Illustration de spécifications temporisées paramétrées sur des signaux continus*

Étienne André[1] ✉, Masaki Waga[2], Natuski Urabe[3] et Ichiro Hasuo[3,4]

Résumé

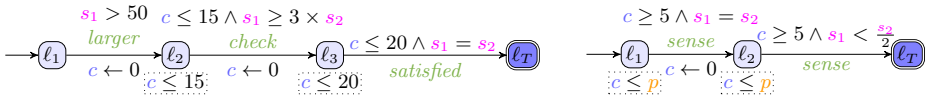
La spécification de propriétés peut s'avérer délicate. Nous proposons ici une approche automatisée pour l'illustration de propriétés données sous forme d'automates étendus avec des contraintes temporelles et des paramètres temporels, et qui peuvent également spécifier des contraintes sur des signaux à valeurs continues. En d'autres termes, étant donné une telle spécification et un automate bornant le comportement admissible de chacun des signaux, notre approche construit des exécutions continues fournissant ainsi des exemples d'exécutions réelles ou impossibles pour la spécification de départ. Dans notre approche, tant la spécification que les automates bornant les comportements admissibles sont donnés sous forme d'une sous-classe des automates hybrides linéaires, plus précisément des automates temporisés étendus par des vitesses d'horloges arbitraires, des contraintes sur les signaux, et des paramètres temporels ; notre méthode génère alors des exécutions concrètes permettant d'illustrer la spécification.

Ce manuscrit court est tiré de l'article récemment accepté [And+22].

1 Introduction

Le *model-checking* a connu de nombreux succès depuis plusieurs décennies (voir par exemple [Kur18]). Néanmoins, son usage dans l'industrie peut être vu comme encore relativement décevant, notamment par rapport aux garanties pourtant très fortes offertes en terme de correction d'un système. C'est d'autant plus vrai pour le model-checking *quantitatif*, qui considère des systèmes étendus avec des quantités telles que les probabilités, le temps, des coûts, etc. Parmi les explications possibles de la pénétration décevante du model-checking dans l'industrie, l'une des raisons est l'expertise poussée demandée par le model-checking aux personnes l'utilisant. Même les personnes expertes dans le domaine sont susceptibles de commettre des erreurs, ce qui donne lieu à des spécifications avec un comportement différent de celui envisagé. Ces erreurs ne pourront ensuite être résolues qu'après une phase de débogage potentiellement fastidieuse.

*Ce travail a bénéficié du soutien du projet ERATO HASUO Metamathematics for Systems Design (N° JPMJER1603), JST et du projet ANR-NRF ProMiS (ANR-19-CE25-0015).



(a) Spécification non paramétrée (b) Spécification paramétrée

FIGURE 1 – Exemples de PTAS

2 Contribution

Nous proposons ici une approche offrant une aide à la spécification, sous forme d'illustration de l'évolution continue de signaux au cours du temps, vérifiant une spécification donnée. Nous introduisons comme formalisme de spécification les automates temporisés paramétrés à signaux (*parametric timed automata with signals*, ou PTAS), une extension des automates temporisés (paramétrés) [AD94 ; AHV93] : nos PTAS bénéficient de la puissance expressive des automates temporisés, où des horloges peuvent être comparées à des constantes, à laquelle nous ajoutons la possibilité de spécifier des contraintes linéaires sur des signaux, telles que « $s_1 \geq 3 \times s_2$ ». Ceci permet d'exprimer des spécifications telles que « lorsque le signal s_1 dépasse 50 alors, en moins de 15 unités de temps, nous avons $s_1 \geq 3 \times s_2$ et ensuite, en moins de 20 unités de temps, les deux signaux sont égaux ($s_1 = s_2$) ». La figure 1a représente le PTAS codant cette spécification (où c est une horloge, et s_1 et s_2 sont des signaux), c.-à-d. que ℓ_T est accessible si et seulement si cette spécification est vérifiée pour une exécution. En outre, nous autorisons des *paramètres temporels* (constantes inconnues), permettant ainsi des spécifications paramétrées combinant des actions discrètes, des contraintes sur les signaux, et des paramètres temporels, telles que « après une première détection à l'aide d'un capteur (action $sense$) se produisant dans un intervalle $[5, p]$, alors nous avons $s_1 = s_2$, puis après une seconde détection dans un intervalle $[5, p]$, nous avons $s_1 < \frac{s_2}{2}$ », où p est un paramètre temporel. Le PTAS codant cette spécification est donné figure 1b. Dans ce cas, notre illustration prendra la forme d'une valeur concrète de p *et* d'une évolution des signaux satisfaisant la spécification.

Afin de borner les comportements possibles des signaux, nous introduisons comme seconde entrée de notre approche des *automates bornant un signal* (*signal bounding automata*, ou SBA). Ces SBA peuvent être déduits d'une connaissance grossière du système considéré ; ils peuvent être également utilisés pour prédéterminer spécifiquement un type de comportement attendu satisfaisant la spécification (par exemple, l'évolution particulière d'un véhicule en cas de freinage, ou au contraire en cas d'accélération). En outre, grâce à nos SBA, nous évitons de générer des exemples d'évolutions de signaux fantaisistes, par exemple avec des changements de valeur arbitrairement rapides. Nos SBA imposent aux signaux de prendre une dérivée arbitraire (mais constante), qui peut évoluer au gré des changements d'états du SBA. Par exemple un SBA pourrait contraindre un signal s à alterner entre une augmentation lente ($\dot{s} = 1$) ou rapide ($\dot{s} = 3$), ou une décélération lente ($\dot{s} = -1$) ou rapide ($\dot{s} = -3$) ; ce SBA est donné figure 2.

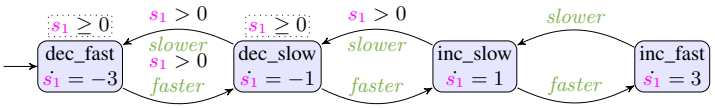


FIGURE 2 – Un exemple de SBA

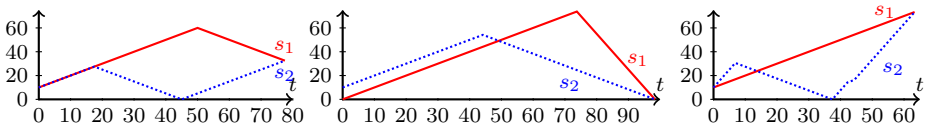


FIGURE 3 – Exemples d'exécutions pour les figures 1a et 2

Nous générons non seulement des exemples d'exécutions positives (correctes), mais aussi négatives (incorrectes, qui ne satisfont *pas* la spécification). En effet, notre paradigme est que, afin d'illustrer une spécification, il peut être nécessaire de produire à la fois des exemples positifs et des exemples négatifs, si possible proches de la « frontière » de la satisfiabilité.

Exemple 1. Soit $\mathcal{A}$ le PTAS figure 1a ; soit $\mathcal{A}_1$ le SBA figure 2, et $\mathcal{A}_2$ le SBA figure 2 où s_1 est remplacé par s_2. Nous supposons que, initialement, $s_1, s_2 \in [0, 10]$. Étant donné le PTAS $\mathcal{A}$ et les deux SBA $\mathcal{A}_1$ et $\mathcal{A}_2$ bornant le comportement de s_1 et s_2, notre approche génère automatiquement plusieurs évolutions possibles des signaux satisfaisant la spécification ; nous donnons trois de ces exécutions en figure 3. Notons que ces trois exécutions représentent trois évolutions très différentes des signaux, avec des valeurs initiales et finales, et des vitesses d'évolution, toutes très différentes.

3 Algorithmes et implémentation

Notre approche résumée en figure 4 se base sur le formalisme des PTAS et des SBA, eux-mêmes définis comme une sous-classe d'un nouveau formalisme dé-

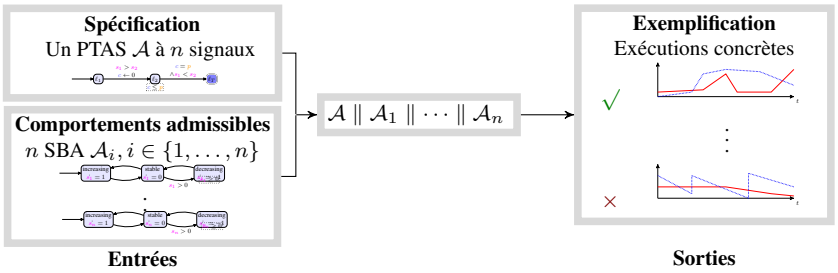


FIGURE 4 – Notre approche

fini comme une sous-classe des automates hybrides rectangulaires [Hen96]. Nous avons défini pour notre nouveau formalisme une syntaxe et une sémantique formelles, et nous avons proposé des algorithmes basés sur une représentation symbolique de l'espace d'états, qui peut être vue comme une extension du graphe des zones tel que défini pour les automates temporisés (paramétrés) [AD94 ; AHV93]. Nos algorithmes proposent d'une part des exécutions concrètes à partir de cette représentation symbolique et, d'autre part (grâce à des heuristiques), des exécutions négatives (impossibles) — sous certaines hypothèses, notamment de déterminisme. Les détails sont disponibles dans [And+22].

Nous avons implémenté nos algorithmes au sein du logiciel IMITATOR [And21] (`v.3.3-alpha` « *Cheese Caramel au beurre salé* »), et avons appliqué notre approche à plusieurs exemples.

Perspectives Les perspectives incluent une extension à des logiques telles que MITL ou STL, mais aussi des garanties de couverture des cas limites.

Références

[AD94] Rajeev ALUR et David L. DILL. "A theory of timed automata". In : *TCS* 126.2 (avr. 1994), p. 183-235. DOI : 10.1016/0304-3975(94)90010-8.

[AHV93] Rajeev ALUR, Thomas A. HENZINGER et Moshe Y. VARDI. "Parametric real-time reasoning". In : *STOC* (16-18 mai 1993). Sous la dir. de S. Rao KOSARAJU, David S. JOHNSON et Alok AGGARWAL. San Diego, California, United States : ACM, 1993, p. 592-601. DOI : 10.1145/167088.167242.

[And+22] Étienne ANDRÉ, Masaki WAGA, Natsuki URABE et Ichiro HASUO. "Exemplifying parametric timed specifications over signals with bounded behavior". In : *NFM* (24-27 mai 2022). Sous la dir. de Klaus HAVELUND, Jyo DESHMUKH et Ivan PEREZ. T. 13260. LNCS. Caltech, Pasadena, CA, USA : Springer, 2022. DOI : 10.1007/978-3-031-06773-0_25.

[And21] Étienne ANDRÉ. "IMITATOR 3 : Synthesis of timing parameters beyond decidability". In : *CAV* (18-23 juil. 2021). Sous la dir. de Rustan LEINO et Alexandra SILVA. T. 12759. LNCS. virtual : Springer, 2021, p. 1-14. DOI : 10.1007/978-3-030-81685-8_26.

[Hen96] Thomas A. HENZINGER. "The Theory of Hybrid Automata". In : *LiCS* (27-30 juil. 1996). Sous la dir. de Moshe Y. VARDI et Edmund M. CLARKE. New Brunswick, New Jersey, USA : IEEE Computer Society, 1996, p. 278-292. DOI : 10.1109/LICS.1996.561342.

[Kur18] Robert P. KURSHAN. "Transfer of Model Checking to Industrial Practice". In : *Handbook of Model Checking*. Sous la dir. d'Edmund M. CLARKE, Thomas A. HENZINGER, Helmut VEITH et Roderick BLOEM. Springer, 2018, p. 763-793. DOI : 10.1007/978-3-319-10575-8_23.