de probabilité seront déterminées grâce à μ. D'un autre côté, lorsque μ est définie sur une plus petite σ-algèbre $\Lambda \subseteq \mathcal{P}(V) \times \mathcal{B}_n$, il peut exister des ensembles *minimaux* E, appelés *atomes* de σ-algèbre, pour lesquels $\mu(E)$ est défini, sans qu'aucun ensemble inclus dans E n'ait de valeur par μ : il s'agit du même non-déterminisme que l'on rencontre dans les ensembles post des PHA.

6.3 La sémantique

Nous définissons, à présent, la sémantique des HLMP. Pour cela, nous devons redéfinir la fonction de transition de la sémantique des PHA donnée dans la définition 4.4.1. Soit $\mu \in \mathit{prob}(v, \mathsf{a}, a)$ une distribution définie sur la σ-algèbre $V \times \Lambda$ et soit $\{\mathsf{E}_i\}_{i\in I}$ la famille d'atomes Λ telle que $\mu(v, \mathsf{E}_i) > 0$ pour $v \in V$. Cette famille peut être dans le meilleur des cas dénombrable (car $\mu(V \times \mathbb{R}^n) \leq 1$). On définit $T(\mu)$ comme l'ensemble de toutes les combinaisons $\langle \mathsf{b} \rangle := \langle \mathsf{b}_1, \mathsf{b}_2, \ldots, \mathsf{b}_m \rangle$, avec $\mathsf{b}_i \in \mathsf{E}_i$. Nous avons également besoin d'ajouter ces valuations à la σ-algèbre Λ avec ces valuations qui ne sont pas, comme le cas des singletons, inclus ; ainsi, on définit $\Lambda^+ := \Lambda \cup \sigma(\{\{\mathsf{b}\} \mid \exists i \in I.\mathsf{b} \in \mathsf{E}_i\})$ [2].

Définition 6.3.1. *Étant donné un HLMP $H = (X, V, \mathrm{init}, A, \mathrm{inv}, \mathit{flow}, \mathit{prob})$, on dérive le système probabiliste concurrent infini associé*

$$(Q, \mathrm{Init}, A \cup \mathbb{R}, \mathit{Steps}) \text{ comme suit :}$$

- $Q := \{(v, \mathsf{a}) \in V \times \mathbb{R}^n \mid \mathsf{a} \in \mathrm{inv}(v)\}$ *est l'ensemble des états admissibles ;*
- $\mathrm{Init} = \{(v_0, \mathsf{a}_0) \in Q \mid \mathsf{a}_0 \in \mathrm{init}(v_0)\}$ *;*
- $\mathit{Steps}(v, \mathsf{a}) := \mathit{Cts}(v, \mathsf{a}) \cup \mathit{Dis}(v, \mathsf{a})$*, pour l'état $(v, \mathsf{a}) \in Q$, et :*
  - *Cts est définie comme dans la définition 4.4.1 ;*
  - $\mathit{Dis}(v, \mathsf{a}) \subseteq A \times \mathit{Sub}(Q)$ *contient pour chaque $\mu \in \mathit{prob}(v, \mathsf{a}, a)$ et $\langle \mathsf{b} \rangle \in \mathit{Cible}(\mu)$, toutes les paires $(a, \mu_{\langle \mathsf{b} \rangle})$ telles que $\mu_{\langle \mathsf{b} \rangle}$ est définie comme, pour $\mathsf{C} \in \Lambda^+$:*

$$\mu_{\langle \mathsf{b} \rangle}(v', \mathsf{C}) := \mu(v', \mathsf{C} \setminus \cup_i \mathsf{E}_i) + \sum_{\substack{i \in I \\ \mathsf{b}_i \in \mathsf{C} \cap \mathrm{inv}(v')}} \mu(v', \mathsf{E}_i).$$

On peut facilement remarquer que les HLMP englobent les PHA. La sémantique des HLMP le prouve bien d'ailleurs. En plus d'englober les PHA, les HLMP offrent une architecture pouvant supporter des transitions plus complexes. Il n'existe donc aucun système qui pourrait être modélisé par les PHA et qui ne pourrait l'être par un HLMP. Cependant, comme nous l'avons mentionné, ces deux types de modèles de systèmes ont de nombreuses similarités, surtout sur le plan architectural. De ce fait, certaines notions s'appliquant aux PHA, s'appliquent également aux HLMP. Ainsi, selon les équations différentielles caractérisant l'évolution des variables, on pourra distinguer les mêmes types de sous-ensembles comme dans le cas des PHA [42]. On distinguera entre autres la classe des HLMP linéaires définie comme suit.

Définition 6.3.2. *Un HLMP $M = (X, V, \mathrm{init}, A, \mathrm{inv}, \mathit{flow}, \mathit{prob}, \mathit{Label})$ est dit linéaire si les fonctions* init, inv *et flow peuvent être représentées sous la forme de $\dot{x} = k$ pour tous les paramètres $x \in X$, avec $k \in \mathbb{R}$.*

2. si W est une famille d'ensembles, $\sigma(W)$ est la plus petite σ-algèbre contenant W.

Dans le cas particulier où la fonction *flow* serait de la forme $\dot{x} = 1$ pour tous les $x \in X$, alors M serait un automate temporisé probabiliste dont les distributions de probabilité seraient définies comme dans la définition 6.2.3, bénéficiant par conséquent des mêmes avantages que les HMLP à ce niveau. De plus, on pourra distinguer la la classe des HLMP rectangulaires.

Définition 6.3.3. *Un HLMP* $M = (X, V, \text{init}, A, \text{inv}, \textit{flow}, \textit{prob}, \textit{Label})$ *est dit* rectangulaire *si les conditions initiales, les invariants, les conditions sur l'évolution, les préconditions et les postconditions sont des sous-ensembles de* $R(X)$.

D'une manière générale, les sous-ensembles des HLMP bénéficient de la granularité et de la généralité de la fonction *prob* donnant dès lors naissance à de nouvelles définitions de ces sous-ensembles. De par ce fait, les sous-ensembles des HMLP se différencient des autres sur les mêmes aspects que les HLMP se différencient des PHA, c'est-à-dire qu'ils les englobent et sont plus expressifs. Toutefois, même si les HLMP sont aussi des modèles de systèmes à espace d'états continu dont les transitions probabilistes peuvent supporter des distributions continues, ils n'incluent pas les LMP. Ceci est simplement dû au choix de l'ensemble fini de locations. En effet, le fait que l'ensemble de modes, V, soit fini, réduit la taille de l'espace d'états puisqu'on est obligé de s'en tenir à un ensemble fini de «groupe d'états». Certes, nos modèles auront des espaces d'états continus, mais ayant des particularités communes qui permettront de les regrouper en un nombre fini de catégories d'états. Par conséquent, il existe des systèmes qui peuvent être modélisés par les LMP mais par aucun HLMP. Comme exemple de tels systèmes, considérons le système théorique dont l'espace d'états est $\mathbb{R}$, et supposons que chacun des états $x \in \mathbb{R}$, le système a un comportement particulier. On suppose, enfin, qu'à tout moment, le système peut passer d'un état à un autre suivant une certaine distribution de probabilité μ. Dans ce système, on peut remarquer que l'ensemble des modes a la même taille que $\mathbb{R}$. Un tel système pourrait être modélisé par un LMP, mais pas par un HLMP puisque l'ensemble des situations identifiées aux modes y est infini.

Malgré la différence liée à la taille de l'espace d'états, il faut dire que les HLMP partagent avec les LMP la même continuité intrinsèque selon les distributions, ce qui fait que tant que l'ensemble des modes, sera fini, tout système pouvant être modélisé par un LMP pourra être modélisé par un HLMP.

Dans tous les cas, nous sommes certain qu'il existe des modèles pouvant être modélisés par des HLMP pour lesquels on n'aurait une représentation formelle fidèle ni avec les LMP ni avec les PHA. Pour illustrer ce fait, nous modélisons, dans la section suivante, le système de l'avion en utilisant le formalisme des HLMP.

6.4 Modélisation de l'avion

Dans cette section, nous montrons que la nouvelle classe de HLMP permet de modéliser exactement l'exemple de l'avion de la section 5.2.1.

Tel que nous l'avions mentionné, l'exemple décrit un système temps-continu pouvant exécuter des transitions discrètes. De plus, les distributions de probabilité qui accompagnent les transitions discrètes sont continues comme dans les LMP. Avec le formalisme des HLMP, nous sommes donc en mesure de représenter chacun de ces comportements. Le modèle désiré est

$$G_{\text{HLMP}} = (\{x\}, \{\mathit{Air}, \mathit{Écrasé}\}, \text{init}, \{\mathit{tourner}\}, \text{inv}, \mathit{flow}, \mathit{prob}) \text{ avec}$$

- X, V , init, A, inv, *flow* sont les mêmes que dans G_{PHA} ;
- *prob* est défini comme suit :
  - si $x \leq H_{\min}$, alors $\mathit{prob}(\mathit{Air}, x, \mathit{tourner})$ contient seulement la distribution qui donne la probabilité 1 à *Écrasé* et 0 ailleurs ;
  - si $H_{\min} \leq x \leq H_{\max}$, $\mathit{prob}(\mathit{Air}, x, \mathit{tourner}) = \{\mu_x\}$ où $\mu_x(\mathit{Écrasé}, 0) = e^{-x}$ et $\mu_x(\mathit{Air}, E) = \mu_{\mathit{tourner}}(E)$ si $E \subseteq \mathbb{R}$, où μ_{tourner} provient de la modélisation avec un LMP de l'avion (Section 5.2.1).

Cette représentation combine les deux représentations de l'avion : celle en tant que PHA à la section 5.2.1 et celle en tant que LMP de la section 5.2.1. La figure 6.1 est une représentation graphique du modèle obtenu. Dans ce modèle, les propriétés **P1** et **P2** peuvent être vérifiées avec précision. De plus, la propriété suivante peut être aussi vérifiée, tandis qu'elle n'aurait pas pu l'être avec précision ni sur les modèles LMP ni sur les les modèles PHA :

P5 «si une rotation se produit entre 20 et 30 secondes, la probabilité que l'avion ne s'écrase pas est supérieure à 1/4».

Intuitivement, le calcul donne l'ensemble non vide $\{(\mathit{Air}, x) \mid x \in [5, 7.5]\}$ d'états atteignables à partir de l'état initial, et par conséquent **P5** est satisfaite par G_{HLMP}.

6.5 Vérification formelle des HLMP

Dans cette section, nous discutons de la manière dont la vérification des HLMP peut être faite. Nous proposons deux approches. Rappelons que seules quelques sous-classes de systèmes hybrides (probabilistes ou non) peuvent être vérifiées.

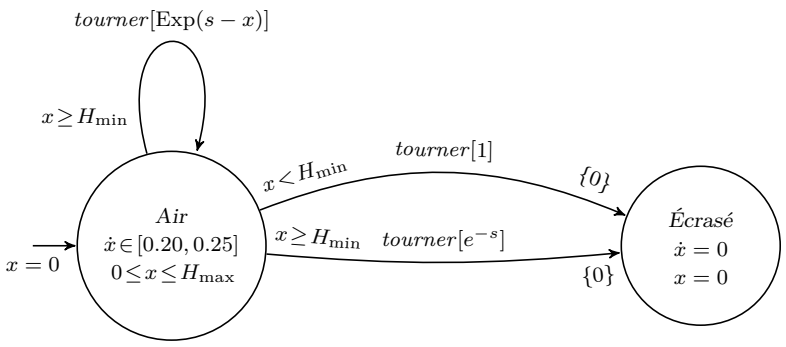

FIGURE 6.1 – Le modèle HLMP de l'avion

L'approche la plus simple que nous proposons pour vérifier un HLMP est sous implémentation et consiste à construire un LMP à partir du HLMP en discrétisant le temps de la même manière que nous l'avions fait lorsque nous modélisions l'avion avec les LMP, dans la section 5.2.1. Toutefois, cette méthode, comme dans le cas de l'avion, ne résulte pas en un modèle fidèle à l'original, et dès lors la vérification faite avec ce modèle pourrait ne pas donner le résultat attendu. Certaines méthodes peuvent néanmoins être employées en vue de contourner les limitations liées à la discrétisation du temps. Nous en proposons quelques-unes ici.

La première méthode consiste en le choix d'une «bonne» unité de discrétisation. En effet, si nous choisissons une unité de temps «raisonnable» ou si nous connaissons à l'avance les propriétés à vérifier, alors nous serons en mesure de déduire l'unité de temps t qui permettra d'obtenir une approximation du HLMP donné en fonction des propriétés à vérifier. Plus précisement, pour chaque état s, une seule transition de délai sera active, une transition τ de délai t, comme dans l'exemple de l'avion. Le résultat, dans ce cas, est un processus plus précis que celui obtenu avec la sémantique des HLMP où une transition de délai d est définie pour toute valeur $d \in \mathbb{R}$. La partie probabiliste des transitions ne requiert aucune transformation. Pourtant, s'il y a une chose de plus à préciser ici, c'est que la vérification sera fidèle au HLMP si celui-çi n'a aucun comportement non déterministe. D'une part, ceci requiert que le HLMP soit linéaire, c'est-à-dire, que toutes les conditions sur l'évolution soient de la forme $\dot{x} = c$ pour une valeur réelle c : de cette manière, l'état atteint après le délai t est déterminé. D'autre part, ne pas avoir de comportement non-déterministe requiert aussi que les transitions probabilistes soient définies sur l'ensemble des états de la σ-algèbre. Dans le cas contraire, le non déterminisme devra être résolu lors de la construction du LMP. D'une manière générale, il revient au contexte de décider si la discrétisation du temps est adéquate en fournissant une distribution appropriée sur des rectangles. Enfin, ayant un LMP à notre portée, nous pouvons utiliser une méthodologie de vérification applicable aux LMP. Au nombre de ces méthodologies, on peut citer les approximations finies [13], les distances [43, 18], et

un vérificateur [38]. On pourrait se demander pourquoi définir les HLMP si nous devons encore les transformer en LMP. Nous croyons qu'avoir un formalisme pour spécifier exactement un système est en soi important. Il reste, tout de même, à savoir s'il existe des techniques pour vérifier exactement, approximativement, automatiquement ou semi-automatiquement ces modèles. Il faut reconnaître qu'un modèle incorrect du système peut convenir dans certaines situations mais pas dans d'autres : en disposant seulement de l'approximation à notre portée, nous devons construire un nouveau modèle, peut être moins complexe, pour un autre usage. Dans les faits, nous voulons que ce processus soit transparent pour l'utilisateur : on pourrait automatiser le processus de vérification. Il consisterait en la création d'un outil automatique qui peut construire un LMP à partir d'un HLMP – et une unité de temps – en temps polynomial. Réciproquement, étant donnés un HLMP et une propriété à vérifier, le même outil pourrait être en mesure de choisir une unité convenable pour la construction du LMP qui se rapproche le plus du modèle original.

L'autre approche que nous proposons est plus complexe et a besoin de travaux plus approfondis. Les HLMP pourraient être vérifiées en les adaptant aux algorithmes conçus pour les PHA et les LMP. Dans le cas non probabilitiste, les méthodes pour vérifier des sous-classes de PHA ont été proposées [42, 41]. Récemment, une analyse symbolique a été établie [21]. Ces méthodes, combinées à la méthodologie et à l'outil développé par Richard [38] devraient s'étendre aux HLMP. Nous laissons cela aux travaux futurs mais nous pensons que la construction de la dite plateforme mathématique dont on aura besoin consistera probablement en la combinaison des techniques existentes pour les PHA et les LMP.

Chapitre 7

Conclusion

Le travail présenté, à travers ce document, a fait l'objet de deux articles publiés dans des ateliers internationaux [6, 7]. Son principal but est de faire la comparaison entre les processus de Markov étiquetés et les automates hybrides probabilistes, deux types de modèles probabilistes qui évoluent dans des environnements continus [6]. Cette continuité provient de différents éléments propres à chacun de ces modèles. De plus, ce travail présente des résultats sur la vérification formelle des automates hybrides probabilistes [7].

Dans le second chapitre, nous avons présenté une introduction aux familles de systèmes probabilistes à temps discret et à temps continu. Nous avons défini chacune de ces familles en nous basant sur deux types de modèles simplifiés, celui des systèmes de transitions probabilistes et celui des automates temporisés probabilistes, qui sont utilisés pour représenter une classe de systèmes de chacune de ces deux familles. Ainsi, les systèmes à temps discret évoluent suivant au rythme d'une unité de temps définie, et les informations sur l'état du système entre deux unités de temps sont inconnues. Contrairement à ces derniers, les systèmes à temps continu fonctionnent de manière continue et les informations sur le système sont accessibles à tout moment.

Dans le troisième chapitre, nous avons étudié les processus de Markov étiquetés ou LMP. Ceux-ci sont utilisés pour modéliser des systèmes probabilistes à espace d'états continu mais à temps discret. Les LMP regroupent des systèmes complexes au comportement aléatoire comme la machine à café qui choisit, selon une loi de probabilité, la quantité de café à servir. Contrairement à un système de transition probabiliste muni de distributions discrètes, la définition des transitions d'un état vers un autre dans un LMP n'est pas suffisante, puisque l'ensemble des états peut être non-dénombrable. De plus, étant un modèle de représentation de systèmes à temps discret, les transitions

dans un LMP sont discrètes. Enfin, ces transitions sont telles qu'elles peuvent être accompagnées par des distributions de probabilité complexes pouvant être des fonctions de transitions probabilistes ou des lois de probabilité continues.

Nous avons, dans le quatrième chapitre, présenté les systèmes hybrides probabilistes. Il s'agit de systèmes probabilistes qui évoluent en continu et qui exécutent des transitions discrètes. Comme tous les systèmes à temps continu, l'ensemble des états dans un système hybride est infini et non-dénombrable. Le formalisme que nous avons utilisé pour représenter ces types de systèmes est celui des automates hybrides probabilistes. Selon ce formalisme, les états du système ayant les mêmes comportements continus sont regroupés et chaque groupe d'états est relié à un autre grâce à des transitions discrètes probabilistes. Contrairement aux LMP, ces transitions sont accompagnées de distributions de probabilité discrètes. Avec des exemples de systèmes, comme celui de la machine d'une usine, nous avons illustré la théorie des automates hybrides probabilistes. La complexité de ce type de modèle fait que peu de travaux dans le domaine de la vérification des automates hybrides ont été faits. Nous nous sommes intéressés à ce domaine d'étude et avons présenté les résultats d'un de nos travaux qui constituent une contribution à l'analyse et à la vérification des automates hybrides probabilistes. Par ce travail, nous avons défini des méthodes d'approximation qui pourraient être utilisées pour vérifier les systèmes hybrides probabilistes.

Au cinquième chapitre, nous avons fait une étude comparative des LMP et des automates hybrides probabilistes. Notre méthode a été d'utiliser deux exemples de systèmes probabilistes. Le premier exemple est celui d'un avion qui perd de l'altitude chaque fois que le pilote le fait tourner. Le comportement probabiliste est attaché à la perte d'altitude. Le second exemple, quant à lui, est celui d'un bateau dont le mouvement sur une grille a un comportement aléatoire. Chacun de ces systèmes a été modélisé dans les deux formalismes. Les résultats de ces modélisations nous ont permis de ressortir les différences et les limitations de ces deux types de modèles et de dégager les caractéristiques des systèmes pouvant être modélisées par chacun des formalismes. Ainsi, nous avons pu constater que dans les LMP, la continuité est essentiellement liée aux distributions de probabilité, tandis que dans les PHA, elle réside dans l'évolution des états. De plus, les LMP ont la particularité de définir une granularité dans leurs états et leurs transitions, ce qui leur permet de représenter des systèmes complexes caractérisés par des distributions de probabilité complexes. Les PHA, eux, ont leurs états définis de façon plus compacte, et les transitions vers ces états sont munies de distributions discrètes. Sur ce point, il existe donc des systèmes qui peuvent être représentés par les LMP et qui ne pourraient l'être par aucun PHA. Toutefois, les PHA sont sémantiquement des LMP, mais ces derniers sont inutilisables en pratique à cause de leur grand nombre d'états et de transitions. De plus, les PHA supportent le temps-continu contrairement aux LMP.

Par conséquent, il existe des systèmes qui peuvent être modélisés par un PHA et par aucun LMP. Enfin, nos modélisations ont permis de constater que ni les LMP ni les PHA ne sont suffisamment généraux pour modéliser des systèmes comme l'avion ou le bateau. Des approximations peuvent être définies, et nous en avons proposé une pour chacun des modèles. Pour chacune des approximations et chacun des systèmes, nous avons indiqué des propriétés qui peuvent être vérifiées avec l'une des approximations mais pas avec l'autre. Il faut noter que les exemples utilisés sont les premiers exemples complexes qui intègrent à la fois les probabilités et l'espace d'états continu. Il est vrai qu'ils sont simples, mais cet aspect fait partie de leur valeur : à la fois riches et compréhensibles. Ils pourront être utilisés, entre autres, pour comparer différents formalismes ou pour tester des outils de vérification.

Enfin, dans le sixième chapitre, nous avons proposé une généralisation des LMP et des PHA en vue de contourner leurs limites respectives. Ainsi, nous avons défini les LMP hybrides (HLMP) et leur sémantique. Le formalisme des LMP hybrides, même s'il est très théorique, permet de définir une nouvelle classe de systèmes probabilistes et de modéliser les systèmes qui combinent aussi bien le temps continu que les distributions continues. Bien entendu, ce dernier permet de modéliser les systèmes de nos études de cas, et nous avons procédé à la modélisation de l'avion. Enfin, nous avons proposé des approches de vérification qui utilisent des techniques de vérification existantes.

Des efforts devront être faits dans le but d'étendre davantage la théorie des LMP et des PHA au monde des HLMP. Pour cela, une approche serait de reprendre l'exercice fait dans ce travail, mais en utilisant des cas plus complexes. Des efforts devraient également être faits en vue d'approfondir la recherche de méthodes de vérification des HLMP. Les méthodes d'approximation que nous avons proposées dans ce travail pourraient être des approches. De plus, l'on pourrait également étendre les techniques d'approximation que nous avons proposées [7] pour la vérification des PHA, sur les HLMP. Pour ce faire, l'on devra trouver une méthode pour appliquer ces techniques dans un environnement où les distributions sont continues. Des techniques d'approximation sur les LMP pourraient, en ce moment, servir. Une autre méthode d'approximation pouvant être considérée est celle qui consiste à discrétiser le temps en utilisant une unité de temps qui permettrait d'obtenir un LMP suffisamment expressif pour être vérifié. Des notions de profondeur et de distance [43, 18] pourront, dès lors, être considérées. Enfin, il faudrait étendre la théorie des HLMP, et définir plus clairement la classe des systèmes qu'ils couvrent. Ceci pourrait être fait en les comparant à d'autres types de modèles probabilistes continus existants.

Bibliographie

[1] 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. *Theor. Comput. Sci.*, 138(1) :3–34, 1995.

[2] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid Automata : An algorithmic approach to the specification and verification of Hybrid Systems. pages 209–229. Springer-Verlag, 1992.

[3] Rajeev Alur, Thao Dang, and Franjo Ivancic. Predicate abstraction for reachability analysis of Hybrid Systems. *ACM Transactions on Embedded Computing Systems*, 5(1) :152–199, 2006.

[4] Saurabh Amin, Ro Abate, Maria Pr, John Lygeros, and Shankar Sastry. Reachability analysis for controlled discrete time stochastic hybrid systems. In *in Hybrid Systems : Computation and Control - HSCC 2006*, pages 49–63. Springer Verlag, 2006.

[5] Bianco Andrea and De Alfaro Luca. Model checking of probabilistic and nondeterministic systems. pages 499–513. Springer-Verlag, 1995.

[6] Joseph Assouramou and Josée Desharnais. Continuous time and/or continuous distributions. In *Proceedings of the 7th European performance engineering conference on Computer performance engineering*, EPEW'10, pages 99–114, Berlin, Heidelberg, 2010. Springer-Verlag.

[7] Joseph Assouramou and Josée Desharnais. Analysis of non-linear probabilistic hybrid systems. In *QAPL*, pages 104–119, 2011.

[8] Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Reachability in continuous-time markov reward decision processes. In J. Flum, E. Graedel, and T. Wilke, editors, *Logic and Automata : History and Perspectives, Aachen, Germany*, volume 2 of *Texts in Logic and Games*, pages 53–71, Amsterdam, 2008. Amsterdam University Press.

[9] Christel Baier and Marta Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11 :125–155, 1998.

[10] Danièle Beauquier. On probabilistic Timed Automata. *Theor. Comput. Sci.*, 292 :65–84, January 2003.

[11] Patrick Billingsley. *Probability and Measure*. Wiley, New York, NY, 1979.

[12] Richard Blute, Josee Desharnais, Abbas Edalat, and Prakash Panangaden. Bisimulation for labelled markov processes. In *Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science*, LICS '97, pages 149–, 1997.

[13] Alexandre Bouchard-Côté, Norm Ferns, Prakash Panangaden, and Doina Precup. An approximation algorithm for labelled markov processes : towards realistic approximation. In *QEST'05*, pages 54–62, 2005.

[14] Josée Desharnais. *Labelled Markov Processes*. PhD thesis, McGill University, School of Computer Science, 1999.

[15] Josée Desharnais, Abbas Edalat, and Prakash Panangaden. Bisimulation for Labelled Markov Processes, 1999.

[16] Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Approximating Labeled Markov Processes. *Information and Computation*, 184(1) :160–200, July 2003.

[17] Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for Labelled Markov Processes. *Theor. Comput. Sci.*, 318(3) :323–354, 2004.

[18] Josée Desharnais, François Laviolette, and Sami Zhioua. Testing probabilistic equivalence through reinforcement learning. In *Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science*, FSTTCS 2006, 2006.

[19] Kim G. Larsen et Anne Skou. Bisimulation through probabilistic testing. *Information and Compututation*, pages 99 :1–28, 1991.

[20] Ansgar Fehnker and Franjo Ivancic. Benchmarks for Hybrid Systems verification. In *In Hybrid Systems : Computation and Control (HSCC 2004) (2004*, pages 326–341. Springer, 2004.

[21] Martin Fränzle, Holger Hermanns, and Tino Teige. Stochastic satisfiability modulo theory : A novel technique for the analysis of probabilistic Hybrid Systems. In *HSCC '08 : Proceedings of the 11th international workshop on Hybrid Systems*, pages 172–186, Berlin, Heidelberg, 2008. Springer-Verlag.

[22] Goran Frehse. Phaver : Algorithmic verification of Hybrid Systems past HyTech. pages 258–273. Springer, 2005.

[23] Ernst Hahn, Holger Hermanns, Stefan Ratschan, Zhikun She, and Lijun Zhang. Safety verification for probabilistic Hybrid Systems. In Tayssir Touili, Byron Cook, and Paul Jackson, editors, *Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 196–211. Springer Berlin / Heidelberg, 2010.

[24] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6 :102–111, 1994.

[25] George Hassapis and Isabella Kotini. Verification of rectangular Hybrid Automata models. *Journal of Systems and Software*, 79(10) :1433–1443, 2006.

[26] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-toi. Algorithmic analysis of nonlinear Hybrid Systems. *IEEE Transactions on Automatic Control*, 43 :225–238, 1996.

[27] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. Hytech : A model checker for Hybrid Systems. In *CAV '97 : Proceedings of the 9th International Conference on Computer Aided Verification*, pages 460–463, London, UK, 1997. Springer-Verlag.

[28] Thomas A. Henzinger and Peter W. Kopke. Discrete-time control for rectangular Hybrid Automata. *Theor. Comput. Sci.*, 221(1-2) :369–392, 1999.

[29] Thomas A. Henzinger, Peter W. Kopke, Anuj Puri, and Pravin Varaiya. What's decidable about Hybrid Automata ? In *STOC '95 : Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 373–382, New York, NY, USA, 1995. ACM.

[30] Ronald A. Howard. *Dynamic Probabilistic Systems, Volume I : Markov Models (Dynamic Probabilistic Systems)*. Dover Publications, June 2007.

[31] Marcin Jurdziński, François Laroussinie, and Jeremy Sproston. Model checking probabilistic timed automata with one or two clocks. In *TACAS'07 : Proceedings of the 13th international conference on Tools and algorithms for the construction and analysis of systems*, pages 170–184, Berlin, Heidelberg, 2007. Springer-Verlag.

[32] Peter William Kopke, Jr. *The theory of rectangular Hybrid Automata*. PhD thesis, Ithaca, NY, USA, 1996. Adviser-Henzinger, Thomas A.

[33] Marta Kwiatkowska, Gethin Norman, and David Parker. Stochastic model checking. In M. Bernardo and J. Hillston, editors, *Formal Methods for the Design of Computer, Communication and Software Systems : Performance Evaluation (SFM'07)*, volume 4486 of *LNCS (Tutorial Volume)*, pages 220–270. Springer, 2007.

[34] Marta Kwiatkowska, Gethin Norman, Jeremy Sproston, and Fuzhi Wang. Symbolic model checking for probabilistic timed automata. In Y. Lakhnech and S. Yovine, editors, *Proc. Joint Conference on Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems (FORMATS/FTRTFT'04)*, volume 3253 of *LNCS*, pages 293–308. Springer, 2004.

[35] Daniel J. Lehmann and Saharon Shelah. Reasoning with time and chance (extended abstract). In *Proceedings of the 10th Colloquium on Automata, Languages and Programming*, pages 445–457, London, UK, 1983. Springer-Verlag.

[36] Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. An approach to the description and analysis of Hybrid Systems. *Lecture notes in Computer Science*, 736 :149–178, 1993.

[37] Stefan Ratschan and Zhikun She. Safety verification of Hybrid Systems by constraint propagation-based abstraction refinement. *ACM Trans. Embed. Comput. Syst.*, 6, February 2007.

[38] Nicolas Richard. Labelled Markov Processes. Master's thesis, Département d'informatique et de génie logiciel, Faculté des sciences et de génie, UNIVERSITE LAVAL, 2003.

[39] Sheldon M. Ross. *Initiation aux probabilités*. Presses polytechniques et universitaires romandes, quatrième édition, 2006.

[40] Jeremy Sproston. Decidable model checking of probabilistic Hybrid Automata. In *FTRTFT '00 : Proceedings of the 6th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, pages 31–45, London, UK, 2000. Springer-Verlag.

[41] Jeremy Sproston. *Model Checking of Probabilistic Timed and Hybrid Systems*. PhD thesis, University of Birmingham, Faculty of Science, 2000.

[42] Jeremy Sproston. Analyzing subclasses of probabilistic Hybrid Automata. *CiteSeerX-Scientific Literature Digital Library and Search Engine [http ://citeseerx.ist.psu.edu/oai2] (United States)*, 2007.

[43] Franck van Breugel, Babita Sharma, and James Worrell. Approximating a behavioural pseudometric without discount for probabilistic systems. In Helmut Seidl, editor, *FoSSaCS*, volume 4423 of *Lecture Notes in Computer Science*, pages 123–137. Springer, 2007.

Annexe A

Rappel sur les probabilités

Cette annexe présente un rappel de notions sur les probabilités, et des variables aléatoires continues présentées plus en détails dans [39].

A.0.1 Les variables aléatoires continues

Définition A.0.1. *X est une* **variable aléatoire continue** *s'il existe une fonction f non-négative définie pour tout $x \in \mathbb{R}$ et vérifiant $\forall E \subseteq \mathbb{R}$:*

$$P\{X \in \mathbb{R}\} = \int_E f(x)dx.$$

f est la **fonction de densité** *associée à la variable X. Donc, la probabilité que X prenne une valeur dans l'ensemble E se calcule en utilisant l'intégrale de E. Pour obtenir une probabilité, f doit respecter la contrainte suivante :*

$$P\{X \in (-\infty, \infty)\} = \int_{-\infty}^{\infty} f(x)dx = 1.$$

À partir de cette définition, on déduit que :

1. $P\{a \leq X \leq b\} = \int_a^b f(x)dx$,
2. $P\{X = a\} = \int_a^a f(x)dx = 0$,
3. $P\{a \leq X \leq b\} = P\{a < X < b\}$.

Définition A.0.2. *La **fonction de répartition** F d'une variable aléatoire continue X, dont la fonction de densité est f, est définie par :*

$$F(x) = P\{X \leq a\} = \int_{-\infty}^{a} f(x)dx$$

ou, autrement

$$\frac{d}{da}F(a) = f(a).$$

Alors, on a :

1. $P\{a \leq X \leq b\} = F(b) - F(a)$
2. $P\{X = a\} = F(a) - F(b)$
3. $\lim_{x\to-\infty} F(x) = 0$
4. $\lim_{x\to\infty} F(x) = 1$.

Définition A.0.3. *L'**espérance mathématique** d'une variable aléatoire continue X, dont la fonction de densité est f, est définie par :*

$$E[X] = \int_{-\infty}^{\infty} xf(x)dx.$$

Définition A.0.4. *La variance d'une variable aléatoire continue X est définie par :*

$$Var(X) = E[X^2] - E[X]^2.$$

Définition A.0.5. *L'écart-type d'une variable aléatoire X, qui se note σ, est définie par :*

$$\sigma = \sqrt{Var(X)}.$$

A.0.2 Quelques lois continues

Loi uniforme

Une variable aléatoire X est dite uniformément distribuée sur l'intervale (α, β) si sa fonction de densité est

$$f(x) = \begin{cases} \frac{1}{\beta-\alpha} \text{ si } \alpha < x < \beta, \\ 0 \text{ sinon.} \end{cases}$$

Loi normale (μ, σ^2)

Une variable aléatoire X est dite **normale** avec les paramètres μ et σ^2 si la densité de X est donnée par :

$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \ \forall x \in \mathbb{R}.$$

La fonction de répartition d'une variable Normale (0,1) est noté $\Phi(x)$ et est définie par :

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-x^2}{2}} dy.$$

De ces définitions, on déduit :

1. E[X] $= \mu$
2. Var[X] $= \sigma^2$
3. Si X est une variable suivant la loi Normale(μ, σ^2), alors $Y = \sigma X + \beta$ suit une loi Normale $(\alpha\mu + \beta, \alpha^2\sigma^2)$.
4. La fonction de répartition d'une variable X Normale (μ, σ^2) est :

$$F(x) = \Phi(\frac{x-\mu}{\sigma}).$$

5. Si X est une variable Normale (μ, σ^2) est :

$$P(a \leq X \leq b) = \Phi(\frac{b-\mu}{\sigma}) - \Phi(\frac{a-\mu}{\sigma}).$$

Loi exponentielle $(\lambda > 0)$

La fonction de densité d'une variable aléatoire suivant une loi exponentielle de paramètre $\lambda > 0$ est :

$$f(x) = \int_0^{\lambda e^{-\lambda x} \text{ si } x>0} \text{ sinon.}$$

De cette densité, on déduit que :

1. $F(x) = 1 - e^{-\lambda x}$, pour $x \geq 0$.
2. $E[X] = \frac{1}{\lambda}$.
3. $Var[X] = \frac{1}{\lambda^2}$.
4. Les variables exponentielles ont la propriété d'être sans mémoire, c'est-à-dire :

$$P\{X > s + t | X > t\} = P\{X > s\} \text{ pour tous } s, t \geq 0.$$

5. Cette propriété est appelée **propriété de Markov** et elle s'applique aux LMP.

Annexe B

Première contribution

Nous présentons, ici, une contribution qui a été publiée à la conférence internationale *European Performance Engineering Workshop* de 2011. Il s'agit d'un article qui traite essentiellement de la comparaison entre les processus de Markov étiquetés et les automates hybrides probabilistes, principal sujet de ce mémoire.

Continuous time and/or continuous distributions

Joseph Assouramou* Josée Desharnais*
joseph.assouramou.1@ulaval.ca josee.desharnais@ift.ulaval.ca

Department of Computer Science and Software Engineering
Université Laval, Québec, Canada

Abstract. We compare two models of processes involving uncountable space. Labelled Markov processes are probabilistic transition systems that can have uncountably many states, but still make discrete time steps. The probability measures on the state space may have uncountable support. Hybrid processes are a combination of a continuous space process that evolves continuously with time and of a discrete component, such as a controller. Existing extensions of Hybrid processes with probability restrict the probabilistic behavior to the discrete component. We use an example of an aircraft to highlight the differences between the two models and we define a generalization of both that can model all the features of our aircraft example.

1 Introduction

For many years now, two models of continuous processes have evolved independently. Labelled Markov processes (LMPs) [2] are probabilistic transition systems that can have uncountably many states, but still make discrete time steps. Hybrid processes [1,12] are a combination of a discrete component, such as a controller and a physical process that evolves continuously with time. Existing extension of Hybrid processes with probabilities [14] restrict the probabilistic behaviour to the discrete component.

For both models, notions of bisimulation and simulation, logics, model checking techniques and tools have been developed in disjoint research communities. These models face similar challenges, as their main complexity is of course their continuous nature. For example, notions of approximations have been developed on both sides [10,16,7]. No systematic comparison between the two models has been proposed until now. Nevertheless, it is not clear at first sight if the continuous time model could not encompass the discrete time one. Moreover, in order to implement a model-checking tool for LMPs, a finite language had to be defined to describe LMPs. This was done by Richard [13]. The input language to the tool is an LMP whose probability transitions are combinations of known distributions such as Uniform, Normal, etc. One could wonder if this finiteness has some link with the discrete component of hybrid automata.

* Research supported by NSERC.

If the two models have evolved independently, is that to say that they are made for different purposes and hence that there is no need for a unifying framework? On another hand, one could wonder whether there is some satisfactory translation from one to the other. In this paper, we answer these questions by way of a case study. We will show how to model with both frameworks a simple process representing an aircraft flying. Though simplified, this aircraft will be interesting enough to highlight the differences and limitations of the two frameworks considered. These observations will lead us to define a generalization of labelled Markov processes and hybrid probabilistic processes, that we will call hybrid LMPs. Hence, this model will combine both distinguishing features of the two frameworks, that is, continuous time and continuous distributions.

1.1 The aircraft

Our base example is an aircraft taking off and travelling. Initially, the aircraft is on the ground and once started, it rises up as long as the maximum altitude is not reached. The rising rate is between 20 and 25 m by unit of time. At any time, the pilot may rotate to the right or left; while rotating, the airplane, under certain friction forces (air resistance, etc.), may lose some altitude, following a probability law. If it gets to a zero altitude after rotating, we will assume that it has crashed. Moreover if the pilot tries to rotate the aircraft when the altitude is below the minimal value $H_{\min}$, the aircraft crashes. Before formalizing further this case study, we observe that this example describes a probabilistic system with a continuous state space since the altitude is a real value. Moreover, the system is a continuous time system because the height evolves with time. The system allows only one action that we call "*rotate*". To keep the example simple, we do not allow other actions such as decreasing altitude or landing.

In order to get hands on a precise instance of this model, we now choose a probability function that witnesses the behavior described above. We expect that after a rotation, the plane is more likely to jump to some close height than far from its previous height. Thus, the probability that from an altitude s, the aircraft loses about 20 meters of height should be greater than its probability to lose 50 meters. Assume that the possible values of the altitude are in $R := [0, H_{\max}] \cup \{\mathit{Crashed}\}$ where $H_{\max}$ is the maximum altitude. We will denote by $p(s, [a, b])$ the probability that from altitude s, the aircraft's altitude gets in $[a, b]$ after a rotation. We choose an exponential distribution on $s - x$ where s is the altitude of the aircraft when the pilot makes a rotation. This will make sure that intervals of height values that are closer to s will get greater probability. We define p as follows, where s is in units of height (not necessarily meters):

- if $0 \leq s < H_{\min}$, $p(s, \{\mathit{Crashed}\}) := 1$
- if $H_{\min} \leq s \leq H_{\max}$, $p(s, \cdot)$ is the unique probability measure extension of the following set function (the case $a \leq s < b$ can be deduced)
  - $p(s, [a, b]) := \begin{cases} \int_a^b e^{-(s-x)} dx & \text{if } 0 < a \leq b \leq s \\ p(s, [a, s]) & \text{if } s < b \quad \text{hence 0 if } s < a \end{cases}$
  - $p(s, \{\mathit{Crashed}\}) := \int_{-\infty}^{0} e^{-(s-x)} dx = e^{-s}$

Note that this choice of probability measure to model the aircraft does not record the possible instability that may arise from the rotation: we assume that the altitude loss is measured once the aircraft has stabilized. We also assume that only a loss of altitude is possible (i.e. if $s \leq a \leq b$, $p(s, [a, b]) = 0$). This simplification will play no role in the point we want to make about this benchmark. In the two models that we will present, we will need numerical values to express properties such as "at a an altitude greater than $H_{\min}$, the probability that the aircraft loses 100 meters or more when rotating is between 25% and 50%": in these cases, our unit of height will be 100 meters and $H_{\min} := 5$.

In the following section, we recall different definitions; in Section 3, we will present models of the aircraft using the two frameworks together with a few properties that can or cannot be verified faithfully; and finally, in Section 4, we propose a generalization of both frameworks.

2 Background

In this section, we recall some definitions, including those of the two models that we will analyze in this paper.

A *measurable space* is a pair (S, Σ) where S is any set and $\Sigma \subseteq \mathcal{P}(S)$ is a *σ-algebra* over S, that is, a collection of subsets of S containing S and closed under complementation and countable intersection. Well-known examples are $[0, 1]$ and $\mathbb{R}$ equipped with their respective *Borel* σ-algebras $\mathcal{B}$, generated by intervals. Throughout the paper, we assume the Borel σ-algebra on $[0, 1]$ and $\mathbb{R}$ and we write $\mathcal{B}_n$ for the Borel σ-algebra on $\mathbb{R}^n$. A map f between two measurable spaces (S, Σ) and (S', Σ') is said to be *measurable* if for all $A' \in \Sigma'$, $f^{-1}(A') \in \Sigma$. A necessary and sufficient criterion for measurability of a map $f : (S, \Sigma) \to ([0, 1], \mathcal{B})$ is that the condition be satisfied for $A' := [r, 1]$, for any rational r. A *subprobability measure* on (S, Σ) (or probability distribution) is a map $p : \Sigma \to [0, 1]$, such that for any countable collection (E_n) of pairwise disjoint sets, $p(\cup_n E_n) = \sum_n p(E_n)$. We say that p is discrete when its support is finite or countable, that is, if there is a countable set $E = \mathrm{supp}(p) := \{s \in S : p(\{s\}) > 0\} \in \Sigma$ such that $p(E) = p(S)$. The set of discrete distributions over S will be denoted by $\mathrm{Dist}(S)$, and the set of all subprobability measures, $\mathsf{Sub}(S)$.

2.1 Labelled Markov Processes (LMPs)

Labelled Markov Processes are used to model reactive and probabilistic systems whose state spaces might be continuous, that is, systems that react to events in their environment and may have uncountably many states.

Definition 2.11 ([2]) *Let Act be a countable set of actions and AP a countable set of atomic propositions. A* Labelled Markov Process (LMP) *is a tuple $(S, \Sigma, i, \{\mu_a\}_{a \in Act}, Label)$ where:*

- *S the set of* states*, (S, Σ) is a measurable space and $i \in S$ is the initial state*
- *$\mu_a : S \times \Sigma \to [0, 1]$ is a transition (sub)probability function $\forall a \in Act$, i.e.,*

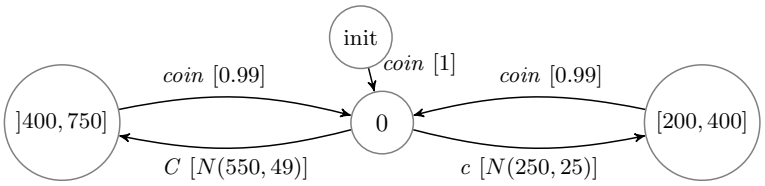

Fig. 1. LMP of a coffee maker 2.12

- *for each* $s \in S$, $\mu_a(s, \cdot)$ *is a subprobability measure and*
- *for each* $E \in \Sigma$, *the function* $\mu_a(\cdot, E)$ *is measurable*

– $Label : S \to \mathcal{P}(AP)$ *is a measurable function used to describe states.*

The value $\mu_a(s, E)$ is the probability that, starting in state s and under the request of action a, the system makes a transition into one of the states in E, in one unit of time; hence time is discrete. To model a continuous time phenomenon using an LMP, one can monitor its values at fixed intervals of time, making the time discrete. An extension of LMPs allowing external non deterministic transitions is obtained essentially by allowing a (countable) set of distributions for every action. For a complete treatment of non deterministic LMPs, one can refer to D'Argenio et al. [6] or Cattani et al. [5]. A typical example of LMP is an interacting machine.

Example 2.12 *Consider a coffee machine with two buttons c and C for serving a small and a large coffee respectively. Once the coffee is served, in* 99% *of the case, the system is ready to serve a coffee as soon as a coin is inserted. We limit the state space to* $\{\text{init}, 0\} \cup [200, 750]$*: initially, the machine is in state* init *and real numbers represent the amount of coffee served when the customer presses one of the two buttons. The small and the large coffee correspond to states in the intervals* $[200, 400]$ *and* $]400, 750]$ *respectively. We associate a distribution* μ_c *that follows a Normal law of mean 250 and variance 25 to the small coffee button and another Normal distribution* μ_C *of mean 550 and variance* 49 *to the large coffee button. An LMP model of such a system is defined over* $AP := \{\text{small,large}\}$*, as* $(\{\text{init}, 0\} \cup [200, 750], \mathcal{B}, 0, \{\mu_c, \mu_C, \mu_{coin}\}, Label)$*, where* $Label([200, 400]) := \{\text{small}\}$ *and* $Label(]400, 750]) := \{\text{large}\}$*. It is depicted in Figure 1.*

In graphic representations, each transition is labelled with an action followed, in brackets, by the probability distribution that is restricted to the target set. Thus, the transition labelled by $c[N(250, 25)]$*, encodes a c-transition from the state* 0 *whose subprobability measure function follows a Normal distribution on the target set* $[200, 750[$*. If the capacity of large glasses is less than* 700 ml*, we can compute the probability that the coffee overflows, which is given by* $\mu_C(0, [700, 750])$*. So, there is* 1% *chance that a coffee overflows.*

Model-checking methods Because of their infinite state space, the model-checking of LMPs can only be done for sub-families of processes that exhibit

some form of regularity. Richard [13] has developed Cismo[1], a model-checking tool to verify next state probabilistic properties (possibly nested and branching). A language for LMPs was defined in order to feed the model-checker: the state spaces allowed are powers of the reals. The input language to the tool is an LMP whose transition probability functions are combinations of known distributions such as Uniform, Normal, or any repartition function.

2.2 Probabilistic Hybrid Systems

Hybrid systems are dynamic systems that combine discrete and continuous changes, and thus, they are defined as the composition of a dynamic continuous system and a discrete one. Contrarily to the LMP model, the change of state is continuous with time. The following (non probabilistic) example of a monitor is typical.

Example 2.21 *Consider a monitor that measures the performance of a cement factory machine and assume that this performance is measured continuously on a scale of* $[0, 1]$*. Thus, when the performance is* 0.5*, the machine is used at* 50% *of its maximum resources. We suppose that there exist three operating machinery modes: "Stable" is the normal mode of operation where the performance of the machine increases linearly in the interval* $[0.8, 1]$*; in mode "Unstable", the performance varies at a rate between* $[-\frac{1}{64}, \frac{1}{64}]$ *in the interval* $[0.4, 0.8]$*; finally, "Low" is the mode in which the machine's performance is smaller than* 0.4 *and decreases gradually at a linear rate. Discrete steps can happen when, depending on the needs, the operator decides to change the machine's mode by pressing on one of the buttons: "low mode", "stable mode", "unstable mode".*

Probabilistic hybrid systems are hybrid systems for which relative likelihoods are associated with certain behavior. Therefore, we can talk about probabilistic timed properties such as "with a probability greater than 0.9, some system will be in some target state within 3 minutes".

Example 2.22 *Consider the monitor of Example 2.21, but now assume that there is only one button to switch from one mode to another. From the mode "Stable", by pressing the button, the machine may switch either to mode "Low", with probability* 0.3 *or to the mode "Unstable", with probability* 0.7*.*

The formalism used for the specification and verification of hybrid systems was introduced independently by Alur and al. [1] and Nicollin and al. [12]. Sproston has extended their formalism with probabilistic behaviour [14,15]. There are many subclasses of hybrid systems, among which linear [10] and rectangular [11]. On the probabilistic side, the same classes are defined; for simplicity, we will restrict to the probabilistic rectangular ones. We first need some notation. Given a finite set X of real variables, we write $\dot{X} := \{\dot{x}_1, ..., \dot{x}_n\}$ where $\dot{x}_i := \frac{dx_i}{dt}$ is the first derivative of x_i with respect to time. A valuation $\mathtt{a}$ is a function $\mathtt{a} : X \to \mathbb{R}$ that assigns values to variables; we equivalently write $\mathtt{a} \in \mathbb{R}^n$ where $n = |X|$.

[1] More can be found on the web site of Cismo: `http://www2.ift.ulaval.ca/~jodesharnais/VPS/`.

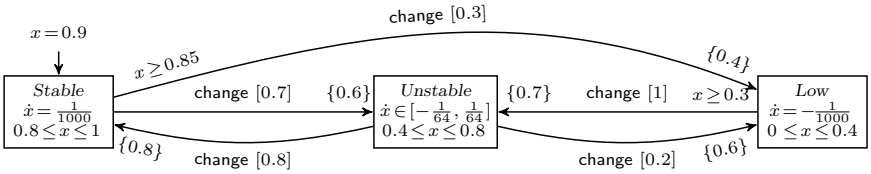

Fig. 2. The PHA of a machine factory's monitor

Definition 2.23 *Let X be a finite set of variables. A set U of valuations is* rectangular *if there exists a family of (possibly unbounded) intervals $(I_x)_{x \in X}$ with rational endpoints such that $U = \{\mathsf{a} \in \mathbb{R}^n \mid \mathsf{a}(x) \in I_x \text{ for all } x \in X\}$. We denote by $R(X)$ the set of rectangles over X.*

Definition 2.24 ([15]) *A* probabilistic rectangular hybrid automaton (PHA) *is a structure $H = (X, V, init, A, inv, flow, prob, \langle pre_{v,a} \rangle_{v \in V, a \in A})$ such that:*

- *X is a finite set of real variables.*
- *V is a finite set of locations or control modes.*
- *$init : V \to R(X)$ is the function that maps every location to an initial set.*
- *A is a finite set of actions.*
- *$inv : V \to R(X)$ defines invariants for the variables in each location.*
- *$flow : V \times \mathbb{R}^n \to R(\dot{X})$ is a flow evolution condition.*
- *$prob : V \times A \to \mathcal{P}_{fin}(\mathrm{Dist}(V \times R(X) \times \mathcal{P}(X)))$ encodes probabilistic transitions.*
- *$pre_{v,a} : prob(v, a) \to R(X)$ defines preconditions for distributions.*

For simplification of notation, we drop the subscripts of *pre* when they are clear from the context. The current state of a PHA is expressed as the current location and the values of every variables of X. Hence, a state is a pair $(v, \mathsf{a}) \in V \times \mathbb{R}^n$. If $\mathsf{a} \in inv(v)$, then we say that the state (v, a) is *admissible*. The continuous evolution is encoded in every $flow(v, \mathsf{a})$, which is a rectangle containing all valuations that the first derivative of the variables of X can take. Given $a \in A$, a transition labelled a can be taken from a state (v, a) if there is some $\mu \in prob(v, a)$ such that $\mathsf{a} \in pre_{v,a}(\mu)$, i.e., the associated precondition is satisfied. Then, the value $\mu(v', \mathsf{post}, Y)$ is the probability that given action a from location v, the system changes location to v', and the valuation of the variables is in the rectangle $\mathsf{post} \subseteq \mathbb{R}^n$. More precisely, only variables that are in Y may change value in this transition. The formal semantics will be given in Definition 2.26.

The flow function of Definition 2.24 is quite general, as it allows, for any location, to specify a different target rectangle for *any* single valuation. Most of the time, the flow is independent of the valuations and variables are often independent of each other; hence the flow function can be more simply defined from V to $R(\dot{X})$, as in the following example.

Example 2.25 *Figure 2 shows a graphical representation of the monitor of Ex-*

ample 2.22. The variable x represents the performance of the machine; since there is only one variable, a valuation is just a real value. The automaton is composed of three locations, Stable, Unstable, Low. The initial state is $(Stable, 0.9)$; hence $init(Stable) = \{0.9\}$ and init is $\emptyset$ elsewhere. For the action change *from Stable, there is one distribution μ satisfying $\mu(Unstable, \{0.6\}, \{x\}) = 0.7$ and $\mu(Low, \{0.4\}, \{x\}) = 0.3$; its preconditions are $pre_{Stable,change}(\mu) = [0.85, 1]$. The admissible states of the location Stable are of the form $(Stable, \mathsf{a})$ where $\mathsf{a}(x) \in inv(Stable) = [0.8, 1]$. For the evolution flow on Stable, the parameter x of admissible states evolve at the rate $\frac{1}{1000}$; so $flow(Stable) = \{\frac{1}{1000}\}$. However, in the location Unstable, the evolution is non-deterministic because x can evolve at any rate in the rectangle $[-\frac{1}{64}, \frac{1}{64}]$, so that we have $flow(Unstable) = [-\frac{1}{64}, \frac{1}{64}]$.*

Model-checking Hybrid systems Because of their continuous nature, model-checking hybrid systems can only be done for sub-classes of hybrid systems. For instance, in the non-probabilistic case, Henzinger and al. [10] proposed two algorithms that allow the verification of safety properties on initialised hybrid systems. In the probabilistic case, Sproston proposed methods to verifiy some sub-classes of PHAs [14], and to verify $\forall$-PBTL on rectangular PHAs [15]. More recently, symbolic analysis have been established [9]and other approximation techniques [16].

Hybrid to concurrent systems When defining PHAs, Sproston proposed a semantics in terms of probabilistic concurrent systems [14]. In these systems, time delay actions encode the continuous time nature of PHAs. They are essentially LMPs with non deterministic and discrete transition functions and with uncountably many actions. It is worth mentioning that the concurrent systems arising from a PHA have essentially a theoretical purpose: PHAs are a language to model dynamical continuous phenomena, and we need to formalise what the syntax of this language means; the semantics as concurrent systems is also used to define relations between PHAs, such as the well-known equivalence notion of bisimulation. The obtained concurrent systems cannot be used in practice, they cannot be, for example, model-checked, as they are way too general: for example, all the state changes that occur through time are represented as uncountable, unrelated non deterministic transitions, where the linearity of time is discarded.

Before defining the semantics, we give some notation[2] and insights that will help to understand the semantics of transitions. Let Q be the set of admissible states of a PHA. For each $(v, \mathsf{a}) \in Q$ and each action $a \in A$, we want to define a (possibly uncountable) set of discrete distributions in $\mathrm{Dist}(Q) \subseteq \mathrm{Dist}(V \times \mathbb{R}^n)$, from the set $prob(v, a)$, which contains distributions in $\mathrm{Dist}(V \times \mathcal{P}(\mathbb{R}^n) \times X)$. Each $\mu \in prob(v, a)$ will give rise to as many distributions on $\mathrm{Dist}(Q)$ as there are combinations $\langle \mathsf{b} \rangle := \langle \mathsf{b}_1, \mathsf{b}_2, \ldots, \mathsf{b}_m \rangle$, with $\mathsf{b}_i \in \mathsf{post}_i$ and $\mathrm{supp}(\mu) = \{(v_i, \mathsf{post}_i, Y_i)\}_{i=1}^m$. Let $Target(\mu)$ be the set of all these combinations. This is necessary, as the sets post_i may overlap and may share the same target location v_i. Every combination represents a non deterministic choice defined from μ.

[2] We change slightly the notation of [14], in order to clarify the definition.

Definition 2.26 *Given a PHA* $H = (X, V, init, A, inv, flow, prob, \langle pre_v \rangle_{v \in V})$, *we derive the associated infinite concurrent probabilistic system*

$$(Q, I, A \cup \mathbb{R}, Steps) \text{ as follows :}$$

- $Q \subseteq V \times \mathbb{R}^n$ *is the set of admissible states;*
- $I = \{(v_0, \mathsf{a}_0) \in Q \mid \mathsf{a}_0 \in init(v_0)\}$ *;*
- $Steps(v, \mathsf{a}) := Cts(v, \mathsf{a}) \cup Dis(v, \mathsf{a})$, *for state* $(v, \mathsf{a}) \in Q$, *and:*
  - $Cts(v, \mathsf{a}) \subseteq \mathbb{R} \times Q$ *contains delay transitions: all pairs* $(d, (v, \mathsf{b}))$ *such that* $d \in \mathbb{R}_{\geq 0}$, $\mathsf{b} \in inv(v)$, *and there exists a differentiable function* $f : [0, d] \to \mathbb{R}^n$ *with* $\dot{f} : (0, d) \to \mathbb{R}^n$ *such that* $f(0) = \mathsf{a}$, $f(d) = \mathsf{b}$, *and for all* $\epsilon \in (0, d)$, $\dot{f}(\epsilon) \in flow(v, f(\epsilon))$ *and* $f(\epsilon) \in inv(v)$*;*
  - $Dis(v, \mathsf{a}) \subseteq A \times \mathrm{Dist}(Q)$ *contains, for each* $\mu \in prob(v, a)$ *with* $\mathsf{a} \in pre(\mu)$, *for each* $\langle \mathsf{b} \rangle \in Target(\mu)$, *all pairs* $(a, \mu_{\langle \mathsf{b} \rangle})$, *where* $\mu_{\langle \mathsf{b} \rangle}$ *is defined as:*

$$\mu_{\langle \mathsf{b} \rangle}(v', \mathsf{c}) := \begin{cases} \sum\limits_{\substack{i=1 \\ \mathsf{c}=\mathsf{b}_i, v_i = v'}}^{|\mathrm{supp}(\mu)|} \mu(v_i, \mathsf{post}_i, Y_i) & \text{if } \mathsf{c} \in inv(v') \\ 0 & \text{otherwise.} \end{cases}$$

The semantics gives us a way of using known notions of equivalence between systems such as bisimulation, simulation.

3 Aircraft modeling

As we have seen, LMPs and PHAs both model continuous state space systems, but in a different way. The purpose of this section is to highlight those differences and the limitations of each of them. To do so, we will attempt to model the aircraft system described in the introduction with the two formalisms.

3.1 The aircraft as an LMP

We first define a model of the aircraft benchmark using an LMP. As the aircraft system is time-continuous, this modelling cannot be done faithfully. We will have to discretize time. Let t be a unit of time, it will be our basic time delay for transitions. The set of states will of course be the possible altitudes of the aircraft together with the crash state: $[0, H_{\max}] \cup \{Crashed\}$, with initial state 0. The σ-algebra is the one generated by the union of $\mathcal{B}$ and the extra state: we denote it by $\mathcal{B} + \{Crashed\}$. The labelling function is not relevant here: it could either label all non-crashed states as *Air*, or those that are between $H_{\min}$ and $H_{\max}$ as *Safe* and the smaller ones as *Low*: this choice depends on the properties one needs to check. The set of actions will contain *rotate*, but also an action τ that will represent internal dynamics that happen at each time unit t. We define the LMP over $Act := \{rotate, \tau\}$ as:

$$G^t_{\mathrm{LMP}} = ([0, H_{\max}] \cup \{Crashed\}, \mathcal{B} + \{Crashed\}, 0, \{\mu_{rotate}, \mu_\tau\}, Label).$$

Two kinds of discrete probabilistic transitions are defined. The *rotate* transitions are defined exactly as in the specification.

- If $0 \leq s < H_{\min}$, $\mu_{rotate}(s, \{Crashed\}) = 1$.
- If $H_{\min} \leq s \leq H_{\max}$, $\mu_{rotate}(s, \cdot)$ is the unique probability measure extension of the following:

$$\mu_{rotate}(s, [a, b]) := \begin{cases} \int_a^b e^{-(s-x)} dx & \text{if } 0 < a \leq b < s \quad \text{i.e. } s - x \sim \text{Exp}(1) \\ 0 & \text{if } s \leq a \leq b \end{cases}$$

$$\mu_{rotate}(s, \{Crashed\}) := \int_{-\infty}^{0} e^{-(s-x)} dx = e^{-s}.$$

Timed transitions happen when the aircraft is rising up to its maximum altitude. The specification says that this is done with a rate between 20 and 25. Hence, if the altitude is $0 \leq s \leq H_{\max} - 25$, the aircraft should end up at an altitude between $s + 20t$ and $s + 25t$ after the time unit t has elapsed. However, this uncertainty is unquantified, which makes it impossible to be modelized as it is in the LMP structure. Hence we have to make a choice on how it will happen. We choose the uniform distribution. Hence,

$$\text{if } 0 \leq s \leq H_{\max} - 25t, \quad \mu_\tau(s, \cdot) := U(s + 20t, s + 25t)$$

For s between $H_{\max} - 25t$ and $H_{\max}$, μ_τ is defined in the obvious way, in order to not exceed the maximal altitude $H_{\max}$. The obtained LMP G^t_{LMP} is depicted in Figure 3. An example of property that can be verified on G^t_{LMP} is **P1**:"if $s >$

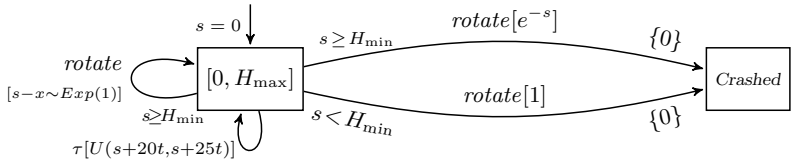

Fig. 3. The aircraft as an LMP

$H_{\min}$, the probability that the aircraft loses 100 meters or more when rotating is between 25% and 50%". The distribution associated to the action *rotate* if $s > H_{\min}$ gives a probability of $p(s, [0, s-100] \cup \{Crashed\}) = e^{-1} - e^{-s} + e^{-s} = e^{-1}$. Hence we conclude that G^t_{LMP} satisfies the property. However, any property that needs an accuracy greater than the one we have chosen with t will not be verified accurately. For example consider the following simple property, **P2**:"if the state is above $H_{\min}$, it takes less than a second to gain 10 meters".

3.2 The aircraft as a rectangular PHA

We now want to model the aircraft by way of a PHA. As the system is a time-continuous system, hybrid systems are perfect for modeling delay transitions.

On the other hand, PHAs allow only discrete probability distributions whereas the aircraft probability distributions are continuous. Hence, the aircraft system can not be modeled faithfully once again. In the PHA model, there will be only one variable, the altitude, hence valuations are just real values. There will be two locations, *Air*, representing the situation where the aircraft can fly and *Crashed*. There is one action, *rotate*. We now define the remaining parameter of our PHA model, that we denote by:

$$G_{\text{PHA}} = (\{x\}, \{Air, Crashed\}, init, \{rotate\}, inv, flow, prob, \langle pre_{Air}, pre_{Crashed}\rangle)$$

- $init(Air) = \{0\}$.
- $inv(Air) = [0, H_{\max}]$ and $inv(Crashed) = \{0\}$.
- $flow(Air) = [0.2, 0.25]$ and $flow(Crashed) = \{0\}$.
- $prob(Crashed, rotate) = \emptyset$, as no action can be done.
- $prob(Air, rotate)$ has to be split in two distributions:
  - μ_1 with $pre(\mu_1) = [0, H_{\min}]$: for all $s \in [0, H_{\min}]$, $\mu_1(Crashed, \{0\}, \{s\}) = 1$, that is, the aircraft ends up in state *Crashed* and s is set to 0.
  - μ_2 with $pre(\mu_2) =]H_{\min}, H_{\max}]$: for all $s \in pre_{Air}(\mu_2)$, the definition of $\mu_2((Air, \mathsf{post}, Y))$ is discussed below.

In defining the distribution μ_2, we have a difficult choice to make. Indeed, to be faithful to the model, we should specify a precondition for every value of x and we would need μ_2 to be non discrete. This is not possible, as we are only allowed a *finite* number of *discrete* distributions. One solution is to partition the interval $]H_{\min}, H_{\max}]$ into as many intervals $I_1, I_2, \ldots, I_n$ as possible, split the *Air* location with respect to these intervals and then, for the transition from I_j to I_k, written $\mu_j(Air, I_k, \emptyset)$, choose a value that would represent the set $\{p(i, I_k) \mid i \in I_j\}$ (where p is the probability defined for the aircraft in the introduction). To simplify, we choose to not split the location for preconditions, but we do consider apart the two post conditions $I_1 := [0, H_{\min}]$ and $I_2 :=]H_{\min}, H_{\max}]$; we must choose values that represent the sets $P_1 := \{p(x, [0, H_{\min}]) \mid x \in [H_{\min}, H_{\max}]\}$ and $P_2 := \{p(x,]H_{\min}, H_{\max}]) \mid x \in [H_{\min}, H_{\max}]\}$. We assign parameter values here of q and q' to simplify, while not making any assumption (one could choose the infima or the mean of the sets P_1 and P_2). In the same way, we assign value q'' to $\mu_2(Crashed, \{0\}, \{x\})$. Figure 4 shows the obtained PHA.

Of course, we are quite far from the exponential distribution that was required in the specification of the aircraft. We now discuss how well this model can be used to check our properties **P1** and **P2**. Recall **P1**:"if $s > H_{\min}$, the probability that the aircraft loses 100 meters or more when rotating is between 25% and 50%". This property could be checked in the LMP version of the aircraft. However, in this model, all we can say about the probability to reach $[0, s - 100] \cup \{Crashed\}$ is that it is greater than the probability to reach $[0, H_{\min}] \cup \{Crashed\} = q' + q''$. Thus we will be able to refute the formula if $q' + q''$ is greater than 50%, but otherwise we cannot say anything. On the other hand, property **P2**: "if the state is above $H_{\min}$, it takes less than a second to gain 10 meters" is verified on G_{PHA}.

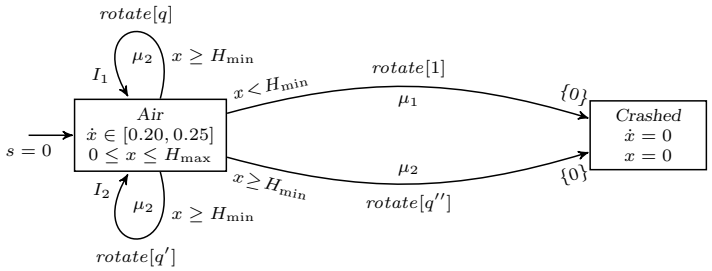

Fig. 4. The aircraft as a PHA: here $I_1 = [0, H_{\min}]$ and $I_2 =]H_{\min}, H_{\max}]$

3.3 Conclusion

We have seen in this case study how to approximate the aircraft example with LMPs and PHAs. We can conclude that neither can faithfully model systems such as the aircraft. There are two main divergences between the two models; they lie in the nature of distributions and in the time class they belong to.

It is important to notice that the failure of PHAs to model faithfully the exponential distribution on our case study is not due to the limitation of rectangular PHAs. Even with general PHA, the *shape* of the definition is inadequate to model continuous distributions. This leads us to highlight a very important observation about LMPs: the continuity of the state space in LMPs crucially depends on the continuous nature of the distribution. Restricting to discrete distribution on any space will result in a discrete process (up to bisimulation).

In the light of those divergences, we propose, in the next section, a generalization that reflects the characteristics of both LMPs and PHAs.

4 Hybrid Labelled Markov Processes

4.1 Definition

Hybrid Labelled Markov Processes (HLMP) combine both Labelled Markov Processes and probabilistic hybrid systems behaviors.

Our starting point is PHAs. We also choose to keep the set of locations finite instead of taking an arbitrary measure space (S, Σ). This is a theoretical restriction that should not prevent further applications and that helps to keep the model tractable. The crucial modification will be in how we will integrate continuous distributions. To do so, it is not sufficient to just permit $prob(v, a)$ to be a finite set of continuous distributions. At first sight, by doing this, the behavior of v would indeed become continuous. However, because this set would only depend on v and the preconditions that we would further attach to this distribution, we would force continuously many valuations, and hence continuously many states, to behave as v does. By doing so, all these states could end up

being bisimilar in the underlying semantics and hence we would be back at our starting point: a discrete distribution. In particular, observe that the exponential distribution in the aircraft example could not be modelled, since it depends on the current value of the altitude and not on the fact that the aircraft is either flying or has crashed.

Consequently, there are two features of a PHA that have to be modified in order to insert continuous distributions to the model: the *prob* and *pre* functions. What is encoded in the precondition function in the PHA will become a parameter of the probabilistic transition function. Hence the subprobability measures that will be returned by *prob* will have the possibility to use this parameter. We also need to define a non standard set of subprobability measures: given a finite set V and a measurable space (S, Σ),

$$\overline{\mathsf{Sub}}(S, V \times \Sigma) := \{\mu \in \mathsf{Sub}(S, V \times \Lambda) \mid \Lambda \text{ is a } \sigma\text{-algebra and } \Lambda \subseteq \Sigma\}.$$

We define an Hybrid Labelled Markov Process as follow:

Definition 4.11 *An Hybrid Labelled Markov Process (HLMP) is a structure* $M = (X, V, init, A, inv, flow, prob, Label)$ *defined as follows:*

- X, V, A *are as in Definition 2.24*
- $init : V \to \mathcal{P}(\mathbb{R}^n)$ *defines an initial set*[3]*.*
- $inv : V \to \mathcal{P}(\mathbb{R}^n)$ *defines invariants for the variables in each location.*
- $flow : V \times \mathbb{R}^n \to \mathcal{P}(\mathbb{R}^n)$ *is a flow evolution condition.*
- $prob : V \times \mathbb{R}^n \times A \to \mathcal{P}_{fin}(\overline{\mathsf{Sub}}(V \times \mathbb{R}^n, V \times \mathcal{B}_n))$ *encodes probabilistic transitions.*
- $Label : V \times \mathbb{R}^n \to \mathcal{P}(AP)$ *is a measurable function used to describe states.*

Let us explain the *prob* function. First observe that in PHAs, there is non determinism coming from a distribution μ if there is a set post of more than one valuations such that $\mu(v, \mathsf{post}, Y) > 0$. It is because we do not have specific probabilities for subsets of post that non deterministic transitions arise. Aside from this, sets of this form can overlap and we have to take account of the multiple copies produced when defining the semantics. This can be done quite easily because the distributions are discrete. With continuous distributions, if post_i are measurable sets, $i = 1, 2$, then the probability of both of them, as well as their union and intersection will be known. Hence, if for some state s and action a the σ-algebra Σ on which $\mu \in prob(s, a)$ is defined is $\mathcal{P}(V) \times \mathcal{B}_n$, then there is no non determinism in μ: every measurable set of this σ-algebra can be split into smaller sets (unless it is a singleton) whose probability will be determined by μ. On the other hand, when μ is defined on a smaller σ-algebra $\Lambda \subseteq \mathcal{P}(V) \times \mathcal{B}_n$, there may be *minimal* sets E (called *atoms* of the σ-algebra), for which $\mu(E)$ is defined, but no set included in E gets a value from μ: this corresponds to the same non determinism that happens in the sets "post" of PHAs.

Depending on the differential equation characterizing the flow evolution, we can distinguish the same kind of sub-classes as for PHAs. For instance, if the

[3] We write it this way for simplicity, but the image is rather $\mathcal{B}_n$.

parameters $x \in X$ evolve with respect to equations $\dot{x} = k$ for $k \in \mathbb{Q}$, then the system would be a *linear HLMP*. Moreover, a rectangular HLMP is obtained if the initialization, the invariants, the flow evolution, the preconditions and the postconditions sets are subsets of $R(X)$.

He now define a semantics for HLMPs, and for this we must redefine the *Target* function. Let $\mu \in prob(v, \mathsf{a}, a)$ be defined on the σ-algebra $V \times \Lambda$ and let $\{\mathsf{E}_i\}_{i \in I}$ be the family of atoms of Λ such that $\mu(v, \mathsf{E}_i) > 0$ for some $v \in V$. This family can be at most countable (because $\mu(V \times \mathbb{R}^n) \leq 1$). We define $Target(\mu)$ as the set of all combinations $\langle \mathsf{b} \rangle := \langle \mathsf{b}_1, \mathsf{b}_2, \ldots, \mathsf{b}_m \rangle$, with $\mathsf{b}_i \in \mathsf{E}_i$. We also need to augment the σ-algebra Λ with these valuations which are not, as singletons, part of it; hence, we define $\Lambda^+ := \Lambda \cup \sigma(\{\{\mathsf{b}\} \mid \exists i \in I . \mathsf{b} \in \mathsf{E}_i\})$[4].

Definition 4.12 *Given a HLMP $H = (X, V, init, A, inv, flow, prob)$, we derive the associated infinite concurrent probabilistic system*

$$(Q, Init, A \cup \mathbb{R}, Steps) \text{ as follows :}$$

- $Q := \{(v, \mathsf{a}) \in V \times \mathbb{R}^n \mid \mathsf{a} \in inv(v)\}$ *is the set of admissible states;*
- $Init = \{(v_0, \mathsf{a}_0) \in Q \mid \mathsf{a}_0 \in init(v_0)\}$ *;*
- $Steps(v, \mathsf{a}) := Cts(v, \mathsf{a}) \cup Dis(v, \mathsf{a})$*, for state $(v, \mathsf{a}) \in Q$, and:*
  - *Cts is defined as in Definition 2.26;*
  - *$Dis(v, \mathsf{a}) \subseteq A \times \mathsf{Sub}(Q)$ contains for each $\mu \in prob(v, \mathsf{a}, a)$ and $\langle \mathsf{b} \rangle \in Target(\mu)$, all pairs $(a, \mu_{\langle \mathsf{b} \rangle})$ such that $\mu_{\langle \mathsf{b} \rangle}$ is defined as, for $\mathsf{C} \in \Lambda^+$:*

$$\mu_{\langle \mathsf{b} \rangle}(v', \mathsf{C}) := \mu(v', \mathsf{C} \setminus \cup_i \mathsf{E}_i) + \sum_{\substack{i \in I \\ \mathsf{b}_i \in \mathsf{C} \cap inv(v')}} \mu(v', \mathsf{E}_i).$$

It is easy to see that HLMPs encompass PHAs. It is only because of the choice of taking the set of locations finite that they do not include LMPs as well. However, they share with LMPs the same intrinsic continuity with respect to distributions.

4.2 Aircraft modeling

We now show that this new framework of HLMP permits to model exactly the aircraft example of Section 1.1. The wanted model is

$$G_{\text{HLMP}} = (\{x\}, \{Air, Crashed\}, init, \{rotate\}, inv, flow, prob)$$

- X, V , $init$, A, inv, $flow$ are the same as G_{PHA}.
- $prob$ is defined as follows
  - if $x \leq H_{\min}$, $prob(Air, x, rotate)$ contains only the distribution that gives probability 1 to $Crashed$ and 0 elsewhere.
  - if $H_{\min} \leq x \leq H_{\max}$, $prob(Air, x, rotate) = \{\mu_x\}$ where $\mu_x(Crashed, 0) = e^{-x}$ and $\mu_x(Air, E) = \mu_{rotate}(E)$ if $E \subseteq \mathbb{R}$, where μ_{rotate} is from the LMP modelisation of the aircraft (Section 3.1).

[4] If W is a set of sets, $\sigma(W)$ is the smallest σ-algebra containing W.

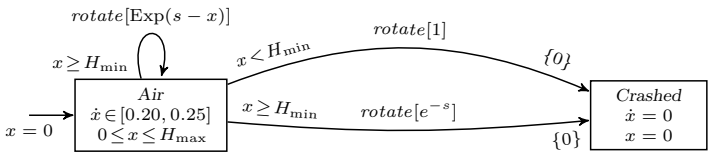

Fig. 5. The HLMP model of the aircraft

This representation combines both the representations as a PHA of Section 3.2 and as the LMP of Section 3.1. Figure 5 shows a graphical representation of this model of the aircraft. In this model, both properties **P1** and **P2** can be verified accurately. Moreover, the following property can also be checked, whereas it could not be checked accurately by neither the LMPs nor the PHAs models: **P3** "if a rotation happens within 20 to 30 seconds, there will be no crashing with probability greater than 1/4". Intuitively, the computation results in the non-empty set $\{(Air, x) \mid x \in [5, 7.5]\}$ of states reachable from the initial state, and therefore **P3** is satisfied by G_{HLMP}.

4.3 Model-checking of HLMP

We discuss how the verification of HLMP can be done. We propose two approaches. Recall that only some sub-classes of hybrid systems (probabilistic or not) can be verified. The simplest approach we propose to verify a HLMP is under implementation and is to construct an LMP from the HLMP by discretizing the time in the same fashion as we did to model the aircraft using LMPs, in Section 3.1. If we have some reasonable unit of time, or if we know in advance the properties to be checked, we can deduce a unit of time t that will make the time discretization a reasonable approximation of the given HLMP with respect to the properties. More precisely, for each state s, only one delay transition will be enabled, a τ-transition of delay t, in the same way as for the aircraft. The result is a narrower process than the one obtained through the semantics of HLMPs where one d-transition for every value $d \in \mathbb{R}$ is defined. The probabilistic part of transitions, needs no transformation. Yet there is one more thing to be precised here, it is that the verification will be faithful to the HLMP if this one contains no non determinism. This requires on one hand that the HLMP be linear, that is, if all flow equations are of the form $\dot{x} = c$ for some real value c: this way, the state reached after the delay t is determined. On the other hand, to not contain non determinism also requires that probabilistic transitions be defined on the total σ-algebra of states. Otherwise, the non determinism will have to be resolved in the construction of the LMP. It will depend on the context to decide if time-discretization is adequate by providing a suitable distribution over the rectangles. Finally, with an LMP in hand, we can use the methodology defined for them. This include finite approximations [3], distances [4,8], and a model

checker [13]. One could wonder why defining HLMPs if we are to further translate them into LMPs. We believe that having a formalism to specify a system exactly is by itself important. Whether there are techniques to verify the models either exactly, approximately, automatically or semi-automatically is another issue. One reason is that an inexact model of the system can be suitable at some point but not later on: with only the approximation at hand, we must start over to build a new one, possibly finer, for another use. Actually, we want this process to be transparent to the user: an automated tool can construct an LMP from an HLMP – and a time unit – in polynomial time. Alternatively, given an HLMP and a property to check, the same tool could choose an appropriate time unit.

The other approach we propose is more involved and needs some more work. HLMPs could be checked by adapting known algorithms for PHAs and LMPs to them. In the probabilistic case, methods to verifiy some sub-classes of PHAs have been proposed [14,15]. More recently, symbolic analysis have been established [9]. These methods, combined with the methodology and tool of Richard [13] should extend to HLMPs. We leave this for future work but we believe that constructing the underlying mathematical framework needed for this will probably consist in combining the existing techniques for PHAs and LMPs.

5 Conclusion

The main purpose of this paper was to compare two models involving continuous state spaces, the continuity arising from different features in each framework. We observed that LMPs' is inherent to the probability distributions of transitions, whereas PHAs' continuity is in the evolution of the model's state. The two models have evolved independently and this paper shows on one hand that they are incomparable, and on the other hand that they are nevertheless compatible, and hence there is a need for a unifying framework. We compared these models through a new case study that permits to highlight their differences and limitations. The aircraft example, despite its simplicity – as only one action and only its altitude are observed – is a continuous process that cannot be modelled faithfully by neither LMPs nor PHAs. Some approximations can be defined, and we exhibited one for each. For both approximations, we also expressed properties that can be verified in one approximation but not in the other.

From the observation that none of the formalisms considered could faithfully model the aircraft, there was only one step to propose a generalization of both to circumvent those limitations. Hence, we defined hybrid LMPs and their semantics. This formalism can model systems that combine both continuous time and continuous distributions. Of course, it can model our aircraft case study. Finally, we proposed verification approaches that use the existing verification techniques of both formalisms. Of course, a lot of work is to be done to extend the theory of LMPs and PHAs to the world of HLMPs.

Acknowledgement
J. Desharnais wishes to thank Marta Kwiatkowska and Oxford University for welcoming her during the current year.

References

1. R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis & S. Yovine (1995): *The algorithmic analysis of hybrid systems. Theoretical Computer Science* 138.
2. R. Blute, J. Desharnais, A. Edalat & P. Panangaden (1997): *Bisimulation for labelled Markov processes. Proc. of the Twelfth IEEE Symposium On Logic In Computer Science, Warsaw, Poland* .
3. A. Bouchard-Cote, N. Ferns, P. Panangaden & D. Precup (2005): *An approximation algorithm for labelled Markov processes: towards realistic approximation.* In: *QEST '05: Proc. of the Second International Conference on the Quantitative Evaluation of Systems*. IEEE Computer Society, Washington, DC, USA, p. 54.
4. F. van Breugel, B. Sharma & J. Worrell (2007): *Approximating a Behavioural Pseudometric Without Discount for Probabilistic Systems.* In: Helmut Seidl, editor: *FoSSaCS*, *LNCS* 4423. Springer, pp. 123–137.
5. S. Cattani, R. Segala, M. Kwiatkowska & G. Norman (2005): *Stochastic transition systems for continuous state spaces and non-determinism.* In: V. Sassone, editor: *Proc. Foundations of Software Science and Computation Structures (FOSSACS'05)*, *LNCS* 3441. Springer Verlag, pp. 125–139.
6. P. R. D'Argenio, N. Wolovick, P. S. Terraf & P. Celayes (2009): *Nondeterministic Labeled Markov Processes: Bisimulations and Logical Characterization. International Conference on Quantitative Evaluation of Systems* 0, pp. 11–20.
7. J. Desharnais, V. Gupta, R. Jagadeesan & P. Panangaden (2000): *Approximating Continuous Markov Processes.* In: *Proc. of the 15th Annual IEEE Symposium On Logic In Computer Science, Santa Barbara, Californie, USA*. pp. 95–106.
8. J. Desharnais, F. Laviolette & S. Zhioua (2006): *Testing Probabilistic Equivalence through Reinforcement Learning.* In: *Proc. of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science*, pp. 664–677.
9. M. Fränzle, H. Hermanns & T. Teige (2008): *Stochastic Satisfiability Modulo Theory: A Novel Technique for the Analysis of Probabilistic Hybrid Systems.* In: *HSCC '08: Proc. of the 11th international workshop on Hybrid Systems*. Springer-Verlag, Berlin, Heidelberg, pp. 172–186.
10. T. A. Henzinger, P.-H. Ho & H. Wong-toi (1996): *Algorithmic Analysis of Nonlinear Hybrid Systems. IEEE Trans. on Automatic Control* 43, pp. 225–238.
11. G. Hassapis & I. Kotini (2006): *Verification of rectangular hybrid automata models. The Journal of Systems and Software* 79, Issue 10.
12. X. Nicollin, A. Olivero, J. Sifakis & S. Yovine (1993): *An Approach to the Description and Analysis of Hybrid Systems. LNCS* 736, pp. 149–178.
13. N. Richard (2003): *Labelled Markov Processes.* Master thesis, Département d'informatique et de génie logiciel, Université Laval.
14. J. Sproston (1999): *Analyzing Subclasses of Probabilistic Hybrid Automata.* In: *Proc. of the 2nd International Workshop on Probabilistic Methods in Verification, Eindhoven*. University of Birmingham, Technical Report, CS-99-8.
15. J. Sproston (2000): *Model Checking of Probabilistic Timed and Hybrid Systems.* Ph.D. thesis, University of Birmingham, Faculty of Science.
16. L. Zhang, Z. She, S. Ratschan, H. Hermanns, & E.M. Hahn, *Safety verification for probabilistic hybrid systems.* In: *Proc. of CAV 2010*. Springer, to appear.

Annexe C

Deuxième contribution

Dans cette annexe, nous joignons un article qui constitue une autre contribution faite au cours de notre maîtrise. Cet article, publié à la conférence *Ninth Workshop on Quantitative Aspects of Programming Languages (QAPL 2011)*, aborde la vérification des automates hybrides probabilistes non-linéaires dont les transitions sont munies de distributions discrètes. Nous avons adapté les deux techniques d'approximations proposées par Henzinger et al [27] pour les automates hybrides non-linéaire et non-probabilistes dans le contexte probabiliste, donnant ainsi lieu à de nouvelles techniques de vérification des automates hybrides non-linéaires probabilistes. Des résultats de vérification des automates hybrides non-linéaires probabilistes peuvent dès lors être déduits grâce à ces techniques.

Analysis of Non-Linear Probabilistic Hybrid Systems

Joseph Assouramou*
Université Laval, Quebec, Canada
Joseph.Assouramou.1@ulaval.ca

Josée Desharnais*
Université Laval, Quebec, Canada
Josee.Desharnais@ift.ulaval.ca

This paper shows how to compute, for probabilistic hybrid systems, the clock approximation and linear phase-portrait approximation that have been proposed for non probabilistic processes by Henzinger et al. The techniques permit to define a rectangular probabilistic process from a non rectangular one, hence allowing the model-checking of any class of systems. Clock approximation, which applies under some restrictions, aims at replacing a non rectangular variable by a clock variable. Linear phase-approximation applies without restriction and yields an approximation that simulates the original process. The conditions that we need for probabilistic processes are the same as those for the classic case.

Keywords: probabilistic hybrid systems, model-checking, linear and rectangular processes, approximation, linearization

1 Introduction

Hybrid processes are a combination of a process that evolves continuously with time and of a discrete component. A typical example is a physical system, such as a heating unit, that is controlled by a monitor. There are discrete changes of modes, like turning on and off the unit, and there is a continuous evolution – the change in temperature. Because of their continuous nature, model-checking hybrid systems can only be done for sub-classes of them. Especially, the largest class for which verification is decidable is the class of rectangular hybrid automata [7, 6]. Another such class for which verification is decidable is that of o-minimal hybrid automata [14], which models hybrid systems whose relevant sets and continuous behavior are definable in an o-minimal structure. In the probabilistic case, Sproston proposed methods to verify $\forall$-PBTL on probabilistic rectangular and o-minimal hybrid processes [11]. Probabilistic timed automata are also a subclass of such processes and have been analyzed extensively [8, 12].

In order to allow the verification of non-rectangular hybrid automata, two translation/approximation methods were proposed by Henzinger et al. [4]: clock-translation and linear phase-portrait approximation. The idea behind those methods is to transfer the verification of any hybrid automaton to the one of a rectangular hybrid automaton which exhibits the same behaviour or over approximates it. In this paper, we show how to apply these methods to probabilistic hybrid processes. We show that both methods apply with the same conditions as for the non deterministic case. The technique of approximation is based on replacing exact values by lower and upper bounds, after splitting the hybrid automaton for more precision in the approximation. Hence, we also show how to split a probabilistic hybrid automaton in order to obtain a bisimilar one. Other side contributions of this paper are: a slightly more general, yet a simpler definition of probabilistic automata than the one proposed by Sproston [11]; and the description, in the next background section, of the two translation techniques in a simpler way than what can be found in [4, 5], mostly because we take advantage of the fact that the definition of hybrid automata has been simplified since then, being presented in terms of functions instead of predicates, and being slightly less general than in the original paper [5].

*Research supported by NSERC

M. Massink and G. Norman (Eds.): 9th Workshop on
Quantitative Aspects of Programming Languages (QAPL 2011)
EPTCS ??, 2011, pp. 1–16, doi:10.4204/EPTCS.??.??

2 Transformation methods for hybrid automata

In this section, we describe the two methods presented by Henzinger et al [4] that will permit the verification of safety properties on any hybrid system.

Let $X = \{x_1, \ldots, x_n\}$ be a finite set of real variables; we write $\dot{X} = \{\dot{x}_1, \ldots, \dot{x}_n\}$ where $\dot{x}_i = \frac{dx_i}{dt}$ is the first derivative of x_i with respect to time. The set of predicates on $\dot{X} \cup X$ is denoted $G(\dot{X} \cup X)$. The set of *valuations* $\mathsf{a} : X \to \mathbb{R}$ is written $\mathbb{R}^X$ or $\mathbb{R}^n$. A set $U \subseteq \mathbb{R}^X$ is *rectangular* if there exists a family of (possibly unbounded) intervals $(I_x)_{x \in X}$ with rational endpoints such that $U = \{\mathsf{a} \in \mathbb{R}^n \mid \mathsf{a}(x) \in I_x \text{ for all } x \in X\}$. We denote by $R(X)$ the set of rectangles over X. For any set Y, we write $\mathscr{P}(Y)$ (resp. $\mathscr{P}_{fin}(Y)$) for the power set of Y (resp. finite power set of Y). For any variable x, belonging to X or not, we write $\mathsf{a}[x \mapsto r]$ for the valuation that maps x to $r \in \mathbb{R}$ and agrees with a elsewhere. Conversely, if $X' \subseteq X$, we write $\mathsf{a}|_{X'}$ for the restriction of a to X'. In the following, we use the notation Set instead of the usual slightly misleading one: Reset.

Definition 1 *[1]* $H = (V, X, \text{Init}, \text{Act}, \text{Inv}, \text{Flow}, E, Pre, \text{Set})$ *is a* hybrid automaton (HA) *if*

- V *is a finite set of locations or control modes;*
- $X = \{x_1, x_2, \ldots, x_n\}$ *is a set of n continuous variables;*
- $\text{Inv} : V \to \mathscr{P}(\mathbb{R}^X)$ *defines invariants for the variables in each location.*
- $\text{Init} : V \to \mathscr{P}(\mathbb{R}^X)$ *defines initial states and satisfies* $\text{Init}(v) \subseteq \text{Inv}(v)$ *for all* $v \in V$.
- Act *is a finite set of actions, possibly including a silent one,* τ*;*
- $\text{Flow} : V \to G(\dot{X} \cup X)$ *is a flow evolution condition;*
- $E \subseteq V \times \text{Act} \times V$ *is a finite set of discrete transitions;*
- $Pre : E \to \mathscr{P}(\mathbb{R}^X)$ *maps to every discrete transition a set of preconditions;*
- $\text{Set} : E \times \mathbb{R}^X \to \mathscr{P}(\mathbb{R}^X)$ *describes change in values of variables resulting from taking edges. We write* $\text{Set}^x(e) := \{\mathsf{d}(x) \mid \exists \mathsf{a} \in \mathbb{R}^X . \mathsf{d} \in \text{Set}(e, \mathsf{a})\}$.

H is said to be rectangular *if the image of* Inv*, Pre and* Set *are included in* $R(X)$ *and* $\text{Flow}(v) = \wedge_{x \in X} \dot{x} \in I_x$ *where each* $I_x \subseteq \mathbb{R}$ *is a (possibly unbounded) interval with rational endpoints.*

The semantics of H is a labelled transition system: the set of states is $S_H := \{(v, \mathsf{a}) \mid \mathsf{a} \in \text{Inv}(v)\}$. There are two kinds of transitions between states: flow transitions and discrete transitions. In a flow transition, the mode of the automaton is fixed and only the variables' values change over time. More formally, there is a flow transition of duration $\sigma \in \mathbb{R}_{\geq 0}$ between states (v, a) and (v, a'), written $(v, \mathsf{a}) \xrightarrow{\sigma} (v, \mathsf{a}')$, if either (1) $\sigma = 0$ and $\mathsf{a} = \mathsf{a}'$ or (2) $\sigma > 0$ and there exists a differentiable function $\gamma : [0; \sigma] \to \mathbb{R}^n$ with $\dot{\gamma} : (0; \sigma) \to \mathbb{R}^n$ such that γ is a solution of $\text{Flow}(v)$ with $\gamma(0) = \mathsf{a}$, $\gamma(\sigma) = \mathsf{a}'$, and $\gamma(\varepsilon) \in \text{Inv}(v)$ for all $\varepsilon \in (0; \sigma)$. For discrete transitions, the control mode of the automaton changes instantaneously. We write $(v, \mathsf{a}) \xrightarrow{a} (v', \mathsf{a}')$, if there exists $e = (v, a, v') \in E$ such that $\mathsf{a} \in Pre(e)$ and $\mathsf{a}' \in \text{Set}(e, \mathsf{a})$.

Example 1 *Figure 1 shows the graphical representation of a thermostat [5] that controls the variation of temperature in a room through a radiator. The whole system has three modes representing that the radiator is either on, off, or down; there is one initial state, where the radiator is on and the temperature has value 2. When the radiator is on the temperature increases with respect to the equation* $\dot{x} = -x + 5$ *whereas it decreases with respect to* $\dot{x} = -x$ *when the radiator is off. When the whole system is down, no variation of the temperature is modeled. The values of the temperature evolve in the range* $[1; 3]$*. The radiator must switch off when it is on and the temperature reaches* 3 *units and on when it is off and the temperature is* 1*. Finally, when we try to turn on the radiator, it might turn on or down.*

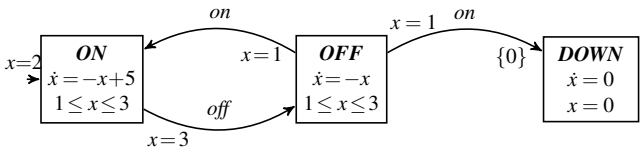

Figure 1: A graphical representation of the thermostat hybrid automaton

Because we need a notion of weak simulation, we define weak transitions through stuttering. Hence, let τ be the (usual) silent action. We write $s \overset{a}{\twoheadrightarrow} s'$ if there exists a finite sequence $s \xrightarrow{\tau} s_1 \xrightarrow{\tau} \ldots \xrightarrow{\tau} s_k \xrightarrow{a} s'$. Similarly, we write $s \overset{\sigma}{\twoheadrightarrow} s'$ if there exists a finite sequence $s \xrightarrow{\sigma_1} s_1 \xrightarrow{\tau} s_2 \xrightarrow{\sigma_2} \ldots \xrightarrow{\tau} s_k \xrightarrow{\sigma_k} s'$ such that $\sum_i \sigma_i = \sigma \in \mathbb{R}_{\geq 0}$. *Simulation* and *bisimulation* are defined on the underlying infinite transition system (and are rather called *time bi/simulation in Henzinger et al. [4]*).

Definition 2 *[4] Let H and H' be two hybrid automata. A relation $\preceq \subseteq S_H \times S_{H'}$ is a* simulation *of H by H' if every initial state of H is related by $\preceq$ to an initial state of H' and if whenever $s \preceq s'$, then for each $a \in \mathrm{Act} \setminus \{\tau\} \cup \mathbb{R}_{\geq 0}$ and each transition $s \overset{a}{\twoheadrightarrow} s_1$, there exists a transition $s' \overset{a}{\twoheadrightarrow} s'_1$ such that $s_1 \preceq s'_1$. If $\preceq^{-1}$ is also a simulation, then $\preceq$ is called a bisimulation. If there is a simulation between H and H' (resp. a bisimulation), we write $H \preceq H'$ (resp. $H \equiv H'$).*

2.1 Clock-translation

The *clock-translation* method is based on the substitution of non-rectangular variables by clocks. Let H be a non-rectangular hybrid automaton. The substitution of a variable x of H by a clock t_x is possible only if, at any time, the value of t_x can be determined by the one of x (i.e., x is solvable).

2.1.1 Preliminaries

We say that a predicate is *simple* if it is a positive boolean combination of predicates of the form $x \sim c$ where $c \in \mathbb{R}$ and $\sim \in \{<, \leq, =, \geq, >\}$. We say that x is *solvable* in H if

- every initial condition, invariant condition, and precondition of H defines simple predicates for x and each flow condition of x in $\mathrm{Flow}(v)$ has the form $(\dot{x} = f_x^v(x)) \wedge P_x$, where P_x is a simple predicate on x; flow evolutions of other variables must not depend on x nor $\dot{x}$;
- the initial-value problem $\dot{y}(t) = f_x^v(y(t))$; $y(0) = c$ has a unique, continuous and strictly monotone solution g_c;
- H is initialised with respect to x. That is, for any transition $e \in E$, x must either stay unchanged in any valuation or get assigned only one value r for all valuations; this will happen if $\mathrm{Set}^x(e)$ is a singleton with the help of the following notation: we will write

$$\mathrm{Set}^x(e) = \{r\} \subseteq \mathbb{R}_* := \mathbb{R} \cup \{*\},$$

where r is either the unique value $r \in \mathbb{R}$, in which case we say that x is reset to r by e, or a special character, $*$, which will represent stability in the value of x. If $r = *$ we must also have that $f_x^v = f_x^{v'}$.

Example 2 *The thermostat automaton of Figure 1 is* solvable *as the flow evolution equation of variable x is solvable in all the modes: in mode ON, the differential equation $\dot{x} = -x+5$ with the initial condition $x(0) = 2$ has the function $x(t) = -3e^{-t}+5$ where $t \in \mathbb{R}_+$ as solution.*

Suppose that $x \in X$ is solvable in the hybrid automaton $H = (V, X, \text{Init}, \text{Act}, \text{Inv}, \text{Flow}, E, Pre, \text{Set})$, and let $c \in \mathbb{R}$ be a constant. We say that c is a *starting value* for a variable x if c is either: the initial value of x in some mode v, that is, $c = \mathsf{a}(x)$ for $\mathsf{a} \in \text{Init}(v)$; or the unique value of $\text{Set}^x(e)$ for some edge $e \in E$ if this value is not $*$. Let $D_v(x)$ be the finite set of starting values of x in v.

Transformation from $x \sim l$ to $t_x \sim' g_c^{-1}(l)$. To simplify the presentation below, we show how predicates on x are transformed into predicates on t_x [4].

Let $g_c(t)$ be the unique solution of the initial-value problem $\dot{y}(t) = f_x^v(y(t)); y(0) = c$, where $c \in \mathbb{R}$. As $g_c(t)$ is strictly monotone, there exists at most one $t \in \mathbb{R}_+$ such that $g_c(t) = l$ for each $l \in \mathbb{R}$. Let $g_c^{-1}(l) = t$ if $g_c(t) = l$ and $g_c^{-1}(l) = -$ if $g_c(t) \neq l$ for all $t \in \mathbb{R}_+$. Let $O := \{<, \leq, =, \geq, >\}$. The transformation from simple atomic predicates over $\{x\}$ to simple atomic predicates over $\{t_x\}$ is the function α_c defined using $\sim \in O$, $lt : O \to O$ and $gt : O \to O$, as follows:

$$\alpha_c(x \sim l) = \begin{cases} \mathit{true} & \text{if } g_c^{-1}(l) = - \text{ and } c \sim l. \\ \mathit{false} & \text{if } g_c^{-1}(l) = - \text{ and } c \nsim l. \\ t_x \; lt(\sim) \; g_c^{-1}(l) & \text{if } g_c^{-1}(l) \neq - \text{ and } c \sim l. \\ t_x \; gt(\sim) \; g_c^{-1}(l) & \text{if } g_c^{-1}(l) \neq - \text{ and } c \nsim l. \end{cases}$$

$\sim$	$lt(\sim)$	$gt(\sim)$
$<$	$<$	$>$
$\leq$	$\leq$	$\geq$
$=$	$=$	$=$
$\geq$	$\leq$	$\geq$
$>$	$<$	$>$

For each (v, c_i) of the hybrid automaton, every predicate $x \sim l$ is replaced by the predicate $\alpha_{c_i}(x \sim l)$, except the invariant predicate which is replaced by $\alpha_{c_i}(x \sim l)$ if $c_i \sim l$, and by *false* otherwise((v, c_i) may be removed in the latter case).

2.1.2 Clock-translation

We are now ready to define clock-translation.

Definition 3 *[4] If $x \in X$ is solvable in $H = (V, X, \text{Init}, \text{Act}, \text{Inv}, \text{Flow}, E, Pre, \text{Set})$, then the clock-translation of H with respect to x is*

$$T = (V_T, X_T, \text{Init}_T, \text{Act}, \text{Inv}_T, \text{Flow}_T, E_T, Pre_T, \text{Set}_T),$$

the hybrid system obtained from the following algorithm:

Step 1: adding the clock t_x.

- *$V_T := \cup_{v \in V} \{v\} \times D_v(x)$, that is, each mode v of H is split. $X_T := X \cup \{t_x\}$.*
- *$\text{Init}_T(v, c) := \{\mathsf{a}[t_x \mapsto 0] \mid \mathsf{a} \in \text{Init}(v)$ and $\mathsf{a}(x) = c\}$.*
- *E_T contains two kinds of control switches; for $c \in D_v(x)$ and $e = (v, a, v') \in E$*

  – if $\text{Set}^x(e) = \{r\} \subseteq \mathbb{R}$, E_T contains the edge $e_T := ((v, c), a, (v', r))$, with $Pre_T(e_T) := Pre(e)$ and $\text{Set}_T(e_T, \mathsf{a}) := \{\mathsf{d}[t_x \mapsto 0] \mid \mathsf{d} \in \text{Set}(e, \mathsf{a}|_X)\}$.

  – if $\text{Set}^x(e) = \{*\}$, E_T contains the edge $e_T := ((v, c), a, (v', c))$ with $Pre_T(e_T) := Pre(e)$ and $\text{Set}_T(e_T, \mathsf{a}) := \{\mathsf{d}[t_x \mapsto \mathsf{a}(t_x)] \mid \mathsf{d} \in \text{Set}(e, \mathsf{a}|_X)\}$.

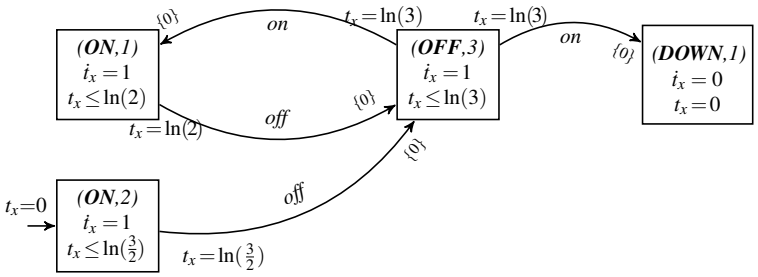

Figure 2: The clock-translation of the thermostat automaton

Step 2: moving to conditions on t_x. We view the images of Init, Inv, *Pre*, Set *and* Flow *as predicates instead of sets of valuations. We replace these predicates over x of the form $x \sim r$ where $\sim \in \{<, \le, =, >, \ge\}$ and $r \in \mathbb{R}$ in T by predicates over t_x of the form $t_x \sim' g_c^{-1}(r)$ as described above. Finally, the variable x can be removed from X_T.*

Example 3 *The timed automaton of Figure 2 is obtained by applying the clock-translation on the thermostat automaton. Each mode v of the automaton is split into $|D_v(x)|$ modes. Since $D_{ON}(x) = \{1,2\}$, we get the modes $(ON,1)$ and $(ON,2)$. For these two modes, the differential equations are respectively "$\dot{x}=-x+5$, where $x(0)=2$" and "$\dot{x}=-x+5$, where $x(0)=1$", and then we have the solutions "$x(t)=-3e^{-t}+5$" and "$x(t)=-4e^{-t}+5$" respectively. Next, we substitute the variable x by the clock t_x in the preconditions, and the invariants. Then, the constraint $x \le 3$ becomes $t \le \ln(2)$ and $t \le \ln(\frac{3}{2})$ respectively. The two automata are bisimilar, by the following theorem.*

Theorem 1 *[4] H is bisimilar to its clock-translation T. The relation is given by the graph of the projection $\eta : S_T \to S_H$, defined as $\eta((v,c),\mathsf{a}) := (v,\mathsf{a}')$, where a' satisfies $\mathsf{a}'|_X = \mathsf{a}|_X$ and $\mathsf{a}'(x) = g_c(\mathsf{a}(t_x))$ where g_c is the solution of the initial-value problem $[\dot{y}(t) = f_x^v(y(t)); y(0) = c]$.*

As a corollary, H and T satisfy the same properties of usual temporal logics.

2.2 Linear phase-portrait approximation

We now present the second method which allows the translation of any hybrid automata into a rectangular one. The linear phase-portrait approximation method can be applied to any hybrid automaton, yielding an *approximation* of the original process which simulates the original automaton (instead of being bisimilar to it, as for clock-translation). This implies that if a safety property is verified on the approximation, then it holds in the original system [5].

The general method is to first split the automaton and then approximate the result. Approximation is done by replacing non-rectangular flow equations by lower and upper bounds on the variables, hence forgetting the true details of the equations. By splitting more finely, one obtains a better approximation.

2.2.1 Splitting a hybrid automaton

Let H be a hybrid automaton with invariant function Inv. A *split function* is a map θ that returns to each mode v of H a finite open cover $\{\mathrm{inv}_1^v, \ldots, \mathrm{inv}_m^v\} \subseteq \mathscr{P}(\mathbb{R}^X)$ of $\mathrm{Inv}(v)$. In splitting, a mode v will be split into several modes according to the cover $\theta(v)$. The fact that $\cup_i \mathrm{inv}_i^v = \mathrm{Inv}(v)$ makes sure that states are preserved whereas the evolution inside mode v is preserved through silent transitions between copies of v, which is possible because $\theta(v)$'s components overlap.

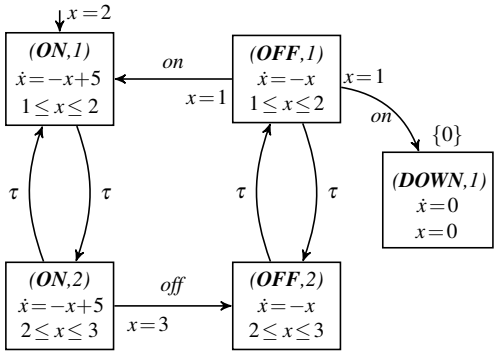

Figure 3: A split of the thermostat

Definition 4 *Let $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, E, Pre, \mathrm{Set})$ be a hybrid automaton. The split of H by θ is the hybrid automaton $\theta(H) = (V_\theta, X, \mathrm{Init}_\theta, \mathrm{Act}_\theta, \mathrm{Inv}_\theta, \mathrm{Flow}_\theta, E_\theta, Pre_\theta, \mathrm{Set}_\theta)$ defined as:*

- $V_\theta = \{(v,i) \mid v \in V \text{ and } 1 \le i \le |\theta(v)|\}$
- $\mathrm{Init}_\theta((v,i)) = \mathrm{Init}(v) \cap \mathrm{inv}_i^v$
- $\mathrm{Act}_\theta = \mathrm{Act} \cup \{\tau\}$
- $\mathrm{Inv}_\theta(v,i) = \mathrm{inv}_i^v$
- $\mathrm{Flow}_\theta(v,i) = \mathrm{Flow}(v)$
- $E_\theta = E_1 \cup E_2$, *where* E_1 *contains the control switch* $((v,i),a,(v',j))$ *for each* $(v,a,v') \in E$, *whereas* $E_2 = \{((v,i),\tau,(v,j)) \mid (v,i),(v,j) \in V_\theta\}$ *allows the automaton to transit silently between the different copies of* v.
- *If* $e_\theta = ((v,i),a,(v',j)) \in E_1$, *we set* $Pre_\theta(e_\theta) = Pre(v,a,v')$ *and* $\mathrm{Set}_\theta(e_\theta, \mathsf{a}) = \mathrm{Set}((v,a,v'), \mathsf{a})$. *If* $e_\theta = ((v,i),\tau,(v,j)) \in E_2$, *we set* $Pre_\theta(e_\theta) = \mathbb{R}^X$ *and* $\mathrm{Set}_\theta(e_\theta, \mathsf{a}) = \{\mathsf{a}\}$.

Note that the cover $\theta(v)$ need not really be open. What is important is that the evolution within any mode be preserved, as pointed out in [4]. This is the case in the following example, where components of the cover are closed and intersect in exactly one point, which is sufficient to allow evolution.

Example 4 *The automaton in Figure 3 is a split of the thermostat automaton with function* $\theta(ON) = \theta(OFF) = \{\{x \mid 1 \le x \le 2\},\ \{x \mid 2 \le x \le 3\}\}$, *and* $\theta(DOWN) = \{\{x \mid x = 0\}\}$. *Note the silent transitions between states* $((ON,i),2)$, $i = 1,2$, *the latter being duplicates of the original thermostat's state* $(ON, \{x \mapsto 2\})$, *that preserve the evolution within mode ON.*

2.2.2 Approximating a hybrid automaton

An (over) approximation of a HA is obtained by weakening all predicates of its evolution.

Definition 5 *Let* $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, E, Pre, \mathrm{Set})$ *be a HA. Another hybrid automaton* $A = (V, X, \mathrm{Init}_A, \mathrm{Act}, \mathrm{Inv}_A, \mathrm{Flow}_A, E, Pre_A, \mathrm{Set}_A)$ *is a* basic approximation *of* H *if:*

- *for all* $v \in V$, $\mathrm{Inv}(v) \Rightarrow \mathrm{Inv}_A(v)$, $\mathrm{Flow}(v) \wedge \mathrm{Inv}(v) \Rightarrow \mathrm{Flow}_A(v) \wedge \mathrm{Inv}_A(v)$, $\mathrm{Init}(v) \Rightarrow \mathrm{Init}_A(v)$;
- *for every discrete transition* $e \in E$, $Pre(e) \Rightarrow Pre_A(e)$ *and* $\mathrm{Set}(e) \Rightarrow \mathrm{Set}_A(e)$;

where sets of valuations are viewed as predicates. If there exists a split θ *on* H *such that* A *is a basic approximation of* H_θ *then* A *is* a phase-portrait approximation *of* H. *If the lower and upper bounds of all the predicates in* A *are rational then* A *is a (rational)* linear *phase-portrait approximation of* H.

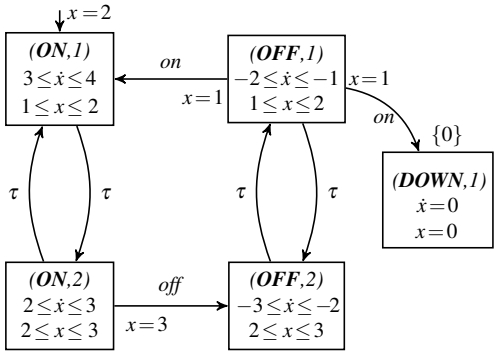

Figure 4: A linear phase-portrait approximation of the thermostat

A straightforward linear phase-portrait approximation is obtained by replacing the invariant in each mode v by a product of rational intervals that contains $\mathrm{Inv}(v)$ and all other predicates, including the flow evolution, by the rational lower and upper bounds implied by the invariant on v.

Example 5 *The automaton of Figure 4 is the linear phase-portrait approximation of the thermostat (with the same split as in Figure 3). Every predicate on $\dot{x}$ is replaced by a predicate that specifies lower and upper bounds on it. For example, the approximation of $\dot{x}$ in mode (ON,1) yields the set $\{\dot{x} \mid 3 \leq \dot{x} \leq 4\}$.*

The following theorem implies that if a safety property is verified for an approximation A, it holds also for H [4, 5].

Theorem 2 *[4] If A is a linear phase-portrait approximation of H, then A simulates H. If it is just a split of H then $A \equiv H$. In both cases, the state $((v,i),\mathsf{a})$ of A is related to (v,a) in H.*

The automaton of Figure 4 simulates the split of the automaton (Figure 3), and then, by transitivity of simulation, simulates the thermostat hybrid automaton.

The verification of initialized rectangular hybrid automata has been widely discussed, particularly in [7] and [9] and it is proved that their verification is decidable. Hence, linear phase-portrait approximation helps to get around the non-decidability of the verification of non-rectangular hybrid automata.

3 Analysis of probabilistic hybrid automata

In this section, we show how the two methods presented above can be used for probabilistic hybrid automata. Our definition of a probabilistic hybrid automaton (PHA) is close to but slightly more general than the one of Sproston [11]. We also add the definition of finitely branching PHAs. A (discrete) probability distribution over a set C is a function $\mu : C \to [0,1]$ such that $\sum_{c \in C} \mu(c) \leq 1$; the support of μ is defined as $\mathrm{supp}(\mu) := \{c \in C \mid \mu(c) > 0\}$, and it is countable. For $U \subseteq C$, we sometimes write $\mu(U) := \sum_{c \in U} \mu(c)$. Let $\mathrm{Dist}(C)$ be the set of all (discrete) distributions over C.

Definition 6 *A tuple $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, prob, \langle pre_{v,a}\rangle_{v \in V, a \in \mathrm{Act}}, \langle pos_{v,a}\rangle_{v \in V, a \in \mathrm{Act}})$ is a* probabilistic hybrid automaton (PHA) *if V, X* Init, Act, Inv *and* Flow *are as in Def. 1 and*

- *$prob : V \times \mathrm{Act} \to \mathscr{P}_{fin}(\mathrm{Dist}(V \times \mathscr{P}(\mathbb{R}^X_*)))$ encodes probabilistic transitions. If $\tau \in \mathrm{Act}$, we require that every $\mu \in prob(v,\tau)$ is concentrated in a unique pair of the form $(v, \{\mathsf{d}\})$.*

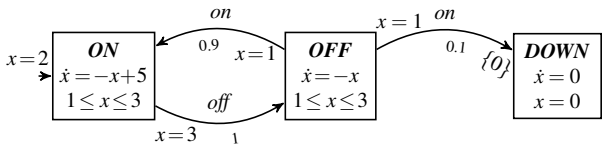

Figure 5: A probabilistic version of the thermostat

- $pre_{v,a} : prob(v,a) \to \mathscr{P}(\mathbb{R}^X)$ *defines preconditions for distributions from* $v \in V$ *and* $a \in$ Act.
- $pos_{v,a} : prob(v,a) \times V \to \mathscr{P}(\mathbb{R}^X)$ *defines postconditions for distributions associated with* $v \in V$ *and* $a \in$ Act.

We say that H is finitely branching *if for every* $v \in V$, $a \in$ Act, $\mu \in prob(v,a)$, μ *is* finitely branching, *that is,* $\mathrm{supp}(\mu)$ *is finite and every set* post *such that* $(v', \mathsf{post}) \in \mathrm{supp}(\mu)$ *is also finite.*

To simplify the notation, we drop the subscripts of *pre* and *pos* when there is no ambiguity.

The semantics of PHAs, is given by probabilistic transition systems. States are defined in the same way. As for non-probabilistic hybrid automata, we distinguish two kinds of transitions in PHAs. Flow transitions are the same, but discrete transitions are now probabilistic and hence defined from a state (v, a) to a distribution. To define transitions, we need some notations on valuations. For $\mathsf{d} \in \mathbb{R}^X_*$, $\mathsf{a} \in \mathbb{R}^X$, $\mathsf{post} \subseteq \mathbb{R}^X_*$, $\mathsf{A} \subseteq \mathbb{R}^X$ and $x \in X$, let

$$\mathsf{d}[\mathsf{a}](x) := \begin{cases} \mathsf{d}(x) & \text{if } \mathsf{d}(x) \neq * \\ \mathsf{a}(x) & \text{if } \mathsf{d}(x) = *, \end{cases} \quad \text{and} \quad \begin{array}{l} \mathsf{post}[\mathsf{A}] := \{\mathsf{d}[\mathsf{a}] \mid \mathsf{d} \in \mathsf{post}, \mathsf{a} \in \mathsf{A}\} \\ \mathsf{post}(x) := \{\mathsf{d}(x) \mid \mathsf{d} \in \mathsf{post}\}. \end{array}$$

Transitions of action a from a state (v, a) in the underlying PTS are as follows. Let $\mu \in prob(v,a)$, $\mathsf{a} \in pre_{v,a}(\mu)$, $\mathrm{supp}(\mu) = \{(v_i, \mathsf{post}_i)\}_{i=1}^m$. Each combination of $\mathsf{d}_i \in \mathsf{post}_i$, $i = 1, \ldots, m$, such that $\mathsf{d}_i[\mathsf{a}] \in pos(\mu, v_i)$, defines a transition

$$(v, \mathsf{a}) \xrightarrow{a} \mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle},$$

where $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ is positive on (arrival) states $(v_i, \mathsf{d}_i[\mathsf{a}])$; the probability that the automaton transits to a state (v', a') is

$$\mu_{\mathsf{a}}{}^{\langle \mathsf{d}_i \rangle}(v', \mathsf{a}') := \begin{cases} \displaystyle\sum_{i=1}^{m} \{\mu(v_i, \mathsf{post}_i) \mid v_i = v',\ \mathsf{d}_i[\mathsf{a}] = \mathsf{a}'\} & \text{if } \mathsf{a}' \in pos(\mu, v') \\ 0 & \text{otherwise.} \end{cases}$$

Example 6 *A probabilistic version of the thermostat is shown in Figure 5. Each discrete transition is labeled by a probability value and an action. Since there is only one variable, a valuation can be represented as a real number. For mode OFF, we have* $prob(OFF, on) = \{\mu\}$ *where* $pre(\mu) = \{1\}$*, such that* $\mu(ON, \{*\}) = 0.9$*, that is, the temperature is unchanged, and* $\mu(DOWN, \{0\}) = 0.1$*. Here, defining* $pos(\mu, -)$ *is not necessary since it is encoded in* μ*. Suppose that in another example one would set* $\mu(ON, [1;3]) = 0.9$*. This would mean that the temperature would end up in the interval* $[1;3]$ *and that the exact value would happen non deterministically. This example would not be finitely branching.*

We now discuss how Definition 6 of PHA slightly differs from previous ones [11]. First note that non deterministic transitions in PHA arise in two ways: when the image of *prob* is not a singleton, and from the possible combinations $\mathsf{d}_i \in \mathsf{post}_i$, $i = 1, \ldots, m$, that we can obtain. Hence, the expression *finitely*

branching is well chosen because every set post, such that $(v,\mathsf{post}) \in \mu$, being finite in the underlying probabilistic transition system, every state (v,a) will have finitely many distributions $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ associated to any action.

The star notation is more general than the reset set of Sproston [11] if there is more than one variable. In the latter, the target of a transition is a distribution over $V \times \mathscr{P}(\mathbb{R}^n) \times \mathscr{P}(X)$, where the third component of a tuple (v,post,X'), called a *reset set*, represents the set of variables that can change value in the transition, with respect to valuations of post. The star notation allows to state, for example, that valuation (x,y) will be modified to $(x,0)$, $(x,2)$, $(0,y)$, $(1,1)$ with probability 1 non deterministically. The corresponding set would be $\mathsf{post}_0 := \{(*,0),(*,2),(0,*),(1,1)\}$. This is an important feature to describe the transitions of the clock-translation (see Section 3.1) and is not possible with the reset set because there is no uniform reset of any variable in post_0. Note in passing that the star notation avoids a third component in the notation and will allow to state very simply the notion of initialised PHA. The use of a postcondition function together with the star notation allows to define distributions on complex sets, such as: $\mathsf{post}_0 \cap ([0;3] \times [1;4])$, the distribution being defined on post_0 and the rectangle being the postcondition. We could not transfer the latter into the distributions by defining μ, for example, as $\mu(v', \mathsf{post}_0 \cap ([0;3] \times [1;4]))$ since post_0 may contain valuations that assign $*$ to a variable which implies that its value depends on the actual state, or valuation for which the distribution μ will be used. Postconditions are necessary for the splitting of PHAs, if one wants to avoid the even more general model that consists in defining probabilistic transitions *prob* from $S_H \times A$ to $\mathscr{P}_{fin}(\mathrm{Dist}(V \times \mathscr{P}(\mathbb{R}^X)))$. This is too general for practical purposes: indeed, any description of a system must be finite and hence states must be described in a parametric (or generic) way.

The condition on τ transitions is to simplify the definition of weak transitions, since it permits to write $s \xrightarrow{\tau} s'$. There may be more than one τ transition from a mode v, but for each $\mu \in prob(v,\tau)$, there is a unique pair $(v',\{\mathsf{d}\})$ such that $\mu(v',\{\mathsf{d}\}) = 1$. Then, following definition of $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ above, there is a τ-transition to state $(v',\mathsf{d}[\mathsf{a}])$ if and only if $\mathsf{d}[\mathsf{a}] \in pos(\mu,v')$ (that is, $\mu_{\mathsf{a}}(v',\mathsf{d}[\mathsf{a}]) = 1$). Weak flow transitions are then defined between states as for hybrid automata, and we write $s \overset{a}{\twoheadrightarrow} \mu$ if there exists a finite sequence of transitions $s \xrightarrow{\tau} s_1 \xrightarrow{\tau} s_2 \ldots \xrightarrow{\tau} s_k \xrightarrow{a} \mu$.

We now define a notion of weak simulation between PHAs. Let $\preceq \subseteq S \times T$ be a relation between two sets S and T. For $X \subseteq S$, we use the notation $\preceq(X) := \{t \in T \mid \exists s \in X . s \preceq t\}$.

Definition 7 *Let H_1, H_2 be two probabilistic hybrid automata. A relation $\preceq \subseteq S_{H_1} \times S_{H_2}$ is a* simulation *if any initial state of H_1 is related to an initial state of H_2 and whenever $s_1 \preceq s_2$, we have:*

- *if $s_1 \overset{a}{\twoheadrightarrow} \mu_1$, for $a \in \Sigma \setminus \{\tau\}$ then $s_2 \overset{a}{\twoheadrightarrow} \mu_2$ and $\mu_1(X) \leq \mu_2(\preceq(X))$ for every X;*
- *if $s_1 \overset{\sigma}{\twoheadrightarrow} s_1'$, for $\sigma \in \mathbb{R}_{\geq 0}$, $s_2 \overset{\sigma}{\twoheadrightarrow} s_2'$ and $s_1' \preceq s_2'$.*

Then we say that H_1 is simulated by H_2, written $H_1 \preceq H_2$. If $\preceq^{-1}$ is also a simulation, it is a bisimulation. Equivalently, an equivalence relation is a bisimulation *if in the condition above we have $\mu_1(X) = \mu_2(X)$ for each equivalence class X.*

This definition is known to be equivalent to the one using weight functions: see Desharnais et al. [3] for a proof that the inequality between μ_1 and μ_2 above is equivalent to the existence of a network flow between them; it is well-known [2], in turn, that the flow condition is equivalent to the existence of a weight function between μ_1 and μ_2.

3.1 Probabilistic clock-translation

In this section, we prove that clock-translation [4], can be applied to a PHA and that it results in a bisimilar PHA, as expected. The general method is to first compute a non probabilistic hybrid automaton from a PHA. Then we apply clock-translation and finally add probabilities in order to obtain a probabilistic clock-translation. There is no condition for computing the underlying non probabilistic HA but we need a notion of solvability so that we will be able to use the clock-translation method thereafter. A variable x of a PHA H is *solvable* if

- the two first conditions of solvability for non probabilistic systems are satisfied
- for every state $s \in S_H$, if $s \xrightarrow{a} \mu$, and $\mu(v', \mathsf{post}) > 0$, then $\mathsf{post}(x) = \{r\}$ for some $r \in \mathbb{R}_*$. Moreover, if $\mathsf{post}(x) = \{*\}$, then we must have $f_x^v = f_x^{v'}$.

Let $H_p = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, prob, \langle pre \rangle, \langle pos \rangle)$ be a PHA. The algorithm has three steps:

Step 1: Define $H{:=}(V, X, \mathrm{Init}, A, \mathrm{Inv}, \mathrm{Flow}, E, Pre, \mathrm{Set})$ the *underlying non probabilistic* HA of H_p as:

- $A := \{\ a_\mu^{\mathsf{post}} \mid \exists v, v' \in V$ such that $\mu \in prob(v,a)$ and $\mu(v', \mathsf{post}) > 0\}$.
- $E := \{(v, a_\mu^{\mathsf{post}}, v') \mid \mu \in prob(v,a)$ and $\mu(v', \mathsf{post}) > 0\}$. Finally, for $e := (v, a_\mu^{\mathsf{post}}, v') \in E$, we set $Pre(e) := pre_{v,a}(\mu)$ and $\mathrm{Set}(e, \mathsf{a}) := \mathsf{post}[\mathsf{a}]$.

Note that solvability of H_p implies that $\mathrm{Set}^x(e) = \{r\} \subseteq \mathbb{R}_*$ and hence H is also solvable.

Step 2: Since H is solvable, let $T = (V_T, X_T, \mathrm{Init}_T, A, \mathrm{Inv}_T, \mathrm{Flow}_T, E_T, Pre_T, \mathrm{Set}_T)$ be the clock-translation of H w.r.t. x. Hence, each transition $e = (v, a_\mu^{\mathsf{post}}, v')$ of H becomes a transition $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', r))$ in T, where $r = c$ if $\mathsf{post}(x) = \{*\}$, otherwise $\mathsf{post}(x) = \{r\}$.

Step 3: Finally, we build $T_p = (V_T, X_T, \mathrm{Init}_T, \mathrm{Act}, \mathrm{Inv}_T, \mathrm{Flow}_T, Prob, \langle Pre \rangle, \langle Pos \rangle)$, the probabilistic clock-translation of H_p from T as follows. Let (v,c) be in V_T, and a in Act. $Prob((v,c),a)$ contains all distributions ν_μ, defined from some $\mu \in prob(v,a)$ as follows:

- for each edge $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', r))$ such that $\mathrm{Set}_T^{t_x}(e_T) = \{0\}$ (i.e., $\mathrm{Set}^x(e) = \{r\}$), let

$$\nu_\mu((v', r), \mathsf{post}[t_x \mapsto 0]) := \mu(v', \mathsf{post});$$

- for each transition $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', c))$ such that $\mathrm{Set}_T^{t_x}(e_T) = \mathrm{Set}^x(e) = \{*\}$, let

$$\nu_\mu((v', c), \mathsf{post}[t_x \mapsto *]) := \mu(v', \mathsf{post}).$$

For both cases, $Pre(\nu_\mu) := Pre_T(e_T)$, and $Pos(\nu_\mu, (v', r)) := \{\mathsf{a} \in \mathbb{R}_*^{X \cup \{t\}} \mid \mathsf{a}|_X \in pos(\mu, v')\}$.

Example 7 *The clock-translation of the probabilistic thermostat automaton is a slight modification of the HA of Figure 2; all transitions get probability* 1 *except for on-transitions from* $(OFF, 3)$ *which have probability* 0.9 *to* $(ON, 1)$ *and* 0.1 *to* $(DOWN, 1)$.

We should now prove that the construction yields a valid PHA: this will be a consequence of the following theorem.

Theorem 3 *If T_p is the clock-translation of H_p, then H_p and T_p are bisimilar.*

Proof. Let H be the underlying non probabilistic automaton of H_p and T its clock-translation. By Theorem 1, H and T are bisimilar through $\eta : S_T \to S_H$ which, being a function, returns a unique state for any state of T. As H_p and H have the same state space, similarly for T and T_p, η can be seen as a function between states of T_p and H_p. We prove that η is a bisimulation between T_p and H_p.

Let $s = ((v,c),\mathsf{a})$ be a state of T_p. There are two kinds of transitions to check in the definition of simulation. For flow transitions, let $s \overset{\sigma}{\twoheadrightarrow} s'$, where $\sigma \in \mathbb{R}_{>0}$. Then since η is a bisimulation between H and T, we obtain $\eta(s) \overset{\sigma}{\twoheadrightarrow} \eta(s')$, as wanted. For discrete transitions, we have to prove that for all $\nu_\mu \in Prob((v,c),a)$ defined from $\mu \in prob(v,a)$, for $\mathsf{a} \in Pre(\nu_\mu)$ and any combination $\langle \mathsf{d}_i \rangle$ from the support of ν_μ, we have $\nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}(\eta^{-1}(U)) = \mu_{\mathsf{a}|_X}^{\langle \mathsf{d}_i|_X \rangle}(U)$ for all $U \subseteq S_{H_p}$. In fact, we need only to prove it for U equals to some state (v',a') since $\{(v',\mathsf{a}')\} \cup \eta^{-1}((v',\mathsf{a}'))$ is an equivalence class.

$$
\begin{aligned}
\nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}(\eta^{-1}((v',\mathsf{a}'))) &= \sum_{\mathsf{b},r} \{\, \nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}((v',r),\mathsf{b}) \mid \underbrace{\mathsf{b}|_X = \mathsf{a}',\ g_r(\mathsf{b}(t_x)) = \mathsf{a}'(x)}_{P(\mathsf{b},\mu)} \,\} \\
&= \sum_{\mathsf{b},r} \sum_{i=1}^{m} \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v', r_i = r,\ \mathsf{d}_i[\mathsf{a}] = \mathsf{b}, \\ \mathsf{post}_i(x) = \{r_i\}, P(\mathsf{b},r_i) \end{array} \,\} \\
&\quad \cup \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v', r_i = r = c,\ \mathsf{d}_i[\mathsf{a}] = \mathsf{b}, \\ \mathsf{post}_i(x) = \{*\}, P(\mathsf{b},c) \end{array} \,\} \\
&= \sum_{i=1}^{m} \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{r_i\},\ \mathsf{b}(t_x) = 0 \end{array} \,\} \\
&\quad \cup \{\, \nu_\mu((v_i,c),\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{*\},\ \mathsf{b}(t_x) = \mathsf{a}(t_x) \end{array} \,\} \\
&= \sum_{i=1}^{m} \{\, \mu(v_i,\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{r_i\} \text{ or } \{*\} \end{array} \,\} \\
&= \mu_{\mathsf{a}|_X}^{\langle \mathsf{d}_i|_X \rangle}(v',\mathsf{a}').
\end{aligned}
$$

In the third equality, the double sum is reduced to a single one because i determines b and r. □

Corollary 1 *The clock-translation of a solvable PHA is a PHA.*

Proof. We only need to prove that every defined ν_μ is a distribution, that is, $\nu_\mu(S_{T_p})$ is 1. We do so by showing that elements of $\mathrm{supp}(\nu_\mu)$ are in bijection with elements of $\mathrm{supp}(\mu)$. By construction, for every $\mathsf{d} \in \mathsf{post}$ such that $\nu_\mu(s,\mathsf{post}) > 0$, we have $\mathsf{d}(t_x) = \{0\}$ if and only if $\mathsf{d}(x) = r$ and $\mathsf{d}(t_x) = *$ if and only if $\mathsf{d}(x) = *$. This implies that $\mathsf{d}|_X = \mathsf{d}'|_X$ if and only if $\mathsf{d} = \mathsf{d}'$, as wanted. Another proof is obtained by taking $U = S_{T_p}$ in the proof of Theorem 3. □

If all variables of H_p are solvable, then its clock-translation with respect to all its variables will yield a probabilistic timed automaton. Knowing that the model-checking of probabilistic timed automata is decidable [10], it implies that the model-checking of solvable PHAs is decidable.

3.2 Probabilistic linear phase-portrait approximation

In this section, we show how to apply the linear phase-approximation method to a probabilistic hybrid automaton, and that it results in a rectangular hybrid automaton which simulates it.

Let $H_p = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, prob, \langle pre \rangle, \langle pos \rangle)$ be a finitely branching PHA. The method of approximation in a probabilistic context follows the same kind of steps as the clock-translation, by going

through an underlying non probabilistic hybrid automaton (this approach is also the one of Zhang et al. [13]). However, this translation is simpler, as well as smaller, here as no condition has to be satisfied by the non probabilistic automaton.

Step 1: We define, $H = (V, X, \text{Init}, A, \text{Inv}, \text{Flow}, E, Pre, \text{Set})$, the *underlying non probabilistic* hybrid automaton of H_p as follows:

- $A := \{a_\mu \mid \exists v \in V \text{ such that } \mu \in prob(v,a)\}$.
- For each $\mu \in prob(v,a)$: E will contain all $e = (v, a_\mu, v')$ such that there is some $(v', \mathsf{post}) \in \text{supp}(\mu)$. $Pre(e) = pre(\mu)$. $\text{Set}(e, \mathsf{a}) = \bigcup_{(v', \mathsf{post}) \in \text{supp}(\mu)} \mathsf{post}[\mathsf{a}] \cap pos(\mu, v')$.

Step 2: We build the linear phase-portrait approximation of H:

$$T = (V_\theta, X_\theta, \text{Init}_\theta, A, \text{Inv}_\theta, \text{Flow}_\theta, E_\theta, Pre_\theta, \text{Set}_\theta).$$

Step 3: Finally, we build

$$T' := (V_\theta, X_\theta, \text{Init}_\theta, \text{Act}, \text{Inv}_\theta, \text{Flow}_\theta, Prob, \langle Pre \rangle, \langle Pos \rangle)$$

the probabilistic linear phase-portrait approximation of H_p from T as follows.

Let (v,i) be in V_θ, and a in Act. $Prob((v,i),a)$ contains two kinds of distributions:

- for each $\mu \in prob(v,a)$ and $e_\theta = ((v,i), a_\mu, (v', j)) \in E_\theta$ and for each post such that $(v', \mathsf{post}) \in \text{supp}(\mu)$, the set $Prob((v,i),a)$ contains all μ_θ defined as

  $$\mu_\theta((v', j), \mathsf{post}) := \mu(v', \mathsf{post}),$$

  with
  - $Pre(\mu_\theta) := Pre_\theta(e_\theta) \cap \text{inv}_i^v$ and
  - $Pos(\mu_\theta, (v', j)) := \overline{\text{inv}}_j^{v'} \cap Pos(\mu, v')$,

  where $\overline{\text{inv}}_j^{v'} := \text{inv}_j^{v'} \setminus (\cup_{k<j} \text{inv}_k^{v'})$ defines a partition of $\text{Inv}(v')$.
- if $a = \tau$, the set $Prob((v,i),a)$ contains in addition all μ_τ^j defined as

  $$\mu_\tau^j((v,j), \{*\}^X) := 1,$$

  that is, the valuation is unchanged during the silent transition from a copy of v to another. These transitions correspond to the edges $((v,i), \tau, (v,j)) \in E_\theta$. There is no special precondition or post-condition, and hence we set $Pre(\mu_\tau^j) := \text{inv}_i^v$ and $Pos(\mu_\tau^j, (v', j)) := \text{inv}_j^{v'}$.

Remark 1 *The use of postconditions, in presence of the star notation, is crucial in the apparently simple definition of μ_θ above, both to make it correct, and to indeed permit a simple and clean formulation. This is on one hand because of the reasons mentioned after Definition 6. On another hand, if we used the less general syntax involving reset sets instead of the star notation in distributions, the preconditions of μ_θ would have to take into account the invariant of the arrival state: if a probability is assigned to a pair that ends up not being valid because of the splitting of transitions, the probability ends up missing, i.e., μ_θ would not sum up to 1: the machinery to overcome this loss of probability would complicate a lot the notation. The use of $\overline{\text{inv}}_j^{v'}$ is also crucial in the definition: it makes sure that we do not use, in*

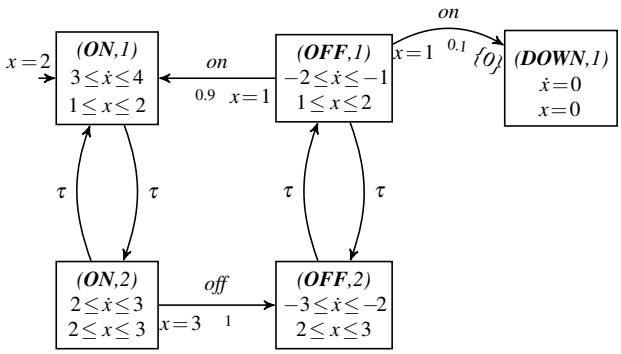

Figure 6: A phase-portrait approximation of the probabilistic thermostat

μ_θ, *a probability value from* μ *more than once. Indeed, some states get duplicated in the split (if some* a *belongs to more than one element of* $\theta(v)$*), which has no negative impact in a non probabilistic HA, since duplicated transitions define bisimilar systems. However, in the probabilistic case, we must make sure that we do not give to both copies the probability value that was meant for one copy: by duplicating the value, we would lose the correspondence between copies and the original mode and we may also end up with a weight of more than one: this would result in a function that is not a distribution. We chose to put the probability on one of the duplicates because silent transitions make sure that the behavior is preserved. We could also have chosen to spread the probability to all duplicates uniformly.*

Example 8 *A linear phase-portrait approximation of the probabilistic thermostat automaton is obtained by slightly modifying the HA of Figure 3 and is illustrated in Figure 6. All transitions get probability* 1 *except for on-transitions from* $(OFF,1)$ *which have probability* 0.9 *to* $(ON,1)$ *and* 0.1 *to* $(DOWN,1)$*. Had we not restricted the postcondition of* $\mu_\theta((ON,j),-)$ *to* $\overline{\mathrm{inv}}_j^{ON}$*, it would give probability one to* $(ON,2)$ *and hence the distribution* $\mu_\theta \in prob((OFF,1),on)$ *would sum up to* 2*.*

We should now prove that the construction yields a valid PHA: this will be a consequence of the following theorem.

Theorem 4 *Any linear phase-approximation* H_θ *of a probabilistic PHA* H_p *simulates it: i.e.,* $H_\theta \preceq H_p$*. The split of a PHA is bisimilar to it.*

Proof. Let R be the relation that relates any $(v,\mathsf{a}) \in S_{H_p}$ to every $((v,i),\mathsf{a})$, with $1 \le i \le |\theta(v)|$. Let $(v,\mathsf{a}) \in S_{H_p}$, $a \in \mathrm{Act}$, $1 \le i \le |\theta(v)|$ and $\mu \in prob(v,a)$. We have to prove that for all $\mathsf{a} \in Pre(\mu_\theta)$ and any combination $\langle \mathsf{d}_j \rangle$ from μ_θ, we have $\mu_{\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(U) \le \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(R(U))$ for all $U \subseteq S_{H_p}$. In fact, we show equality. This does not give us a bisimulation because of the flow transitions which only satisfy simulation. The second equality below relies on the $\overline{\mathrm{inv}}_k^{v'}$'s being disjoint and cumulating to $\mathrm{inv}(v)$: indeed, for each k, there is only one $\mathsf{a}' \in \overline{\mathrm{inv}}_k^{v'}$ and all $\mathsf{a} \in \mathrm{inv}(v)$ is in one of those. Consequently the summation over k can

be inserted freely.

$$\begin{aligned}
\mu_{\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(U) &= \sum_{(v',\mathsf{a}')\in U} \sum_{j=1}^{m} \{\, \mu(v_j, \mathsf{post}_j) \mid v_j = v',\ \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \text{ where } m = |\mathrm{supp}(\mu)| \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in \overline{\mathrm{inv}}_k^{v'}}}^{|\theta(v')|} \sum_{j=1}^{m} \{\, \mu(v_j, \mathsf{post}_j) \mid v_i = v',\ \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in \mathrm{inv}_k^{v'}}}^{|\theta(v')|} \sum_{j=1}^{m} \{\, \mu_\theta((v_j,k), \mathsf{post}_j) \mid v_i = v', \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in \mathrm{inv}_k^{v'}}}^{|\theta(v')|} \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_k \rangle}((v',k),\mathsf{a}') \;=\; \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(R(U))
\end{aligned}$$

This completes the proof of the first claim. The second claim follows easily. □

Example 9 *The linear phase-portrait approximation of the probabilistic thermostat automaton of Figure 6 simulates the thermostat automaton of Figure 5. In particular, consider the states $s_1 = (OFF, 1)$ and $s_2 = (ON, 1)$ of the original thermostat automaton, such that $\mu(ON, 1) = 0.9$ for $\mu \in prob(OFF, on)$. The states ((OFF,1), 1) and ((ON,1),1) simulate respectively s_1 and s_2 since $\mu_\theta((ON, 1), 1) = 0.9$ for $\mu_\theta \in Prob((OFF, 1), on)$.*

Corollary 2 *Phase-portrait approximation and splitting of finitely branching PHAs are PHAs.*

Proof. We only need to prove that every defined $\mu_{\theta\mathsf{a}}$ is a distribution, that is, the total probability out of $\mu_{\theta\mathsf{a}}$ is 1. This is guaranteed by μ_{a} being a distribution and by taking $U := S_\theta$ in the proof of Theorem 4; one obtains that $\mu_{\mathsf{a}}(S_H) = \mu_{\theta\mathsf{a}}(S_\theta)$. Since the distribution has nothing to do with the flow evolution, it is the same argument for both cases of approximation and splitting. □

Since an approximation of a probabilistic hybrid automaton is a rectangular PHA, its model-checking is decidable [11]. Therefore, by taking the right approximation to it, any probabilistic hybrid automaton can be verified.

4 Conclusion

In this paper, we proved that the two methods of Henzinger et al. [4], clock-translation and linear phase-portrait approximation, can also be applied in the probabilistic context to verify non-rectangular PHAs. The adaptation of the methods to PHAs were facilitated by a modification of the syntax over PHAs: a star notation to represent stability of a variable after a transition and postcondition functions. The advantage of adapting the methods instead of defining them from scratch is mainly that proving bi/simulation had only to be checked for discrete transitions. The correctness of the constructions is ensured by bi/simulation relations.

The first method, when it is applicable, results in a probabilistic timed automaton which satisfies exactly the same properties as the original PHA. The inconvenience of this method is that it requires, as in the non-probabilistic case, that the non-rectangular variables be solvable: all the equations induced by the flow evolution have solutions in $\mathbb{R}$.

For linear phase-portrait approximation, there is no restriction on the PHA, and its application results in a rectangular PHA that simulates the original one. When a safety property is satisfied by the rectangular approximation, we can assert that the original PHA satisfies the same property. However, in the case when a safety property is not satisfied by the approximation, more splits should be done; this could be costly in time depending on the property and the size of the PHA.

Side contributions of this paper are also: new additions to the definition of PHA that make them more general; a splitting construction on PHAs that results in a bisimilar PHA; a simpler description of the two techniques than what can be found in the original papers [4, 5]. About the additions to the definition of PHAs, it is interesting to note that the star notation and the postcondition function on distributions permit to represent constraints on variables in terms of set of valuations instead of predicates. When working with distributions, set of valuations are more natural than predicates.

Let us discuss how our approximation relates to the construction of a recent paper by Zhang et al [13], which also defines approximations for probabilistic hybrid automata. That paper also gives a method to over approximate the original automaton. It is clear that the latter is less abstract than the former since Zhang et al. define a *finite* approximation. As in the construction of Henzinger et al. [4], they start from a cover of the state space and abstract according to it. The difference is that the cover is over states instead of over the variables' space of values. More importantly, whereas we use the cover to "linearize" each piece that the cover defines, they group together all these states into one single state. The result is a finite probabilistic transition system, whereas we obtain a PHA. They prove, as we do, that their abstraction simulates the original PHA. However, their approximation is more abstract, which has the advantage of being smaller but of course with less information.

Future work includes the implementations of the technique into a probabilistic model checker and taking advantage of other approximation techniques that have been developed for probabilistic systems in order to widen the class of PHAs for which model-checking is supported.

Acknowledgement

J. Desharnais wishes to thank Marta Kwiatkowska and the Computing Laboratory of Oxford University for welcoming her during year 2009-2010.

References

[1] Rajeev Alur, Thomas A. Henzinger, Gerardo Lafferriere and George J. Pappas, *Discrete abstractions of hybrid systems*, Proceedings of the IEEE, 2000, pp. 971–984.

[2] Christel Baier, *Polynomial time algorithms for testing probabilistic bisimulation and simulation*, Proceedings of the 8th International Conference on Computer Aided Verification (CAV'96), Lecture Notes in Computer Science, no. 1102, 1996, pp. 38–49.

[3] Josée Desharnais, François Laviolette, and Mathieu Tracol, *Approximate analysis of probabilistic processes: logic, simulation and games*, Qest 08, 2008.

[4] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-toi, *Algorithmic analysis of nonlinear hybrid systems*, IEEE Transactions on Automatic Control **43** (1996), 225–238.

[5] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi, *Hytech: A model checker for hybrid systems*, CAV '97: Proceedings of the 9th International Conference on Computer Aided Verification (London, UK), Springer-Verlag, 1997, pp. 460–463.

[6] Thomas A. Henzinger and Peter W. Kopke, *Discrete-time control for rectangular hybrid automata*, Theor. Comput. Sci. **221** (1999), no. 1-2, 369–392.

[7] Isabella Kotini and George Hassapis, *Verification of rectangular hybrid automata models*, Journal of Systems and Software **79** (2006), no. 10, 1433–1443.

[8] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang, *Symbolic model checking for probabilistic timed automata*, Proc. of FORMATS/FTRTFT'04 (Y. Lakhnech and S. Yovine, eds.), LNCS, vol. 3253, Springer, 2004, pp. 293–308.

[9] Jr. Peter William Kopke, *The theory of rectangular hybrid automata*, Ph.D. thesis, Cornell University, Faculty of the Graduate School of Cornell University, 1996.

[10] Jeremy Sproston, *Analyzing subclasses of probabilistic hybrid automata.*, Proceedings of the 2nd International Workshop on Probabilistic Methods in Verification, Eindhoven, University of Birmingham, Technical Report, CS-99-8, August 1999.

[11] Jeremy Sproston, *Decidable model checking of probabilistic hybrid automata*, FTRTFT '00: Proceedings of the 6th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (London, UK), Springer-Verlag, 2000, pp. 31–45.

[12] Jeremy Sproston, *Model checking of probabilistic timed and hybrid systems*, Ph.D. thesis, University of Birmingham, Faculty of Science, 2000.

[13] Lijun Zhang, Zhikun She, Stefan Ratschan, Holger Hermanns, and Ernst Moritz Hahn, *Safety verification for probabilistic hybrid systems*, Proc. of CAV 2010, Springer, 2010, to appear.

[14] G. Lafferriere, G. Pappas, and S. Yovine. *A new class of decidable hybrid systems*, Proc. of HSCC99, LNCS, vol. 1569, Springer-Verlag, 1999, pp. 137151.