Université Joseph Fourier
U.F.R Informatique & Mathématiques Appliquées

DEA INFORMATIQUE: SYSTÈMES ET COMMUNICATIONS

SIMULATION DES SYSTÈMES CONTINUS OUVERTS

APPLICATION À LA SYNTHÈSE

Réalisé par
Olfa BEN SIK ALI

Encadré par
Oded Maler

Date : 16 *Juin* 2003

Remerciements

Je tiens à exprimer ma gratitude envers toutes les personnes qui ont contribué de près ou de loin au bon déroulement de mon projet de DEA. Je remercie :

- *M. Oded Maler*, directeur de chercheur à VERIMAG, qui m'a encadré durant ce projet et avec qui travailler est un réel plaisir;
- *M. Evgueni Asarin*, chercheur à VERIMAG pour ses précieux conseils concernant les aspects théoriques du projet;
- *Mme Thao Dang*, chercheur à VERIMAG, pour m'avoir éclairé sur certains points inhérants au projet et pour avoir réviser ce rapport.

Resumé

Ce projet étudie une nouvelle approche pour l'analyse des systèmes continus ouverts. Nous nous intéressons à l'application des techniques de la simulation pour l'étude de l'atteignabilité, la vérification et la synthèse.
Dans une première partie, nous présentons les systèmes hybrides et la difficulté de leurs résolution qui est dûe à la complexité de leurs dynamiques continues. Nous nous intéressons, par la suite aux systèmes continus ouverts. Nous proposons une technique basée sur la discrétisation de l'axe du temps et des entrées admissibles et donc, l'exploration des trajectoires induites par un ensemble d'entrées discrétisées. Nous formalisons les problèmes de verifiacation et de la synthèse dans ce cadre. Comme une application, nous devellopons des algorithmes pour trouver des signaux de commande qui dirigent un système vers un état but tout en respectant des contraintes.

Abstract

We study in this report a new approach for the analysis of open continuous systems. We are interested on the application of simulation techniques to solve atteignability, verification and synthesis problems.
First, we present hybrid systems and notice the difficulty of their resolution which is due to the complexity of their continuous dynamics. Then, we are interested on open continuous systems. We propose a technique based on the discretization of the time and the set of possible input signals and thus, the exploration of trajectories induced by the discretized inputs. We formalize verification and synthesis problems in this context. As application, we develop algorithm to check control signal which make the system reach a set of goal state while avoiding a set of bad states.

Table des matières

Table des figures

Chapitre 1

Introduction

1.1 Contexte du projet

L'utilisation des régulateurs numériques a nettement amélioré le niveau d'automatisation des systèmes industriels. Un exemple typique est une usine de production chimique par lots où les ordinateurs sont utilisés pour ordonner les séquences des réactions chimiques, dont chacune est modelisée comme un procédé continu. L'intégration de tels contrôleurs digitaux donne lieu à des systèmes complexes qui combinent des dynamiques continues et discretes qu'on désigne par les *systèmes hybrides*. Ils sont utilisés dans divers applications, telles que la production chimique, les systèmes de contrôle du trafic aérien et la robotique.

Dans la conception des systèmes, la vérification formelle et la synthèse de contrôleurs sont deux problèmes importants. Le but de la *vérification formelle* est de montrer que le système s'exécute comme prévu. La *synthèse des contrôleurs* s'intéresse à la conception des contrôleurs pour que le système soit conforme à une spécification donnée.

Pour vérifier et synthétiser des systèmes hybrides, il faut au préalable disposer de méthodes d'analyse rigoureuses permettant de caractériser l'ensemble des comportements possibles du système.

De manière générale les systèmes hybrides sont souvent complexes et il n'existe pas de méthode exacte de résolution des problèmes de vérification et de la synthèse, sauf pour des cas particuliers des systèmes à dynamiques continues linéaires [11].
Plusieurs techniques approximatives ont été développées. Ces techniques sont basées sur l'idée d'approcher l'espace des états atteignables par des ellipsoïdes ou par des polyhèdres [5], [3]. Elles s'intérèssent essentiellement à la résolution de l'atteignabilité et de la vérification. L'application à la synthèse se limite généralement à la conception des *contrôleurs discrets* dont le rôle est de décider du changement de mode du système, par commutation d'un état discret à un autre, pour éviter des mauvais états [1].
Dans la pratique, les systèmes sont souvent commandés par des signaux continus qui règlent leurs comportements afin d'éviter certains dysfonctionnements ou de les piloter vers des états prédéfinis. En d'autres termes, ces signaux affectent la dynamique continue de façon

à corriger les trajectoires. Le problème majeur revient donc aux choix de ces signaux et de l'instant de leur application.

1.2 But du projet

Dans ce projet il est question d'étudier la dynamique des systèmes continus ouverts. Nous nous intéressons essentiellement à ce type de système car nous estimons qu'ils présentent la difficulté majeure dans la résolution de la synthèse des systèmes hybrides. Le projet contient une partie théorique et une autre pratique.

La partie théorique commence par l'étude des systèmes hybrides en général, et les systèmes continus ouverts en particulier. La complexité de ces systèmes et le coût des méthodes analytiques exactes nous motive à adapter les techniques de la simulation numérique pour l'analyse de l'atteignabilité, la vérification et la synthèse.
La simulation va se baser sur le choix d'un sous ensemble discret de l'axe temporel et d'un ensemble assez grand de signaux discrétisés. La génération des trajectoires s'effectue successivement à travers des intégrations numériques.

Nous avons développé par la suite différents algorithmes pour appliquer cette technique à la synthèse. La synthèse, dans notre contexte, consiste à trouver un ensemble de commande qui permettent de piloter le système d'un ensemble de départ à un ensemble d'états finaux tout un respectant certaines contraintes.
Dans la partie pratique, nous avons implémenté les algoritmes développées sous *Matlab* que nous avons appliqué sur des problèmes de synthèse.

1.3 Organisation du document

Le présent rapport détaille le travail réalisé durant mon stage de DEA.
Dans le *chapitre 2* nous introduisons les systèmes hybrides et leurs modélisations. Nous explicitons, en particulier les notions de la vérification et de la synthèse et nous constatons leurs complixités.
Le *chapitre 3* s'intéresse à l'analyse de la dynamique des systèmes continus ouverts. Nous présentons la méthode adoptée pour la discrétisation et l'exploration des trajectoires discrétisées. Nous expliquons par la suite les algorithmes développés et nous donnons des exemples concrets de leurs applications.

Chapitre 2

Les Systèmes Hybrides

L'analyse des systèmes hybrides exige un modèle suffisament riche pour décrire des dynamiques continues et discrètes. Les systèmes dynamiques continus sont traditionnellement modélisés par des équations différentielles et les systèmes à évènements discrets par des automates. Les automates hybrides sont un formalisme qui combine ces deux modèles.

2.1 Exemple introductif

Tout d'abord nous allons prendre un exemple pour expliquer et introduire les concepts de base ainsi que la notation graphique des systèmes hybrides. L'exemple que nous considérons est un thermostat utilisé pour contrôler la température d'une pièce. Le thermostat est composé d'un réchauffeur et d'un thermomètre. Le seuil maximal et le seuil minimal de la température sont fixées respectivement à θ_M et θ_m.
Le réchauffeur fonctionne (état "on") tant que la température de la pièce est inférieure à θ_M et il s'arrête (état "off") dés que la température atteint θ_M. Réciproquement, le réchauffeur est maintenu à l'état "off" aussi longtemps que la température est supérieur à θ_m, il bascule à l'état "on" dés que la température atteint cette limite.
Nous pouvons considérer le thermostat comme un système combinant des dynamiques continues dont les états sont définis par la température x, et des dynamiques discrètes représentées par les modes du thermostat qui prennent les valeurs "on" et "off".
L'évolution de la température est décrite par les équations suivantes:

$$\dot{x} = \begin{cases} f_1(x) = -x + 4 & \text{état "on"} \\ f_2(x) = -x & \text{état "off"} \end{cases}$$

Une description graphique du système est donnée par la figure 2.1.
Un exemple de problème de vérification est de montrer que la température reste toujours dans une marge désirée $[m, M]$.
Pour cet exemple, si on considère un état initial $x(0) = \theta_0$, la solution des équations diffé-

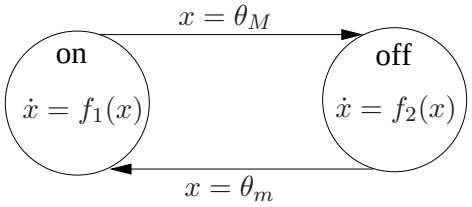


FIG. 2.1 – *Modèle du thermostat.*

rentielles est respectivement $x(t) = \theta_0 e^{-t} + 4(1 - e^{-t})$ et $x(t) = \theta_0 e^{-t}$. La trajectoire de la température est illustrée par la figure 2.2.

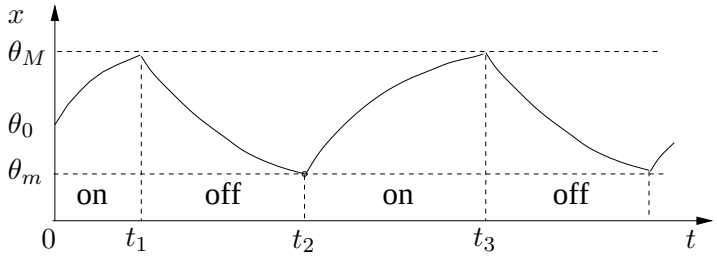


FIG. 2.2 – *Une trajectoire de la température.*

On considère maintenant que la structure du thermostat est fixe mais qu'on peux régler librement les seuils θ_m et θ_M afin de maintenir la température dans la marge désirée. Dans ce cas le problème devient un problème de synthèse de contrôleurs: comment choisir les règles qui génèrent les commutations entre les deux modes pour atteindre l'objectif fixé.

2.2 Modélisation des systèmes hybrides

Avant de donner la définition des automates hybrides, nous présentons les systèmes à dynamiques continues.

2.2.1 Systèmes à dynamiques continues

Définition 1 (Système à dynamique continue) *Un système à dynamique continue est un tuplé $\mathcal{C} = (\mathcal{X}, f)$ où*

- $\mathcal{X} = \mathbb{R}^n$ *est l'espace d'état.*
- $f : \mathcal{X} \to \mathbb{R}^n$ *est un champ vectoriel continu.*

Les comportements du système sont déterminés par l'équation différentielle suivante :

$$\dot{\mathbf{x}} = f(\mathbf{x}) \tag{2.1}$$

Définition 2 (Trajectoires d'un système à dynamique continue)
Soit $\mathcal{T} = \mathbb{R}^+$ *l'* axe temporel
Une trajectoire de $\mathcal{C}$*, à partir d'un point initial* $\mathbf{x} \in \mathcal{X}$*, est un comportement continu* $\xi_{\mathbf{x}} : \mathcal{T} \to \mathcal{X}$ *tel que* $\xi_{\mathbf{x}}$ *est la solution de (2.1) sous la contrainte* $\mathbf{x}(0) = \mathbf{x}$*.*

2.2.2 Automates hybrides

Syntaxe

Définition 3 (Automate hybride) *Un automate hybride est un tuple* $\mathcal{A} = (\mathcal{X}, Q, f, H, G, R)$ *où*

- $\mathcal{X} \subseteq \mathbb{R}^n$ *est l'espace des états continus. Les variables continues sont notées par* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$*.*
- Q *est un ensemble fini d'états discrets.*
- $f : Q \to (\mathcal{X} \to \mathbb{R}^n)$ *à un état discret* q*, l'évolution des variables continues est décrite par l'équation différentielle* $\dot{\mathbf{x}} = f(q)$*.*
- $H : Q \to 2^{\mathcal{X}}$ *défini les invariants des états discrets.*
- $G : (Q \times Q) \to 2^{\mathcal{X}}$ *sont Les gardes qui déterminent les conditions de commutation d'un état discret à un autre*
- $R : (Q \times Q) \to (\mathcal{X} \to 2^{\mathcal{X}})$ *associe à chaque transition une fonction à valeurs multiples.* $R(q, q')$ *définit comment les variables continues changent lorsque la transition de* q *à* q' *est effectuée.*

Dans la définition 3, l'ensemble des états discrets Q représente les "modes" continus du système. Le système peut rester dans l'état q tant que $H(q)$ est vérifiée par x. Il évolue selon la dynamique $f(q)$. Dés qu'il atteint un point $\mathbf{x}' \in G(q, q')$ la transition de q à q' est activée. Dans ce cas, le système peut commuter à l'état q'.
L'automate hybride $\mathcal{A}$ est souvent représenté graphiquement par un graphe orienté dont les sommets sont les états discrets et les arcs représentent les transitions. Les invariant et les équations différentielles sont associés aux sommets. Les gardes sont associées aux arcs.

pour simplifier, nous utiliserons la notation f_q pour $f(q)$, $G_{qq'}$ pour $G(q, q')$, G_q pour $\bigcup_{q'} G_{qq'}$, H_q pour $H(q)$, H pour $\bigcup_q H_q$ et $R_{qq'}$ pour $R(q, q')$.

Sémantique

Nous étudions les comportements de l'automate hybride $\mathcal{A}$ dans l'espace d'état hybride $\mathcal{X} \times Q$. Un état $(q, \mathbf{x})$ de $\mathcal{A}$ varie:

- Par *évolution continue*: les variables continues évoluent selon la dynamique f_q, l'état discret q reste constant.
- Par *évolution discrète*: le système change d'état discret en franchissant une transition.

Soit $\alpha : \mathcal{T} \to \mathcal{X}$ la solution de $\dot{\mathbf{x}} = f_q(\mathbf{x})$ sous la condition initiale $\alpha(0) = \mathbf{x}$. L'état $(q, \mathbf{x}')$ est atteignable de $(q, \mathbf{x})$ par la dynamique continue f_q s'il existe $t < \infty$ tel que $\alpha(t) = \mathbf{x}'$ et $\alpha(t') \in H_q$ pour chaque $0 \leq t' \leq t$. Dans ce cas, $\mathbf{x}'$ est dit *q-atteignable* de $\mathbf{x}$, qu'on note par $\mathbf{x} \xrightarrow{q,t} \mathbf{x}'$.

Définition 4 (Trajectoires d'un automate hybride) *Une trajectoire d'un automate hybride $\mathcal{A}$, à partir d'un état $(q_0, \mathbf{x}_0)$, est une paire $\gamma = (\alpha, \beta)$ où:*

- *$\alpha : \mathcal{T} \to \mathcal{X}$ est continue par morceaux;*
- *$\beta : \mathcal{T} \to Q$ est constante par morceaux c'est à dire, il existe une séquence temporel $\mathcal{J}(\beta) = 0, t_1, t_2, \ldots$ telle que pour tout $k \in \mathbb{N}$, β est constante sur l'intervalle $I_k = [t_k, t_{k+1})$. La séquence de ces intervalles est notée $\mathcal{I}(\beta)$.*

Une trajectoire doit satisfaire les conditions suivantes:

1. *Conditions initiales : $\alpha(0) = \mathbf{x}_0$ et $\beta(0) = q_0$.*
2. *Evolution continue : pour tout intervalle $I_k = [t_k, t_{k+1}) \subseteq \mathcal{I}(\beta)$ tel que $\beta(k) = q$, $\alpha(t_k) \xrightarrow{q,t} \alpha(t_k + t)$ pour tout $t \in [0, t_{k+1} - t_k)$.*
3. *Condition de transition : une transition est possible à $t_k \in \mathcal{J}(\beta)$ tel que $\beta(k-1) = q$ et $\beta(k) = q'$ si $\alpha(t_k^+) \in G_{qq'}$, où $\alpha(t_k^+)$ est la limite à droite de α à t_k.*

2.3 Notion d'atteignabilité

Notre approche de vérification et de contrôle se base sur l'exploration de l'ensemble des états atteignables des systèmes.
L'ensemble atteignable par un automate hybride $\mathcal{A}$, à partir d'un ensemble F, est défini comme l'ensemble des états visités par les trajectoires dont les états initiaux sont F. Une trajectoire dans l'espace des états continus est la séquence des trajectoires induites par la dynamique continue. Nous considérons ici l'atteignabilité de la dynamique continue.

Les ingrédients principaux pour déterminer les trajectoires sont les opérateurs successeurs et prédécesseurs des états. Nous nous intéressons ici au calcul de l'atteignabilité en avant, et donc à l'opérateur *successeur*. Soit $\mathcal{C}$ un système continu (définition 1).

Définition 5 (Successeurs) *L'opérateur* $\delta : 2^X \to 2^X$ *est défini pour un sous-ensemble* F *de* $\mathcal{X}$ *et un intervalle temporel* $I \subseteq \mathcal{T}$ *comme suit :*

$$\delta_I(F) = \{\mathbf{x}' \mid \exists \mathbf{x} \in F \ \exists t \in I \ : \mathbf{x} \xrightarrow{t} \mathbf{x}'\}.$$

$\mathbf{x} \xrightarrow{t} \mathbf{x}'$ indique que $\mathbf{x}' = \xi_{\mathbf{x}}(t)$ où $\xi_{\mathbf{x}}$ est la trajectoire de $\mathcal{C}$ partant du point $\mathbf{x}$.

L'algorithme standard de résolution de l' atteignabilité en avant à partir d'un ensemble d'états F est:

Algorithme 1

$R_0 := \mathcal{F}$
Répéter
  $R_{i+1} := \delta(R_i)$
jusqu'à
  $R_{i+1} := R_i$

2.4 Principe de la vérification

Nous nous intéressons à la *méthodologie algorithmique* de la vérification et à des *propriétés de sûreté*. Les propriétés de sûreté peuvent être exprimées comme suit : aucune trajectoire du système ne devrait jamais atteindre un certain ensemble d'états P, autrement dit, le système restera toujours dans le complément de P. Le problème de vérification pour tout système discret fini peut être résolu en utilisant l'analyse d'atteignabilité en avant car la fonction de transition, l'ensemble initial F, l'ensemble P et l'ensemble des états atteignables accumulés au cours du calcul sont finis et peuvent être représentés explicitement.

L'application de cette approche aux automates hybrides concerne le calcul des fonctions sur les ensembles suivantes :

- Successeur ou prédécesseur : $Q \times 2^{\mathcal{X}} \to Q \times 2^{\mathcal{X}}$
- Union et intersection : $2^{\mathcal{X}} \times 2^{\mathcal{X}} \to 2^{\mathcal{X}}$
- Test d'équivalence : $2^{\mathcal{X}} \times 2^{\mathcal{X}} \to \{0, 1\}$

Pour calculer ces fonctions, le premier ingrédient dont nous avons besoin est une représentation syntaxique finie des ensembles produits pendant l'exécution des algorithmes. L'espace d'état continu $\mathcal{X}$ des automates hybrides est dans $\mathbb{R}^n$, et par conséquent les sous-ensembles

de $\mathcal{X}$ n'admettent pas une représentation énumérative et peuvent seulement être *symboliquement* représentés, par exemple, par des formules logiques.

Une autre difficulté est liée à l'évolution biphasée des systèmes hybrides, qui exige la capacité de calculer les successeurs non seulement par des transitions discrètes mais aussi par des dynamiques continues. Dans la phase continue, ce problème se ramène au problème de caractériser les trajectoires des systèmes continus.
Nous illustrons cette difficulté à l'aide d'un automate hybride avec un seul état discret dont l'invariant est tout l'espace d'états. Supposons que l'ensemble initial F peut être caractérisé par une formule $\phi_F(\mathbf{x})$ dont la valeur de vérité est 1 ssi $\mathbf{x} \in F$, et de la même manière, l'ensemble P est caractérisé par une formule $\phi_P(\mathbf{x})$. Supposons également que l'équation différentielle $\dot{\mathbf{x}} = f(\mathbf{x})$ de la dynamique continue admet une solution $\xi_{\mathbf{x}}(t)$ de forme close pour chaque état initial $\mathbf{x}$. Par conséquent, l'ensemble atteignable de F contient tous les points $\mathbf{x}$ pour lesquels la formule

$$r(\mathbf{x}) = \exists \mathbf{x}' \ \phi_F(\mathbf{x}') \wedge \exists t \geq 0 \ \mathbf{x} = \xi_{\mathbf{x}'}(t)$$

est vraie. De même, pour prouver que le système satisfait la propriété de sûreté, il suffit de prouver que la formule

$$\forall \mathbf{x}' \ \phi_F(\mathbf{x}') \Rightarrow \ \forall t : t \geq 0 \ \neg \phi_P(\xi_{\mathbf{x}'}(t)) \tag{2.2}$$

est vraie, ce qui peut être fait par l'élimination des quantificateurs. Si ϕ_P, ϕ_F et $\xi_{\mathbf{x}}(t)$ sont définissables dans une théorie pour laquelle l'élimination de quantificateurs est possible, le problème de vérification peut, en principe, être résolu par des méthodes de manipulation symbolique de formules.

Pour les systèmes avec des dynamiques continues non triviales, le problème est beaucoup plus compliqué. D'abord, dans plusieurs cas nous ne connaissons pas les solutions explicites des équations différentielles. En outre, même lorsque nous connaissons de telles solutions, leurs formes ne permettent souvent pas une méthode générale pour prouver l'équation (2.2). Par exemple, pour les systèmes linéaires $\dot{\mathbf{x}} = A\mathbf{x}$ nous avons une solution $\xi_{\mathbf{x}}(t) = \mathbf{x}e^{At}$ de forme close, mais une preuve de (2.2) est possible seulement pour une classe très restreinte des matrices A.

2.5 Notion de la synthèse des contrôleurs

Contrôller un système, consiste à lui fournir à tout moment des signaux de commande qui permettent de satisfaire des spécifications prédéfinies. La synthèse revient donc à synthétiser des fonctions de retour qui permettent au système de réduire ses comportements à un comportement souhaité (figure 2.3). Les objectifs de la synthèse peuvent se résumer à garantir

- la sûreté des trajectoires c'est à dire éviter au système d'entrer dans des états indésirables.

– l'atteignabilité d'un ensemble d'états but, à partir d'un sous ensemble d'états initiaux.

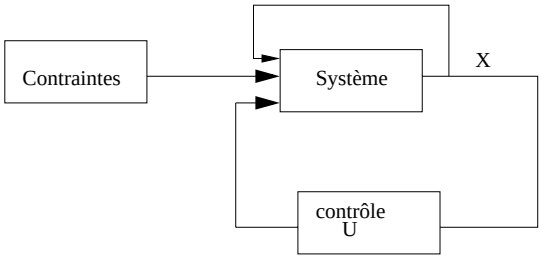


FIG. 2.3 – *Un système ouvert.*

Une approche de synthèse des contrôleurs de sûreté pour des systèmes hybrides à été développée dans [5]. Elle se base sur le principe de commande par commutation entre les états discrets. Le système a plusieurs modes de fonctionement, dont chacun a une dynamique continue distincte. Dans certaines zones de l'espace d'état $\mathcal{X} \subseteq \mathbb{R}^n$, le système peut commuter d'un mode à un autre. Le choix entre les modes est fait par un *contrôleur discret*, qui observe continuellement l'état du système et décide quel mode choisir. Le contrôleur discret est modélisé par un automate dont chaque état discret correspond à un mode du système. Le contrôleur est supposé avoir l'observabilité totale du système, en d'autres termes, l'espace d'observation du contrôleur est $\mathcal{X}$. Par conséquent, le domaine du retour d'état est $Q \times \mathcal{X}$ où Q est l'ensemble des états discrets correspondant à tous les modes possibles.

L'approche de synthèse que nous développons, s'intérèsse à la recherche des commandes qui permettent de diriger le système vers des états cibles prédéfinis tout en respectant des contraintes. Les commandes dans ce cas consistent à appliquer des signaux d'entrées continues qui affectent la dynamique du système.

Dans la suite nous nous limitons à l'étude de la dynamique continue. La technique que nous allons présenter peut être etendue pour le traitement des systèmes hybrides.

Chapitre 3

La simulation des systèmes ouverts

3.1 Les systèmes ouverts

Un système ouvert un système soumi à un ensemble d'entrées externes. Plus formellement:

Définition 6 (Système continu ouvert) *Un système ouvert est un tuplet $\mathcal{C} = (\mathcal{X}, V, f)$ où $\mathcal{X} \subseteq \mathbb{R}^n$ est l'ensemble des état et $V \subseteq \mathbb{R}^m$ est l'ensemble des entrées externes.*
La dynamique d'un tel système est définie par l'équation différentielle:

$$\dot{\mathbf{x}} = f(x, v, u) \tag{3.1}$$

On utilisera l'entrée externe soit pour modéliser les perturbations soit les contrôles.

Définition 7 (Trajectoires d'un système ouvert) *Soit $S(V)$ les valuations des signaux d'entrée, c'est à dire, l'ensemble des fonctions $\psi : \mathcal{T} \to V$. Chaque signal d'entrée $\psi \in S(V)$ et chaque état initial x_0 engendrent une trajectoire $\xi(\psi, x_0) : \mathcal{T} \to \mathcal{X}$ du système.*
Un état x' est atteignable à partir d'un état x, s'il existe un signal ψ et $t \in \mathcal{T}$ telque $\xi(x, \psi)[t] = x'$ qu'on note par $x \xrightarrow{\psi,t} x'$.

Vérification des systèmes ouverts

L'objectif de la vérification est de montrer que pour tout $\psi \in S(V)$, l'intersection de la trajectoire $\xi(\psi)$ avec un ensemble P d'états indésirables est vide. Dans ce cas $S(V)$ représente l'ensemble des perturbations. Plus formellement:

$$\{x' : \exists \psi \in S(V) \exists t \in \mathcal{T} x \xrightarrow{\psi,t} x'\} \cap \mathcal{P} = \varnothing$$

Synthèse des systèmes ouverts

La synthèse qu'on considère ici est relative à la sûreté et à l'atteignabilité. Elle consiste à extraire un sous ensemble de signaux de commandes ξ, dont les trajectoires qui engendrent vérifient:

- L'intersection avec un ensemble d'états P est vide.
- l'atteignabilité d'un ensemble d'états but $\mathcal{F}$.

Formellement:

$$\{\xi(x_o, \psi) \cap P = \varnothing \wedge \xi(x_o, \psi) \cap G \neq \varnothing\}$$

Les signaux d'entrées ici ont pour objet de rediriger les trajectoires du système vers les bons états dés qu'elles risquent de violer les contraintes (Figure 3.1). Ces contraintes peuvent exprimer des conditions de saturation, des obstacles etc. On retrouve ce type de problème

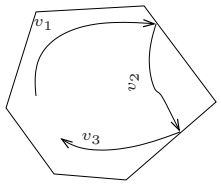


FIG. 3.1 – *Synthèse avec contraintes*

dans le cadre de l'analyse des trajectoires des automates hybrides, les contraintes sont dans ce cas les gardes, des transitions entre les états discret, et les invariants de l'automate. La redirection s'effectue par les transitions entre les états discret. De ce fait, nous estimons que l'approche que nous développons ici peut être appliquée aussi sur les systèmes hybrides.

3.2 La simulation

3.2.1 idée de base

Considérons un système à dynamique continue définie par l'équation différentielle $\dot{x} = f(x)$. La simulation numérique est basée sur la selection d'un ensemble discret de points temporels $\bar{\mathcal{T}} \in \mathbb{R}^+$ et d'un sous espace discret de l'espace des états $\bar{\mathcal{X}} \in \mathcal{X}$. Des integrations numériques successives permettent de générer une approximation discrète de la trajectoire de la forme $\xi' : \bar{\mathcal{T}} \rightarrow \bar{\mathcal{X}}$.
Une observation triviale est que le niveau de confiance dans les résultats de la simulation des systèmes continus est plus bas par rapport à ceux des systèmes discrets, car il existe une certaine distance entre $\xi[t]$ et $\xi'[t]$ pour chaque $t \in \bar{\mathcal{T}}$.

Pour simuler les systèmes ouverts, une première solution consiste à considérer individuellement chaque signal d'entrée. Le problème se réduit donc, à la simulation d'un système fermé. Cependant, l'espace des signaux d'entrée est infini, même si on le restreint au sous ensemble des signaux mesurables ou continus par morceaux, ce qui exclu la possibilité d'une simulation exhaustive. Notre approche va se baser, donc, sur la discrétisation de ces signaux.

3.2.2 discrétisation des signaux d'entrées

Définition 8 (discrétisation (δ, ε)) *Une discrétisation (δ, ε) de $S(V)$ consiste à:*

- *Un ensemble discret de points temporels $\mathcal{T}^\delta = n\delta : n \in \mathbb{N}$.*
- *Un espace des entrées discret $\mathbf{V}^\varepsilon = V \cap (n_1\varepsilon, ...n_m\varepsilon) : (n_1, ...n_m) \in (Z^m)$.*
- *Un signal d'entrée discret est une fonction $\psi' : \mathcal{T}^\delta \to V^\varepsilon$. Elle induit un signal constant par morceaux $\psi'' : \mathcal{T} \to V^\varepsilon$ défini pour tout, $r \in \mathbb{R}^+$, par $\psi''[r.\varepsilon] = \psi'[\lfloor r \rfloor.\varepsilon]$. l'ensemble de ces signaux est $S_\delta(V_\varepsilon)$.*

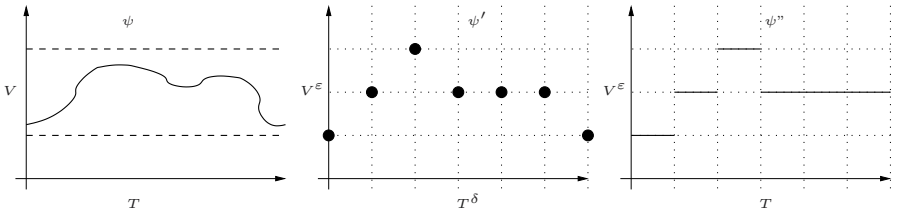


FIG. 3.2 – *Un signal ψ, sa discrétisation ψ' et le signal constant par morceau ψ'' engendré*

La discrétisation est illustrée par la figure 3.2.
Dans la suite nous utiliserons les notations suivantes:

- $\bar{V}$ est l'ensemble des entrées discrètisées.
- $\bar{V}^*$ est l'ensemble des séquences de signaux. Étant donné δ, une séquence v_1, v_2, v_3 représente un signal de longeur 3δ qui prend les valeurs v_1 dans $[0, \delta]$, v_2 dans $[\delta, 2\delta]$ et v_3 dans $[2\delta, 3\delta]$.
- V^k est l'ensemble des signaux de longeur k. En particulier, V^0 correspond à l'ensemble des séquences vides.
- $\bar{V}^{\leq k} = \bigcup_{i=1}^{k} \bar{V}^i$ est l'ensemble de séquences de longeur inférieure ou égale à une constante k.

L'ensemble $\bar{V}^*$ peut être visualisé comme un arbre dont la racine est le mot vide et chaque noeud possède $|\bar{V}|$ successeurs. Ceci est illustré par la figure 3.3.

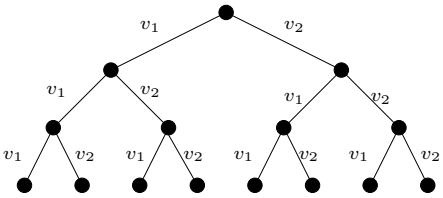


FIG. 3.3 – *Un fragment de l'arbre de* $\bar{V}^*$

3.2.3 Atteingabilité des systèmes discrets

L'ensemble des états atteignables par les signaux constants par morceaux est:

$$R_{\delta,\varepsilon} = \{x' : \exists \psi'' \in S_{\delta,\varepsilon} \exists t \in \mathcal{T}\, x \xrightarrow{\psi'',t} x'\}.$$

La figure 3.4 illustre les trajectoires induites par les éléments de $\bar{V}^*$ de la figure 3.3. Dans la suite, on restraint l'ensemble des états atteingnables au sous ensemble des états atteignables à des points temporels discrets:

$$\bar{R}_{\delta,\varepsilon} = \{x' : \exists \psi'' \in S_{\delta,\varepsilon} \exists t \in \mathcal{T}_\delta x \xrightarrow{\psi'',t} x'\}$$

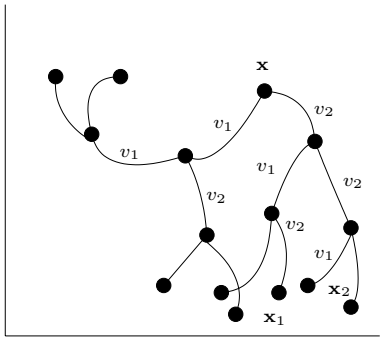


FIG. 3.4 – *Les trajectoires induites sur* $\mathcal{X}$

Nous allons présenter une méthode de calcul d'une approximation de l'ensemble des états atteignable d'un système ouvert. À la différence des approches traditionnelles [1], notre méthode est basée sur une exploration de l'espace des états guidée par les signaux d'entrée et les trajectoires qu'ils induisent.

Notation algébrique

Pour décrire l'action d'une séquence $\psi \in \bar{V}^*$ sur un état x qui mène à x' nous utiliserons la notation algébrique: $x \cdot \psi = x'$.
Cette notation s'applique aussi sur un ensemble d'états ou un ensemble d'entrées. L'ensemble des points atteignables à partir de x à l'instant $k\delta \in \bar{\mathcal{T}}$ est: $R^k(x) = \{x \cdot \psi : \psi \in \bar{V}^k\}$. L'ensemble des états atteignable jusqu'à l'instant $k\delta$ est $R^{\leq k}(x) = \{x \cdot \psi : \psi \in \bar{V}^{\leq k}\}$. Il est claire que $R^{k+1} = R^k \cdot \bar{V}$.
La propriété $x \cdot (\psi \cdot v) = (x \cdot \psi).v$ permet de réutiliser des parties de la simulation. En effet, au lieu d'exécuter deux simulations $\psi \cdot v_1$ et $\psi \cdot v_2$ à partir d'un même état initial, on effectue une première étape de simulation avec ψ pour atteindre un état x', ensuite on continue à partir de x' une fois avec v_1 et une fois avec v_2. À la fin de la k^{ieme} itération, de notre algorithme de calcul, l'ensemble des états atteignables est égal à $R^{\leq k}(x)$ qui est exactement l'ensemble des points que nous obtenons si on exécute $|\bar{V}|^k$ simulations.

Dans ce contexte, la vérification consiste à tester l'intersection de l'ensemble des points atteignables $R^k(x)$ dans un ensemble d'états P.
Cependant, il s'avère que la condition $x \notin P$ n'est pas suffisante pour affirmer que les trajectoires n'atteignent pas un ensemble d'états P. En effet, l'exemple de la figure 3.5 montre que malgrès qu'il existe une trajectoire du système dont l'intersection avec P est non vide, il n'y a pas de points qui sont inclus dans P. Pour atténuer l'erreur nous pouvons procéder à l'interpolation linéaire entre le point courant et son prédecesseur ou considérer des voisinages. Une approche décrite dans [6] calcule des voisinages, sous forme d'ellipsoïdes, au lieu de points. Cette approche permet, entres autres, de fusionner des points qui sont proches. Dans certains cas, elle limite l'evolution exponentielle du nombre des points explorés. Un exemple d'application de cette approche est donné dans l'annexe.

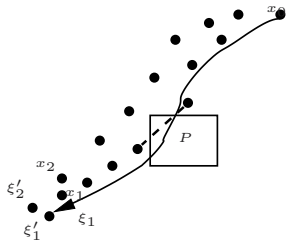


FIG. 3.5 – *Intersection avec P*

3.3 Application à la synthèse

Nous présentons dans cette section une technique de synthèse de contrôleurs basée sur la simulation. L'objectif du contrôle est d'atteindre un ensemble G d'états buts, tout en évitant un ensemble P d'états indésirables. L'ensemble P exprime les contraintes subies par le système. Notre objectif est de calculer un ensemble de signaux de contrôle qui permettent

d'induire des trajectoires conformes aux spécifications données.
L'approche standard de résolution de ce type de problème utilise les techniques du contrôle optimal [9] qui se base sur la théorie de l'optimisation sous contraintes. Généralement, ces techniques discrétisent uniquement le temps. La recherche d'une séquence de contrôle de longeur k se réduit à un problème d'optimisation sur k variables $v[1], \ldots, v[k]$.
Notre approche implique l'exploration d'un ensemble de trajectoires discrètes dont le nombre évolu exponentiellement en fonction de k. D'un point de vue théorique, elle est moins efficace que les approches traditionnelles. Cependant, dans certains cas elle est plus avantageuse:

- Un ensemble P de contrainte assez serrées engendre le rejet d'un grand nombre de trajectoires.
- La méthode d'exploration de l'espace d'état peut être appliquer pour un type quelconque de système, notamment les systèmes à dynamique non linéaires et en particulier les systèmes hybrides.
- L'application de différents algorithmes et heuristiques d'exploration peut éliminer les branches de l'arbre qui ont moins de chance d'aboutir à une trajectoire correcte.

Différents algorithmes d'exploration de l'espace des états discrétisés peuvent être utiliser. Ces algorithmes sont inspirées de la théorie des automates.

3.3.1 Synthèse avec exploration en largeur

Une première alternative consiste à explorer l'espace d'états $\bar{\mathcal{X}}$ en largeur: *breadth-first*. En d'autres termes, nous calculons à chaque pas k d'itération les états atteignables à l'instant $k\delta$. L'algorithme suivant illustre cette méthode:

Algorithme 2 (Algorithme de synthèse en largeur)

Atteignable=En-attente:=$\{x_0\}$*; Nouveau:=* $\varnothing$*;*
Répeter $k = 0, 1, 2, \ldots$
Pour tout $x \in$ *En-attente*
Pour tout $v \in \bar{V}^*$
calcule $x' := x \cdot v$*;*
Si $l(x, x') \cap P = \varnothing$
Nouveau := Nouveau $\cup\{x'\}$*;*
**/* la fonction l représente l'interpolation linéaire*
*entre x et x'ou un voisinage de x' */**
Si $l(x, x') \cap G \neq \varnothing$
/ on teste ici si on est arrivé au but si oui l'algorithme termine*/
Fin
En-attente := En-attente $\setminus\{x\}$*;*
En-attente:=Nouveau; atteignable:= Atteignable $\cup$ *Nouveau; Nouveau:=*$\varnothing$*.*

3.3.2 Exemple

Nous avons tester notre approche sur l'exemple suivant. Considérons une particule lancée, à partir d'un point initial, avec une vitesse horizontale fixe v_x. Elle est soumise à la force de gravitation. Il est possible de lui appliquer des accelérations verticales a_y qui représentent les signaux d'entrées. Notre objectif est de piloter la particule de l'état initial vers un ensemble d'états finaux tout en évitant de franchir des obstacles prédifinis (Figure 3.6).
Nous considérons trois variables du système: les positions x et y et la vitesse v_y. La dynamique est gouvernée par les équations différentielles suivantes:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v_y} = -mg + a_y \end{cases}$$

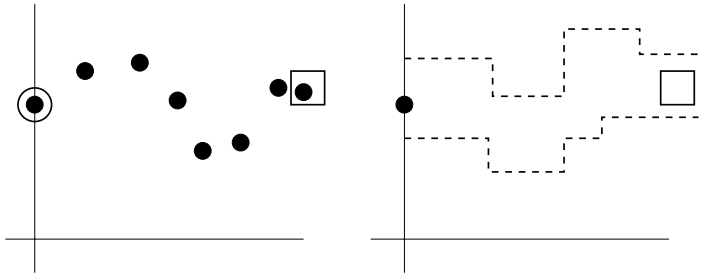


FIG. 3.6 – *Exemple du projectile*

La discrétisation avec un pas δ donné:

$$\begin{bmatrix} x^{k+1} \\ y^{k+1} \\ v_y^{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^k \\ y^k \\ v_y^k \end{bmatrix} + \begin{bmatrix} v_x \\ \frac{-mg+a_y}{2m}\delta^2 \\ \frac{-mg+a_y}{m}\delta \end{bmatrix}$$

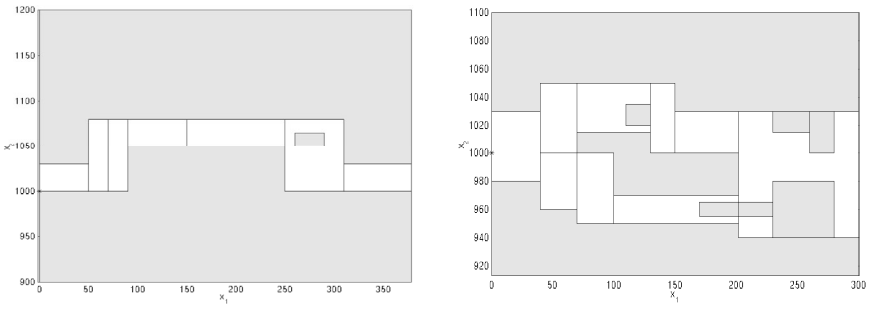
Nous avons appliqué l'algorithme 2 en discrétisant a_y en trois valeurs:$\{400, 0, 200\}$. L'ensemble des entrées est donc:

$$\bar{V} = \left\{ \begin{bmatrix} 1000 \\ 190\delta^2 \\ 380\delta \end{bmatrix}, \begin{bmatrix} 1000 \\ -100\delta^2 \\ -200\delta \end{bmatrix}, \begin{bmatrix} 1000 \\ 60\delta^2 \\ 120\delta \end{bmatrix} \right\}$$

Les contraintes appliquées définissent un ensemble d'états, sous forme de polyhèdres, interdits au système. Nous avons tester deux environnements E_1 et E_2 (figure 3.7)

L'application de l'algorithme 2 sur notre exemple pour $\delta = 0.02$ est illustrée par la figure 3.8. Les contraintes permettent d'eliminer un nombre important de mauvaises trajectoires. Deux solutions, pour E_1 et E_2 sont représentées par la figure 3.10.

Il est évident que le contrôle du système devient plus difficile pour des pas de discrétisation grand. Par exemple pour un pas $\delta = 0.3$, notre algorithme s'arrête sans trouver de solutions. Ceci montre que la commande est impossible avec la discrétisation proposée.



(a) (b)

FIG. 3.7 – *Les environnement E_1 et E_2, l'ensemble P est ombré*

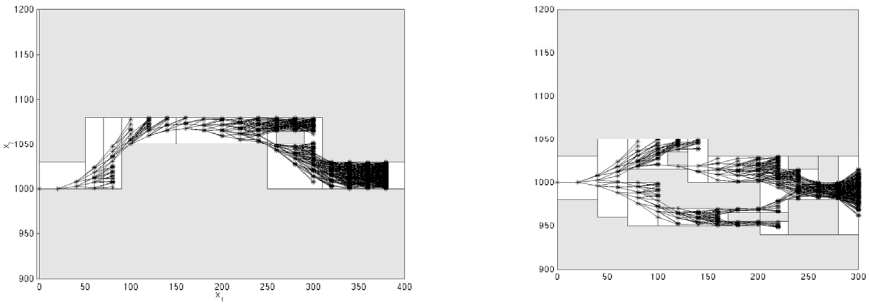


FIG. 3.8 – *Les trajectoires induites par $E1$ et E_2*

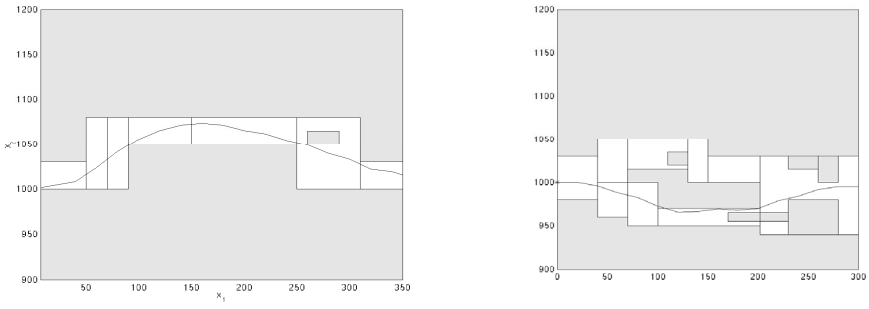


FIG. 3.9 – *Une trajectoire de E_1 et E_2*

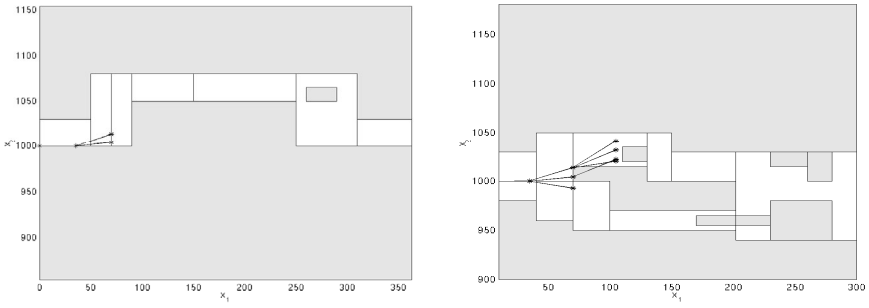


FIG. 3.10 – *Un pas* $\delta = 0.3$ *donne un ensemble de solution vide*

3.3.3 Synthèse avec exploration *best-first*

Les contraintes imposées au systèmes réduisent l'espace d'états à explorer. Cependant, dans certains cas, elles ne permettent pas d'éviter l'*explosion du nombre des états* et donc, d'avoir un temps de calcul raisonable. Nous proposons, donc d'appliquer des méthodes de recherches plus sophistiquées. L'idée consiste à explorer d'abord la trajectoire estimée la plus prometteuse. Pour appliquer cette approche nous avons besoin de définir une fonction d'évaluation pour chaque point dans $\bar{\mathcal{X}}$. cette fonction détermine un ordre de priorité entre les points à explorer. Les ingrédients naturels d'une tel fonction sont:

- La distance du point x à l'ensemble des mauvais états P : si x est trés proche de P alors il a peu de chance d'engendrer une trajectoire correcte.
- La distance à l'ensemble des états but G.

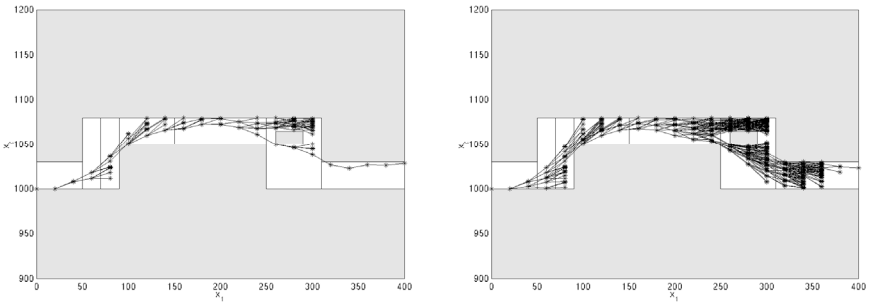
Nous pouvons combiner ces deux critères par la définition de la fonction d'évaluation comme suit: $h(x) = \alpha d(x, G) - \beta d(x, P)$. Les coeficients α et β sont des paramètres positifs. L'algo-

rithme 3 se base sur le tri des points à explorer selon la fonction h.

Algorithme 3 (Algorithme de synthèse *best first*)

Atteignable=En-attente:=$\{x_0\}$*;*
Répeter $k = 0, 1, 2, \ldots$
Pour tout $x \in$ *En-attente*
Pour tout $v \in \bar{V}^*$
calcule $x' := x \cdot v$*;*
Si $l(x, x') \cap P = \varnothing$
Atteignable=Atteignable $\cup\{x'\}$
Pout tout $x'' \in$ *En-attente* $\{x\}$
Si $h(x') < x''$
decaler(x'')
inserer(x')
/* la fonction* decaler *décale les éléments de En-attente à partir de* x'' **/
/* la fonction* inserer *ajoute* x' *dans la position de* x'' **/
Si $l(x, x') \cap G \neq \varnothing$
Fin
En-attente := *En-attente* $\setminus\{x\}$*;*

. La figure 3.11 donne le résultat de l'application de l'algorithme 3 sur l'environnement E_1 de l'exemple de la section 3.3.2 avec différentes valeurs de α et β.



(a) (b)

FIG. 3.11 – *Synthèse* best-first*; (a)* $\alpha = 10$, $\beta = 1$*; (b)* $\alpha = 1$, $\beta = 10$

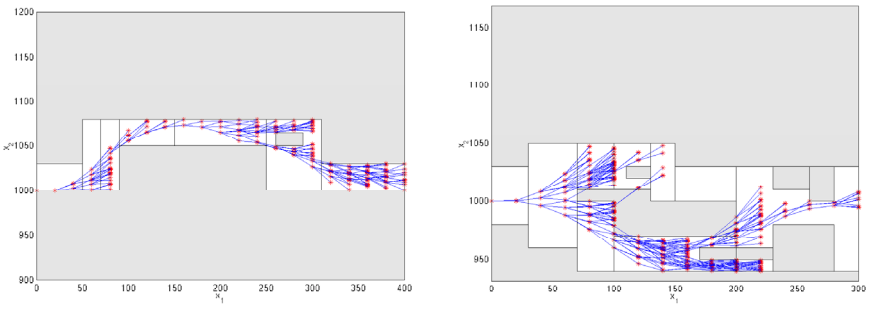
3.3.4 Synthèse avec exploration bornée

L'objective de notre troisième approche d'exploration est la limitation de l'evolution exponentielle du nombre des points explorés. Elle consiste à réduire la largeur de l'arbre de l'espace

d'état discret à une valeur préfixée. À chaque pas de discrétisation, nous développons tous les successeurs des points en cours d'exploration. Nous choisissons parmi eux w points qui sont estimées meilleurs selon la fonction d'évaluation h. Le résultat de l'application de cette heuristique dépend de la valeur choisie de w. Cette heuristique ne garantie pas d'aboutir à des solutions mêmes si elles existent. Un exemple est donné par la figure 3.12.

Algorithme 4 (Algorithme de synthèse avec exploration bornée)

Atteignable=En-attente:=$\{x_0\}$*; Nouveau:=* $\varnothing$*;*
long-fenetre:=constante;
Répeter $k = 0, 1, 2, \ldots$
  Pour tout $x \in$ *En-attente*
    Pour tout $v \in \bar{V}^*$
      calcule $x' := x.v$*;*
      Si $S(x, x') \cap P = \varnothing$
        Si *longueur(Nouveau)* $<$ *long-fenetre*
          Nouveau:=Nouveau $\cup\{x'\}$*;*
        sinon pour i=1 à long-fenetre
          si *coût(*x'*)* $<$ *coût(Nouveau(i))*
            Nouveau(i):=x'*;*
            break
  En-attente $:=$ *En-attente* $\setminus\{x\}$*;*
En-attente:=Nouveau; atteignable:= *Atteignable* $\cup$ *Nouveau; Nouveau:=*$\varnothing$*.*



(a) (b)

FIG. 3.12 – *(a) Environnement* E_1 *avec* $w = 20$*; (b) Environnement* E_2 *avec* $w = 100$

Récapitulatif des trois approches

Les tableaux 1 et 2 présentent les résultats numériques des trois approches décrites. L'inconvinient de l'exploration en largeur est qu'elle nécessite beaucoups plus de temps que les deux autres puisqu'elle explore d'une façon exhaustive toutes les trajectoires possibles.
Le résultat de l'exploration du *best-first* dépend fortement de la fonction d'évaluation choisie. Elle se base sur un tri local des trajectoires et elle se termine dès qu'elle trouve une trajectoire qui atteint le but. Cependant, la trajectoire obtenue n'est pas forcemment la meilleure globalement.
L'exploration bornée est une approche pour réduire l'espace des états. Cependant, son résultat est influencé par le paramètre w et elle ne garantie pas qu'on aboutit à une solution. Elle sera trés utile pour des systèmes ayant un large espace d'états et dont l'exploration exhaustive n'est pas possible.

Tableau 1 (Résultats pour E_1)

	Temps (s)	*No de points simulés*	*No de points acceptés*
En largeur	*775.86*	*4503*	*1689*
Best first $\alpha = 10$, $\beta = 1$	*75*	*951*	*330*
Best first $\alpha = 1$, $\beta = 10$	*332.62*	*4185*	*1973*
Bornée $w = 20$	*96.61*	*1485*	*546*

Tableau 2 (Résultats pour E_2)

	Temps (s)	*No de points simulés*	*No de points acceptés*
En largeur	*1390*	*5475*	*2282*
Best first $\alpha = 10$, $\beta = 1$	*148.47*	*468*	*169*
Bornée $w = 100$	*298.63*	*4020*	*1195*

Chapitre 4

Conclusion et Perspectives

Ce rapport est la synthèse du travail réalisé durant mon projet de DEA.
Nous avons commencé par présenter la modélisation des systèmes hybrides et les notions de la verification et de la synthèse . Nous avons constaté la complexité de ces tâches et avons observé les limitations des approches existantes.
Nous nous sommmes intéressés par la suite à l'études des systèmes à dynamique continue ouverts qui représentent le problème majeur dans la résolution des systèmes hybrides. L'approche que nous avons développée est inspirée de la théorie des automates discrets et des techniques de la simulation numérique . Nous avons procédé à la discrétisation du temps et des signaux d'entrées et donc, l'exploration des trajectoires discrètes induites.
La méthode de simulation développée est appliquée pour la résolution de la synthèse des systèmes ouverts. Le problème de synthèse qu'on a considéré, consiste à quider le système d'un ensemble d'états initiaux vers un ensemble d'états finals tout en respectant des contraintes. Nous appliquer trois types d'algorithme d'exploration: en largeur, *best-first* et bornée.

De nombreux points n'ont encore été que succintement abordés, et restent donc à l'étude. C'est le cas notament de l'exploitation de la méthode de simulation des voisinages abordée dans [6]. L'application des méthodologies plus rigoureuses pour la recherche du contrôle optimale peut améliorer les algorithmes développés
Un autre point qui est en cours d'étude mais reste encore à l'état de perspective est la généralisation de la méthodologie développée pour l'étude des systèmes hybrides.

Annexe A

Exemple: application à la vérification

Cet exemple est extrait de [6]. Le système est illustré par la figure A. L'objectif est de vérifier que pour tous les signaux d'entrée, la distance entre la valeur actuelle et la référance reste bornée. x_1 représente la sortie du système, v est le signal d'entrée, x_2 est la sortie filtrée du signal et $d = x_2 - x_1$ est l'erreur. La dynamique du système est définie par: $\dot{\mathbf{x}} = A\mathbf{x} + Bv$ avec:

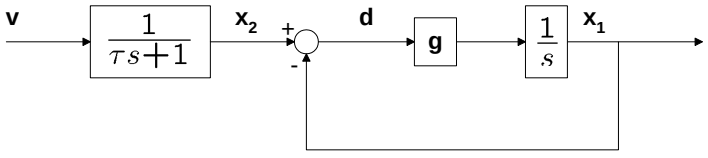
$$A = \begin{bmatrix} -g & g \\ 0 & -\frac{1}{\tau} \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \frac{1}{\tau} \end{bmatrix}$$

La dynamique du système discrétisé avec un pas δ est:

$$\mathbf{x}_{k+1} = \Phi\mathbf{x}_k + \Gamma v_k$$
$$\Phi = e^{A\delta} \qquad \Gamma = A^{-1}(e^{A\delta} - I)B$$

pour $g = 10$, $\tau = 0.1$, et $\delta = 0.1$:

$$\Phi = \begin{bmatrix} 0.368 & 0.368 \\ 0.000 & 0.368 \end{bmatrix} \qquad \Gamma = \begin{bmatrix} 0.264 \\ 0.632 \end{bmatrix}$$



Le but est de verifier que le système n'atteint pas l'ensemble: $P = \{x : |x_1 - x_2| > 1\}$ pour les séquences d'entrées:$\bar{V}^* = 0.0, 0.5, 1.0^*$. La figure **??** illustre les trajectoires du système.

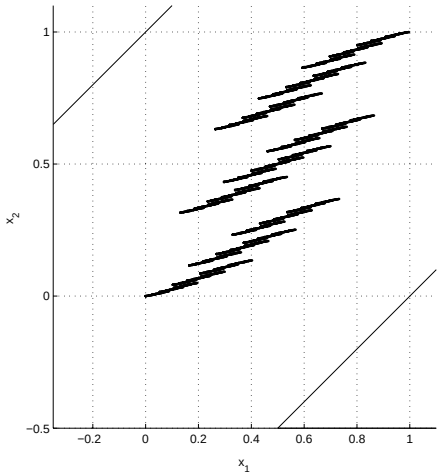


FIG. A.1 – *Les états atteignables à partir de (0.0)*

Annexe B

Implémentation

Nous avons implémenté les algorithmes, présentés dans le chapitre précédent, sous *MATLAB*. L'implémentation considère les systèmes à dynamiques continues linéaires discrétisés, c'est à dire qui admettent des équations d'états de la forme:

$$x_{k+1} = Ax_k + Bv_k$$

Notre outil *explore* est constitué de trois modules:

- Module *def* permet de définir la dynamique discrétisée du système sous étude.
- Module *Param* contient les paramètres relatifs au choix des signaux d'entrée, du pas de discrétisation et du type de l'exploration (atteignabilité, vérification ou sysnthèse).
- Module *exploration* se charge du calcul itératif des trajectoires du système. Dans ce module nous avons utilisé des fonctions de la librairie de *CheckMate* [12]. Ces fonctions sont relatives à la définition des contraintes appliquées (Polyhedron, linearcon) et au calcul de la satisfaction de ces contraintes par les trajectoires explorées (feasible_point, isfeasible).

Bibliographie

[1] E. Asarin, O. Bournez, T.Dang, O. Maler, A. Pneuli, *Effective Synthesis of Switching Controllers for Linear Systems*, Proceeding of the IEEE,88,1011-1025,2000.

[2] E. Asarin, O. Maler, A. Pneuli, *Symbolic Controller Synthesis for Discrete and Timed Systems*, in P.Antsaklis, W.Kohn, A.Nerode and S.Sastry(Eds) Hybrid SystemsII, 1-20, LNCS 999, springer, 1995.

[3] O. Botchkarev, S. Tripakis, *Verification of hybrid systems with Linear Differential Inclusion using Ellipsoidal Approximations*. In Hybrid Systems: Computation and Control,2000. Lecture Notes in Computer Science, Springer-Verlag.

[4] P.E. Caines, S. Shaikh, *On the Optimal Control of Hybrid Systems: Optimization of Trajectories, Switching Times, and location Schedules*, 6th International Workshop, HSCC 2003 Prague, Czech Republic, April 2003.

[5] T. Dang, *Verification and Synthesis of Hybrid Systems*, Verimag, These doctorale 2000.

[6] B H. Krogh, O. Maler, J. Kapinski, O. Strusberg,*On Systematic Simulation of Open Continuous Systems*, Hybrid Systems Computation and Control, 6th International Workshop, HSCC 2003 Prague, April 2003.

[7] O. Maler, *Control from Computer Science*, Annual Reviews in Control, 2003

[8] O. Maler, *A Unified Approach for studying Discrete and Continuous Dynamical Systems*, Proc. CDC'98, IEEE, 1998.

[9] J. Macki, A. Strauss, *Introduction to optimal Control Theory*, Springer, 1982.

[10] A. Olivero, *Modélisation et Analyse des Systèmes Temporisés et Hybrides*, Thèse doctorale, Septembre 1994.

[11] G. Pappas, G. Lafferriere, S. Yovine, *A new class of decidable hybrid systems*, In Proceeding of "Hybrid Systems: Computation and Control, HSCC'00". Nijmegen, the Netherlands, March 29-31,1999.

[12] http://www.ece.cmu.edu/ webk/checkmate/content.htm