A. Labroue

Conditions de vivacité dans les automates temporisés

Research Report LSV–98–7, Sep. 1998

Laboratoire Spécification et Vérification

Ecole Normale Supérieure de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex France

http://www.lsv.ens-cachan.fr/Publis/
Research Report LSV-98-7, Lab. Spécification et Vérification, CNRS & ENS de Cachan, France, Sep. 1998
First published as *Mémoire de DEA, DEA d'Informatique, Orsay, June 1998.*

Table des matières

1 Introduction **3**

2 Rappels **6**
2.1 Systèmes de transitions 6
2.1.1 Définitions 6
2.1.2 Les bisimulations entre systèmes de transitions 7
2.1.3 Les isomorphismes entre systèmes de transitions 8
2.2 Automates temporisés 8
2.2.1 Les horloges 8
2.2.2 Définition d'un automate temporisé 10
2.2.3 Système de transitions associé à un automate temporisé 11

3 Automates temporisés avec invariants ou conditions limites **13**
3.1 Les invariants 13
3.2 Différentes interprétations des invariants 14
3.2.1 Automates de Uppaal 14
3.2.2 Automates de Deadline 15
3.3 Structures des ensembles de valuations 15
3.3.1 Clôture par le passé 16
3.3.2 Convexité pour le temps 16
3.3.3 Fermeture à droite 17
3.4 Simulation des automates de Deadline par des automates de Uppaal, dans le cas où les invariants sont clos par le passé 20
3.4.1 Introduction 20
3.4.2 Automate de Uppaal associé 21
3.4.3 Les automates ont des systèmes de transitions bisimilaires 22
3.5 Ensembles particuliers de valuations d'horloges 23
3.5.1 Présentation du problème 23
3.5.2 Les régions d'horloges 24
3.5.3 Les domaines d'horloges 25
3.6 Simulation des automates de Deadline par des automates de Uppaal, dans le cas général 28
3.6.1 Automate de Uppaal associé 28
3.6.2 Les automates ont des systèmes de transitions bisimilaires 28
3.7 L'horloge supplémentaire est-elle nécessaire? 29
3.8 Simulation des automates de Uppaal par des automates de Deadline 31

3.8.1 Situation 31
3.8.2 La bisimulation mixte 31
3.8.3 Automate de Deadline associé 32
3.8.4 Propriétés des systèmes de transitions relatives au temps 34
3.9 Les automates avec conditions limites 36
3.9.1 Définition d'un automate avec conditions limites 36
3.9.2 Automate avec conditions limites associé à un automate de Deadline . 37
3.9.3 Conclusion 38

4 Produits de synchronisation **39**
4.1 Produit de synchronisation de deux systèmes de transitions 39
4.1.1 Définition de la composition de deux systèmes de transitions temporisés 39
4.1.2 Propriétés du produit de synchronisation de deux systèmes de transitions 40
4.2 Produit de synchronisation pour les automates avec invariants 40
4.2.1 Composition d'automates avec invariants 40
4.2.2 Théorème de correction pour les automates de Uppaal 43
4.2.3 Théorème de correction pour les automates de Deadline 44
4.2.4 Conséquences des théorèmes de correction 45
4.3 Produit de synchronisation pour les automates avec conditions limites 46
4.3.1 Composition d'automates avec conditions limites 46
4.3.2 Théorème de correction pour les automates avec conditions limites . . 47

5 Conclusion **48**

A Démonstrations Techniques **50**
A.1 A propos de la relation d'ordre sur les régions 50
A.2 Simulation des automates de Deadline par des automates de Uppaal dans le cas général 52
A.2.1 Résultats intermédiaires pour la démonstration du fait que $\mathcal{R}$ a la propriété de transfert 52
A.2.2 Démonstration que $\mathcal{R}$ a la propriété de transfert 53
A.3 Démonstrations de plusieurs propriétés du produit de synchronisation de deux systèmes de transitions 54
A.3.1 L'associativité de l'opération de composition 54
A.3.2 La commutativité de l'opération de composition 55
A.3.3 La bisimulation est une congruence par rapport à l'opération de composition 56
A.4 Démonstration du théorème de correction pour les automates de Uppaal . . . 56

Chapitre 1

Introduction

La modélisation et la vérification des systèmes réactifs sont aujourd'hui un enjeu scientifique et industriel majeur. La vérification automatique est devenue faisable en pratique quand Alur et *al.* ont montré qu'il était possible d'appliquer les techniques de model-checking aux automates temporisés, une famille modélisant les systèmes temps-réel ayant un nombre fini d'états de contrôle, [ACD93].

Depuis, de nombreux systèmes temps-réel, parfois de taille industrielle, ont été validés par des model-checkers, tels que Kronos, [DY95], Uppaal, [BCJ+], Hy-Tech, [HHW95]. Par exemple, Uppaal a permis de détecter une erreur dans un protocole audio/vidéo de Bang & Olufsen, de la corriger et de valider le protocole obtenu [HSLL97].

En fait, les automates utilisés sont différents du modèle présenté dans [ACD93] et [AD90]. Ils sont munis d'**invariants**, de **conditions limites (deadlines)** ou d'**actions urgentes** qui permettent de garantir la vivacité des systèmes en cours d'exécution. Toutes ces conditions de vivacité ayant le même but, leurs interprétations sont très proches.

Cependant leurs disparités ont des conséquences très importantes, le comportement d'un même automate peut changer radicalement suivant l'interprétation utilisée. Contrairement aux apparences, il n'est pas évident, étant donné un automate et une interprétation, de trouver un autre automate qui, avec une autre interprétation, aura le même comportement que l'automate donné. Lorsque cela est possible, cette traduction peut ne pas être compatible avec les produits d'automates.

Lorsque l'on étudie la littérature, on constate que les problèmes posés par la diversité des automates temporisés avec conditions de vivacité ne sont pas abordés, ni même soulevés.

De nombreux travaux étudient les **langages** reconnus par les automates temporisés. Ils s'intéressent aux propriétés de clôture de ces langages [AD90], au pouvoir d'expressions des automates temporisées suivant la forme des gardes choisie et l'utilisation éventuelle d'actions invisibles [BGP96, BGP97] ou suivant le choix des paramètres suivants : le domaine de temps (discret ou dense), le nombre d'horloges, le nombre d'états et de la taille des constantes utilisés dans les contraintes d'horloges [HKW95]. Ces exemples ne prennent pas en compte les invariants : ils utilisent des conditions de Büchi/Muller pour garantir la vivacité.

D'autres articles présentent les model-checkers [Yov97, BCJ+, HHW95]. En fait, chacun d'eux présente une version précise d'un outil. Ils permettent tous de construire le système à vérifier par composition (produit de synchronisation) d'automates simples. La syntaxe, la sémantique et le produit de synchronisation des automates temporisés utilisés sont définis de façon plus ou moins précis. En particulier, le choix du produit de synchronisation est rarement

justifié et ses propriétés souvent omises. Par exemple, la lecture de [DY95] ne permet pas de voir facilement, à moins de faire un calcul, si le produit de synchronisation proposé est commutatif.

Un dernier groupe de travaux propose des études théoriques des automates avec conditions de vivacité. [BS97] présente pour les automates hybrides les rapports entre les fonctions de progression du temps et les conditions limites, et mentionne à la dualité qui existe entre les invariants et les conditions limites dans le cas des automates temporisés. Cette dernière est également évoquée dans [SY96] qui propose plusieurs produits de synchronisation pour les automates avec invariants et les compare au produit de synchronisation des automates avec conditions limites et à celui des réseaux de Pétri. [Alu98] associe un système de transitions à chaque automate temporisé avec invariants. Il propose également un produit de synchronisation pour les systèmes de transitions et un pour les automates avec invariants et vérifie qu'ils sont cohérents.

Notre travail a tout d'abord consisté à recenser les différentes sémantiques d'automates avec invariants ou avec conditions limites proposés dans la littérature et à les définir de façon précise, notamment en associant à chacune un système de transitions. Nous en avons distingué trois : deux pour les automates temporisés avec invariants, les automates de Uppaal et les automates de Deadline, et une sémantique d'automates avec conditions limites.

Nous les avons ensuite comparées entre elles, en cherchant à associer à tout automate d'un type X (de Deadline, de Uppaal ou avec conditions limites), un automate d'un type Y qui ait un "comportement semblable" . Nous avons vérifié que, comme l'affirmaient [SY96, BS97], étant donné un automate de Deadline, on peut trouver un automate avec conditions limites ayant le même système de transitions (le sens inverse est une tautologie d'après les définitions choisies). Nous avons associé, à tout automate de Deadline, un automate de Uppaal qui lui est bisimilaire et à tout automate de Uppaal, un automate de Deadline qui lui est **bisimilaire mixte** (la bisimulation mixte est une variante de la bisimulation classique). Ces deux résultats sont nouveaux et répondent à une problématique qui n'a pas été étudiée pour l'instant. Nous avons également exhibé un automate de Uppaal dont le langage n'est reconnu par aucun automate de Deadline et un automate de Deadline tel que tout automate de Uppaal reconnaissant le même langage a au moins une horloge de plus que lui.

Nous avons ensuite défini un produit de synchronisation sur les systèmes de transitions et des produits de synchronisation pour les automates avec invariants et avec conditions qui lui sont compatibles. Ces définitions de produits de synchronisation sont très proches, parfois identiques à celles proposées dans la littérature [Alu98, SY96, DY95]. Ce qui est nouveau est que nous avons défini précisément ce que signifie "compatible" : *étant donnés deux automates temporisés du même type (de Deadline, de Uppaal ou avec conditions limites)* A_1 *et* A_2*, le produit de synchronisation des systèmes de transitions de* A_1 *et* A_2 *est isomorphe au système de transitions associé à l'automate égal au produit de synchronisation de* A_1 *et* A_2 et nous en avons donné une preuve. Nous avons également vérifié que tous ces produits sont commutatifs et associatifs.

Enfin, nous avons montré que notre traduction d'automate de Deadline en automate de Uppaal est compatible avec le produit de synchronisation. Plus précisément, étant donnés deux automates temporisés de Deadline A_1 et A_2, si B_1 et B_2 sont les deux automates de Uppaal que nous leurs avons associé, alors les équivalences habituelles pour les comparer.

Plan : Dans le chapitre 1, nous rappelons ce que sont les systèmes de transitions et les automates temporisés, ainsi que plusieurs moyens de comparer des systèmes de transitions ou des automates temporisés.

Dans le chapitre 2, nous définissons deux sémantiques d'automates temporisés avec invariants : les automates de Uppaal et les automates de Deadline (3.2), et une sémantique d'automates avec conditions limites(3.9).

Nous montrons comment trouver, étant donné un automate de Deadline, un automate avec conditions limites ayant le même système de transitions (l'autre sens est trivial)(3.9), comment associer à tout automate de Deadline, un automate de Uppaal qui lui soit bisimilaire (3.4 et 3.6) et comment déterminer, étant donné un automate de Uppaal, un automate de Deadline qui lui soit bisimilaire (3.8).

Pour cela, nous définissons de nouveaux outils : plusieurs propriétés sur les ensembles de valuations d'horloges, *clôture par le passé*, *convexité pour le temps* et *fermeture à droite* (3.3) et des nouveaux ensembles de valuations : les *domaines d'horloges* (3.5) et une nouvelle notion de bisimulation : la *bisimulation mixte* (3.8).

Nous exibons également un automate de Uppaal tel qu'aucun automate de Deadline n'ait le même langage (3.8).

Dans le chapitre 3, nous définissons des produits de synchronisation sur les systèmes de transitions, les automates avec invariants et les automates avec conditions limites. Nous vérifions que, étant donné deux automates temporisés du même type (de Deadline, de Uppaal ou avec conditions limites) A_1 et A_2, le produit de synchronisation des systèmes de transitions de A_1 et A_2 est égal au système de transitions associé à l'automate égal au produit de synchronisation de A_1 et A_2. Nous montrons aussi que tous ces produits de synchronisation sont commutatifs et associatifs.

Enfin, nous prouvons que, étant donné deux automates de Deadline A_1 et A_2, si B_1 et B_2 sont les deux automates de Uppaal que nous leur avons associés, alors le produit de synchronisation de A_1 par A_2 est bisimilaire au produit de synchronisation de B_1 par B_2.

Chapitre 2

Rappels

2.1 Systèmes de transitions

L'un des objectifs de ce travail est d'étudier les sémantiques des différents automates temporisés rencontrés dans la littérature. Pour définir ces sémantiques, nous associons un système de transitions à chaque type d'automates. Ensuite, nous comparons ces sémantiques en comparant leur systèmes de transitions. Dans cette section, nous définissons les systèmes de transitions et présentons deux équivalences entre systèmes de transitions : la bisimulation et l'isomorphisme.

2.1.1 Définitions

Définition 2.1.1 *Un* **système de transitions** *est un quadruplet* $\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ *où :*

- S *est un ensemble d'états (qui peut être infini),*
- s_0 *est l'état initial,* $s_0 \in S$*,*
- $\mathcal{A}ct = \{\ell, ..\}$ *est un ensemble d'actions (il sera souvent sous-entendu),*
- $\rightarrow \subset S \times \mathcal{A}ct \times S$ *est un ensemble de transitions, on notera indifféremment* $(s, \ell, s') \in \rightarrow$ *ou* $s \xrightarrow{\ell} s'$*,* ℓ *est l'étiquette de la transition* (s, ℓ, s')*.*

Exemple 2.1.2 (le producteur-consommateur)
Le producteur produit des objets un par un. Le consommateur consomme les objets produits. Le système de transitions de la figure 2.1, noté $\mathcal{S}_1$, modélise ce système.

Les états sont des paires d'entiers ((nombre d'objets produits, nombre d'objets consommés)), l'état initial est (0,0). Il y a deux actions : p (production) et c (consommation). Les transitions sont représentées par les flèches.

Définition 2.1.3 *Une* **exécution** *d'un système de transitions est une suite d'états* (u_i) *commençant par l'état initial,* s_0 *et une suite de transitions* (t_i) *telles que pour tout i, il existe* $\ell_i \in \mathcal{A}ct$ *tel que* $t_i = (u_i, \ell_i, u_{i+1})$*.*
La **trace** *d'une exécution du système de transitions est la suite formée des étiquettes des transitions.*
Le **langage** *d'un système de transitions est l'ensemble de ses traces.*

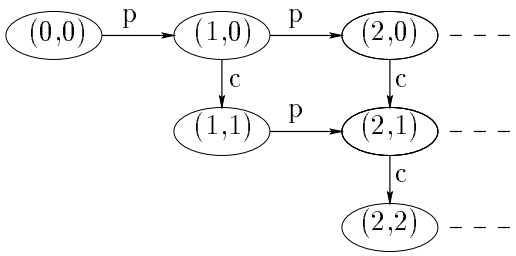


FIG. 2.1 – *Exemple d'un système de transitions: le producteur-consommateur*

Exemple 2.1.4 $(0,0) \xrightarrow{p} (1,0) \xrightarrow{p} (2,0) \xrightarrow{c} (2,1) \xrightarrow{p} (3,1)$ est une exécution de l'exemple précédent. (p,p,c,p) en est la trace.

Une première façon de comparer deux systèmes de transitions est de comparer leur langages.

Définition 2.1.5 *Soient $\mathcal{S}$ et $\mathcal{S}'$ deux systèmes de transitions. Lorsque $\mathcal{S}$ et $\mathcal{S}'$ ont le même langage, on note $\mathcal{S} \equiv_L \mathcal{S}'$.*

2.1.2 Les bisimulations entre systèmes de transitions

Soient $\mathcal{S} = (S, s_0, \rightarrow)$ et $\mathcal{S}' = (S', s'_0, \rightarrow')$ deux systèmes de transitions. On souhaite les comparer. Une première façon consiste à comparer leurs langages. Une autre, plus fine, est de regarder s'ils sont bisimilaires (notion due à Park et Milner [Mil89]).

Définition 2.1.6 *Soit $\mathcal{R}$ une relation incluse dans $S \times S'$. $\mathcal{R}$ a la* **propriété de transfert** *si $\forall (s,s') \in \mathcal{R}$, $\forall \ell \in \mathcal{A}ct$,*

1. *$\forall u \in S$ tel que $s \xrightarrow{\ell} u$, $\exists u' \in S'$ tel que $s' \xrightarrow{\ell} u'$ et $(u,u') \in \mathcal{R}$.*
2. *$\forall u' \in S'$ tel que $s' \xrightarrow{\ell} u'$, $\exists u \in S$ tel que $s \xrightarrow{\ell} u$ et $(u,u') \in \mathcal{R}$.*

Définition 2.1.7 *Une relation $\mathcal{R}$ incluse dans $S \times S'$ est une* **bisimulation** *entre $\mathcal{S}$ et $\mathcal{S}'$ si elle a la propriété de transfert et si $(s_0, s'_0) \in \mathcal{R}$. On note alors $\mathcal{R} : \mathcal{S} \leftrightarrow \mathcal{S}'$.*

Définition 2.1.8 *$\mathcal{S}$ et $\mathcal{S}'$ sont* **bisimilaires** *s'il existe une bisimulation entre $\mathcal{S}$ et $\mathcal{S}'$, on note alors $\mathcal{S} \leftrightarrow \mathcal{S}'$.*

Exemple 2.1.9 (suite du producteur-consommateur)
Le système de transitions de la figure 2.2, noté $\mathcal{S}_2$, modélise le système de producteur-consommateur d'une manière différente de celle proposée par l'exemple 2.1.2.

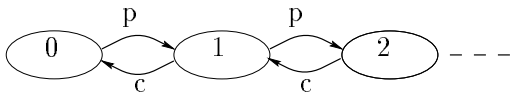


FIG. 2.2 – *Deuxième exemple de système de transitions pour le producteur-consommateur*

Les états sont des entiers (nombre d'objets produits - nombre d'objets consommés), l'état initial est 0.

Soit $\mathcal{R} = \{((np, nc), np - nc)$ tel que $0 \leq nc \leq np\}$, $\mathcal{R}$ est une bisimulation entre $\mathcal{S}_1$ et $\mathcal{S}_2$.

Proposition 2.1.10 *La bisimilarité est une relation d'équivalence.*

Proposition 2.1.11 *Soient $\mathcal{S}$ et $\mathcal{S}'$ deux systèmes de transitions. Si $\mathcal{S}$ et $\mathcal{S}'$ sont bisimilaires, alors ils ont le même langages.*

2.1.3 Les isomorphismes entre systèmes de transitions

Une autre façon de comparer deux systèmes de transitions est de regarder s'ils sont isomorphes. Nous utiliserons la définition suivante.

Définition 2.1.12 *Soit $h : S \longrightarrow S'$ une bijection telle que $(s, \ell, u) \in \rightarrow$ si et seulement si $(h(s), \ell, h(u)) \in \rightarrow'$ et telle que $h(s_0) = s'_0$. h est un* **isomorphisme** *entre $\mathcal{S}$ et $\mathcal{S}'$. On note $h : \mathcal{S} \equiv \mathcal{S}'$.*

Définition 2.1.13 *On dit que $\mathcal{S}$ et $\mathcal{S}'$ sont* **isomorphes** *s'il existe un isomorphisme h entre $\mathcal{S}$ et $\mathcal{S}'$. On notera alors $\mathcal{S} \equiv \mathcal{S}'$.*

Proposition 2.1.14 *Soient $\mathcal{S} = (S, s_0, \rightarrow)$ et $\mathcal{S}' = (S', s'_0, \rightarrow')$ deux systèmes de transition.*
Si $h : \mathcal{S} \equiv \mathcal{S}'$ alors il existe $\mathcal{R} \in S \times S'$ telle que $\mathcal{R} : \mathcal{S} \leftrightarrow \mathcal{S}'$.
Si $\mathcal{S}$ est isomorphe à $\mathcal{S}'$, alors $\mathcal{S}$ est bisimilaire à $\mathcal{S}'$.

Démonstration :
Soit $\mathcal{R} \in S \times S'$ est définie par $(s, s') \in \mathcal{R}$ si et seulement si $s' = h(s)$.

□

Remarque 2.1.15 La bisimulation n'entraîne pas l'isomorphisme. Par exemple, les deux systèmes de transitions présentés pour modéliser le producteur-consommateur ne sont pas isomorphes, mais sont bisimilaires.

2.2 Automates temporisés

Dans cette partie, nous présentons un modèle pour les systèmes temps-réel ayant un nombre fini d'états de contrôle, les automates **temporisés**, [AD90]. Ce modèle est obtenu à partir des automates d'états finis en introduisant le temps par l'intermédiaire d'horloges et de contraintes sur ces horloges.

2.2.1 Les horloges

Dans toute la suite, le domaine du temps, noté $\mathbb{T}$, sera l'ensemble des réels positifs ou nuls. Dans cette sous-partie, X sera un ensemble d'horloges.
Informellement, les horloges sont des sortes de chronomètres qui avancent à la même vitesse que le temps et que l'on peut remettre à zéro indépendamment les unes des autres. Une

valuation est l'ensemble formé des valeurs prises par chacune de ces horloges à un instant donné.

Définition 2.2.1 *Une* **valuation d'horloge** *est une application v de X dans $\mathbb{T}$.*

Remarque 2.2.2 Si $n = card(X)$, l'ensemble des applications de X dans $\mathbb{T}$ est isomorphe à $\mathbb{T}^n$. Etant donné un ordre sur X, on note la valuation $v : x_1 \mapsto a_1, ..., x_n \mapsto a_n$, $v = (a_1, ..., a_n)$.

Définition 2.2.3 *Les* **contraintes d'horloge** *sont les formules définies par la grammaire:*

$$A ::= x \sim t |\ x - y \sim t |\ A \wedge A |\ \neg A |\ True$$

où $\sim \in \{<, \leq, =, \geq, >\}$, $x \in X, y \in X, t \in \mathbb{N}$.
$\mathcal{C}(X)$ *est l'ensemble des contraintes d'horloge sur* X.

Exemple 2.2.4 Si $X = \{x, y\}$, "$x - y \leq 3$" et "$(x = 3) \wedge (y < 5)$" sont des contraintes d'horloges.

Définition 2.2.5 *Les* **contraintes d'horloge positives** *sont les formules définies par la grammaire:*

$$A ::= x \sim t |\ x - y \sim t |\ A \wedge A |\ True$$

où $\sim \in \{<, \leq, =, \geq, >\}$, $x \in X, y \in X, t \in \mathbb{N}$.
$\mathcal{C}^+(X)$ *est l'ensemble des contraintes d'horloge positives sur* X.

Remarque 2.2.6 Une contrainte d'horloge positive est une contrainte d'horloge qui peut s'écrire sans $\neg$ et sans $\vee$.

Définition 2.2.7 *Soit $v = (a_1, ..., a_n)$ une valuation et d un réel positif ou nul, on notera $v + d$ la valuation $(a_1 + d, ..., a_n + d)$.*

Remarque 2.2.8 Etant donnés une valuation v et deux entires d et d', $(v + d) + d' = v + (d + d')$.

Définition 2.2.9 *Soit v une valuation d'horloge et A une contrainte d'horloge, v* **satisfait** *A, noté $v \vDash A$, si $A(v)$ a la valeur vraie. Sinon v ne satisfait pas A.*

Exemple 2.2.10 On ordonne X de la façon suivante $x = x_1$ et $y = x_2$, alors $(2, 3) \vDash (x - y \leq 3)$ et $(3.2, 3) \nvDash ((x = 3) \wedge (y < 5))$.

Proposition 2.2.11 *Une contrainte d'horloge peut s'écrire comme une disjonction de contraintes d'horloge positives.*

Démonstration : Soit ϕ une contrainte d'horloge, ϕ a une forme normale disjonctive :

$$\phi \equiv (\phi_{1,1} \wedge \phi_{1,2} \wedge ... \wedge \phi_{1,n_1}) \ \vee \ (\phi_{2,1} \wedge \phi_{2,2} \wedge ... \wedge \phi_{2,n_2}) \ \vee ... \vee \ (\phi_{k,1} \wedge \phi_{k,2} \wedge ... \wedge \phi_{k,n_k}).$$

où les $\phi_{j,l}$ appartiennent à la grammaire $A ::= x \sim t |\ x - y \sim t |\ \neg A |\ True$ avec $\sim \in \{<, \leq, =, \geq, >\}$, $x \in X, y \in X, t \in \mathbb{N}$.

On fait disparaître les négations en remarquant par exemple que $\neg(x < n) \equiv x \geq t$ ou que $\neg(x - y = t) \equiv ((x - y > t) \vee (x - y < t))$. On développe les conjonctions afin de faire disparaitre les $\vee$ qui sont apparus et on regroupe les conjonctions. On obtient

$$\phi \equiv \phi_1 \vee \phi_2 \vee ... \vee \phi_k$$

où les ϕ_j sont des contraintes d'horloge positives.

□

Exemple 2.2.12

$$\begin{aligned}\neg((x < 3) \vee (x = 2)) &\equiv (\neg(x < 3)) \wedge (\neg(x = 2)) \\ &\equiv (x \geq 3) \wedge ((x > 2) \vee (x < 2)) \\ &\equiv ((x \geq 3) \wedge (x > 2)) \vee ((x \geq 3) \wedge (x < 2))\end{aligned}$$

Définition 2.2.13 *Une contrainte A définit un sous ensemble de $\mathbb{T}^n$ appelé une* **zone** *: $Z_A = \{v \text{ tel que } v \models A\}$.*

Les zones associées aux contraintes d'horloges positives sont des polyèdres de $\mathbb{T}^n$ où un polyèdre est une intersection de demi-espaces (ouverts, fermés ou dégénérés) qui sont délimités par des hyperplans d'équations $x = t$ et $x - y = t$ avec $(x, y) \in X^2$ et $t \in \mathbb{T}$. Elles sont donc convexes. Une conséquence de la proposition 2.2.11 est que les zones associés aux contraintes d'horloges sont donc des réunions finies de polyèdres de $\mathbb{T}^n$. Par exemple, la figure 2.3 représente les zones définies par les contraintes $(y \geq 1) \wedge (y \leq 3) \wedge (x \leq 2)$, $(x \leq 1) \wedge (y \geq 2) \wedge (y - x \geq 0)$ et $((y \geq 1) \wedge (y \leq 3) \wedge (x \leq 2)) \vee ((x \leq 1) \wedge (y \geq 2) \wedge (y - x \geq 0))$. Les deux premières zones sont convexes, pas la troisième.

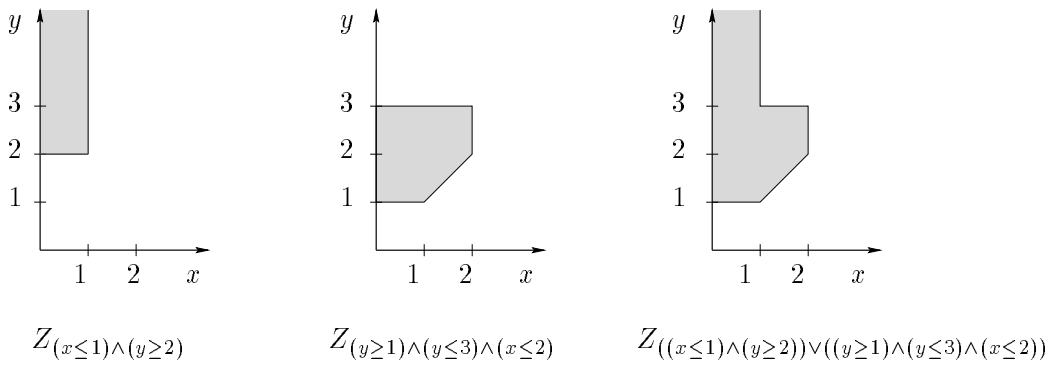


FIG. 2.3 – *Zones associées à différentes contraintes*

2.2.2 Définition d'un automate temporisé

Définition 2.2.14 *Un* **automate temporisé** *A est un quintuplet (Σ, Q, q_0, X, T), où :*

- *Σ est un alphabet fini.*
- *Q est un ensemble fini d'états.*

- $q_0 \in Q$ *est l'état initial.*
- X *est un ensemble fini d'horloges,* $\mathcal{P}(X)$ *est l'ensemble des parties de* X*.*
- $T \subset Q \times \mathcal{C}(X) \times \Sigma \times \mathcal{P}(X) \times Q$ *est un ensemble de transitions. Soit* $(q, \delta, a, \lambda, q') \in T$ *une transition,* $\delta \in \mathcal{C}(X)$ *est la garde sur les horloges et* $\lambda \in \mathcal{P}(X)$ *est l'ensemble des horloges à remettre à zéro.*

Remarque 2.2.15 Nous avons défini ici une première sorte de transitions associées à un automate temporisée, ce sont les **transitions de l'automate**. Dans ce qui va suivre nous allons définir d'autres transitions : les **transitions d'états** et les **transitions de temps**.

Remarque 2.2.16 La transition $(q, \delta, a, \lambda, q')$ est également notée $q \overset{(\delta, a, \lambda:=0)}{\longrightarrow} q'$.

Exemple 2.2.17 (Une minuterie)
L'automate de la figure 2.4 représente le comportement d'une minuterie.

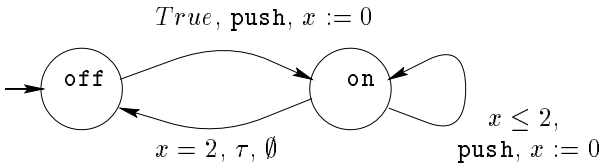


FIG. 2.4 – *Exemple d'automate temporisé : une minuterie*

L'alphabet de cet automate est $\{\texttt{push}, \tau\}$, son ensemble d'états $\{\texttt{on}, \texttt{off}\}$, son état initial on, son ensemble d'horloges $\{x\}$ et son ensemble de transitions $\{(\texttt{off}, True, \texttt{push}, x := 0, \texttt{on}), (\texttt{on}, x = 2, \tau, \emptyset, \texttt{off}), (\texttt{on}, x \leq 2, \texttt{push}, x := 0, \texttt{on})\}$.

2.2.3 Système de transitions associé à un automate temporisé

Voici la sémantique associée aux automates temporisés :

Définition 2.2.18 *Une* **configuration** *de l'automate temporisé* A *est un couple formé d'un état et d'une valuation d'horloges. La configuration initiale* $s_{A,0}$ *est* (q_0, v_0)*, où* $v_0(x) = 0$*, pour tout* x *de* X*. L'ensemble des configurations de* A *est noté* S_A*.*

Exemple 2.2.19 Dans l'exemple 2.2.17, (on, 2.5) est une configuration et (off, 0) la configuration initiale de l'automate.

Définition 2.2.20 *Soit* $A = (\Sigma, Q, q_0, X, T)$ *un automate temporisé. Les transitions entre deux configurations de* S_A *sont de deux types :*

- *Les* **transitions de temps**. *Soient* $(q, v) \in S_A$ *et* $d \in \mathbb{T}$*,* $(q, v) \xrightarrow{d} (q, v + d)$ *est une transition de temps,*
- *Les* **transitions d'états**. *Soient* $(q, \delta, a, \lambda, q') \in T$ *et* $v \in \mathbb{T}^X$ *tels que* $v \vDash \delta$*. Soit* $v' = v[\lambda \leftarrow 0]$*,* $(q, v) \xrightarrow{a} (q', v')$ *est une transition d'états.*

L'ensemble des transitions entre deux configurations de S_A *est noté* $\rightarrow_A$*.*

Remarque 2.2.21 Dans la définition précédente, dans les transitions de temps, d peut être nul.

Exemple 2.2.22 $(\texttt{on}, 3.1) \xrightarrow{2.3} (\texttt{on}, 5.4)$ est une transition de temps du système de transitions associé à l'automate de la figure 2.4. $(\texttt{off}, 3.1) \xrightarrow{\texttt{push}} (\texttt{on}, 0)$ et $(\texttt{on}, 2) \xrightarrow{\tau} (\texttt{off}, 2)$ sont des transitions d'états.

Définition 2.2.23 *Soit $A = (\Sigma, Q, q_0, X, T)$ un automate temporisé. Le système de transitions $\mathcal{S}_A = (S_A, s_{A,0}, \Sigma \cup \mathbb{T}, \rightarrow_A)$ est le* **système de transitions associé** *à A.*

On notera $\mathcal{A}ct$ l'union de l'ensemble des actions Σ et l'ensemble des délais $\mathbb{T}$.

Remarque 2.2.24 Il y a donc deux sortes de "transition" :

1. Les transitions de l'automate, elles appartiennent à l'ensemble $T \subset Q \times \mathcal{C}(X) \times \Sigma \times \mathcal{P}(X) \times Q$,
2. Les transitions du système de transitions, elles appartiennent à l'ensemble $\rightarrow_A \subset S_A \times (\Sigma \cup \mathbb{T}) \times S_A$.

Remarque 2.2.25 Soit A un automate temporisé et $\mathcal{S}_A$ son système de transitions associé. Une exécution de A est donc

$$(q_0, v_0) \xrightarrow{\ell_1}_A (q_1, v_1) \xrightarrow{\ell_2}_A (q_2, v_2) \xrightarrow{\ell_3}_A \ldots$$

où $q_i \in Q$, $v_i \in \mathbb{T}^X$ et $\ell_i \in (\Sigma \cup \mathbb{T})$.
$\bar{\ell} = \ell_1 \ell_2 \ldots$ en est la trace.

Définition 2.2.26 *Soit A un automate temporisé, A reconnait un langage L si L est le langage du système de transitions associé à A.*

Nous pouvons alors définir la bisimilarité de deux automates temporisés.

Définition 2.2.27 *Soient $\mathcal{A}$ et $\mathcal{B}$ deux automates temporisés.*
A est **bisimilaire** *à B si et seulement si $\mathcal{S}_A$ est bisimilaire à $\mathcal{S}_B$.*

Chapitre 3

Automates temporisés avec invariants ou conditions limites

3.1 Les invariants

Avec les définitions présentées dans la partie précédente, l'automate peut rester dans un état indéfiniment et ne jamais franchir de transition d'états. Par exemple dans la figure 2.4, on peut avoir le début d'exécution suivant $(\texttt{off}, 0) \xrightarrow{\texttt{push}} (\texttt{on}, 0) \xrightarrow{3} (\texttt{on}, 3)$. Une fois dans l'état $(\texttt{on}, 3)$, on ne peut plus faire de transition d'états, les seules transitions possibles sont les transitions de temps. Donc la fin de l'exécution ne présentera que des transitions de temps.

Pour remédier à ce problème, le modèle théorique de Alur et Dill ajoute des conditions d'acceptation (de Büchi ou de Muller) pour savoir si un mot infini appartient au langage associé à l'automate temporisé. Avec ce modèle, on ne peut dire qu'une exécution est rejetée par un automate qu'une fois celle-ci achevée. Les outils de model-checking ont besoin de rejeter une exécution qui n'est pas reconnu pas le système le plus tôt possible. Aussi ont-ils eu recours à un moyen différent pour assurer qu'un automate ne reste indéfiniment dans le même état : ils associent un **invariant** $I(q)$ à chaque état q de l'automate. L'idée informelle de l'utilisation de ces invariants est qu'on ne peut "rester" dans une configuration (q, v) que si v satisfait $I(q)$. Sinon, le système doit quitter q. Ici "rester" signifie pouvoir ne faire que des transitions de temps.

Définition 3.1.1
Un **automate temporisé avec invariants** A *est un sextuplet* $(\Sigma, Q, q_0, X, T, I)$*, où :*

- (Σ, Q, q_0, X, T) *est un automate temporisé,*
- I *est une application de* Q *dans* $\mathcal{C}(X)$*.*

Exemple 3.1.2 (suite de la minuterie)
Dans l'exemple de la figure 2.4, on rajoute l'invariant $I(\texttt{on}) = x \leq 2$ pour empêcher le système de rester plus de 2 unités de temps dans l'état `on`. On obtient alors la figure 3.1.

Remarque 3.1.3 De même que les gardes *True* et les resets $\emptyset$ sont souvent omis des figures, les invariants *True* n'apparaissent pas dans les états.

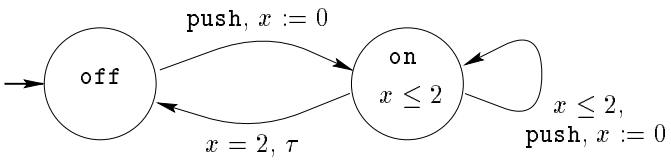


FIG. 3.1 – *une minuterie avec invariants*

Dans la littérature et dans les outils de model-checking, plusieurs interprétations des invariants sont données. Une première divergence concerne les configurations : est-ce que (q, v) avec $v \not\models I(q)$ est une configuration? Ce qui entraîne deux questions concernant la définition des transitions des systèmes de transitions associés :

Q1 : Lors des transitions de temps, le système peut-il rester dans un état q jusqu'à l'instant précis où il viole son invariant pour la première fois, ou doit-il l'avoir quitté avant?

Q2 : Lors d'une transition d'états, le système doit-il vérifier l'invariant de l'état d'arrivée?

Dans la littérature, nous avons trouvé deux modèles qui répondent à ces questions : les automates que nous dirons de "de Deadline" et ceux que nous dirons de "de Uppaal".

3.2 Différentes interprétations des invariants

3.2.1 Automates de Uppaal

Définition 3.2.1 *Soit $A = (\Sigma, Q, q_0, X, T, I)$ un automate temporisé avec invariants. A est un automate dit de Uppaal si on lui associe le système de transitions $\mathcal{S}_{A,Upp} = (S_{A,Upp}, \rightarrow_{A,Upp})$ défini par :*

- $(q, v) \in S_{A,Upp}$ *si et seulement si* $v \models I(q)$,
- $(q, v) \xrightarrow{d}_{A,Upp} (q, v + d)$ *est une transition de temps si et seulement si* $\forall t,\ 0 \leq t \leq d,\ v + t \models I(q)$
- $(q, v) \xrightarrow{a}_{A,Upp} (q', v')$ *est une transition d'états si et seulement si* $\exists\ (q, \delta, a, \lambda, q') \in T,\ v \models \delta,\ v' = v[\lambda \leftarrow 0]$ *et* $v' \models I(q')$.

Ainsi, un automate de Uppaal répond "avant" et "oui" à Q1 et Q2.
Cette interprétation est implantée dans Uppaal et dans Kronos [DY95, DOTY96].

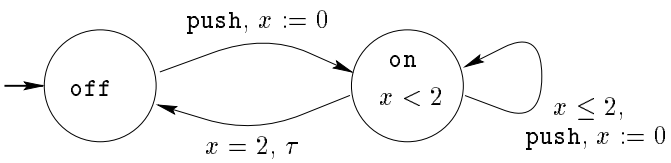


FIG. 3.2 – *une troisième version de la minuterie*

Exemple 3.2.2 (une troisième version de la minuterie)
L'automate de la figure 3.2 présente une nouvelle modélisation de notre minuterie. Il va permettre de montrer les différences entre les deux interprétations d'automates avec invariant. En fait, ce modèle n'a le comportement attendu pour une minuterie dans aucune des deux interprétations. Si cet automate est un automate de Uppaal, $(\texttt{off},0) \xrightarrow{\texttt{push}} (\texttt{on},0) \xrightarrow{2} (\texttt{on},2)$ et $(\texttt{off},0) \xrightarrow{\texttt{push}} (\texttt{on},0) \xrightarrow{2} (\texttt{on},2) \xrightarrow{\tau} (\texttt{off},2)$ ne sont pas des exécutions de l'automate. En effet $(\texttt{on},0) \xrightarrow{2} (\texttt{on},2)$ n'est pas une transition de temps car $2 \not\models I(\texttt{on})$.

3.2.2 Automates de Deadline

Définition 3.2.3 *Soit $A = (\Sigma, Q, q_0, X, T, I)$ un automate temporisé avec invariants. A est un automate dit de Deadline si on lui associe le système de transitions $\mathcal{S}_{A,Dead} = (S_{A,Dead}, \rightarrow_{A,Dead})$ défini par :*

- $S_{A,Dead} = Q \times \mathbb{T}^X$,
- $(q,v) \xrightarrow{d}_{A,Dead} (q, v+d)$ *est une transition de temps si et seulement si* $\forall t,\ 0 \leq t < d,\ v+t \models I(q)$
- $(q,v) \xrightarrow{a}_{A,Dead} (q',v')$ *est une transition d'états si et seulement si* $\exists\ (q,\delta,a,\lambda,q') \in T,\ v \models \delta$ *et* $v' = v[\lambda \leftarrow 0]$.

Un automate de Deadline opte pour la première solution de Q1 et répond "non" à Q2.
Cette interprétation est présentée dans [SY96, BS97]. L'idée sous-jacente, est que "si le système est dans un état (q,v) tel que v ne satisfait pas v, alors le temps s'arrête jusqu'à ce qu' une transition d'états soit franchie". Yovine et Sifakis sous-entendent que cette interprétation permet de représenter plus de systèmes que celle de Uppaal.

Le choix du nom d'automates de Deadline est expliqué plus loin.

Exemple 3.2.4 Si l'automate de la figure 3.2 est un automate de Deadline, $(\texttt{off},0) \xrightarrow{\texttt{push}} (\texttt{on},0) \xrightarrow{2} (\texttt{on},2)$ et $(\texttt{off},0) \xrightarrow{\texttt{push}} (\texttt{on},0) \xrightarrow{2} (\texttt{on},2) \xrightarrow{\tau} (\texttt{off},2)$ sont des exécutions du système. Comme $2 \not\models I(\texttt{on})$ le système ne peut pas rester dans l'état `on` mais, il peut faire une transition d'états.

3.3 Structures des ensembles de valuations

Dans les trois parties suivantes, nous construisons, étant donné un automate de Deadline A, un automate de Uppaal B, tels que leurs systèmes de transitions soient bisimilaires. Une première idée, consiste à garder les états de A pour B et à modifier les invariants afin que si $\forall t < d,\ v+t \models I_A(q)$, alors $\forall t \leq d,\ v+t \models I_B(q)$.

Nous allons donc définir une opération de "fermeture à droite" sur les contraintes d'horloges, F_d telle que si $\forall t \in [0,d[,\ v+t \models \phi$, alors $\forall t \in [0,d],\ v+t \models F_d(\phi)$ et r'eciproquement. Pour vérifier que la fermeture à droite d'une contrainte d'horloges est bien une contrainte d'horloges, nous définissons une propriété de "convexité pour le temps".

D'autre part, nous serons amenés à distinguer une autre propriété sur les contraintes d'horloges, la "clôture par le passé", la justification de ce choix est donn'ee dans la partie 3.5.1. Toutes ces nouvelles notions : clôture par le passé, fermeture à droite et convexité pour le temps, sont présentées dans cette partie.

3.3.1 Clôture par le passé

Définition 3.3.1
Soit X un ensemble d'horloges,

- *soit V un ensemble de valuation d'horloges, V est* **clos par le passé** *si :*

$$\forall v \in \mathbb{T}^X \text{ tel que } v \notin V, \text{ alors } \forall t \geq 0,\ v + t \notin V.$$

- *Une contrainte d'horloges ϕ est close par le passé si Z_ϕ ,la zone associée à ϕ, est close par le passé.*

Proposition 3.3.2 *Les contraintes d'horloges définies par la grammaire :*

$$A ::= x \sim t |\ x - y \sim t |\ A \wedge A |\ A \vee A |\ True$$

où $\sim \in \{<, \leq\}$, sont closes par le passé.

Corollaire 3.3.3 *Les invariants de l'outil de model-checking Uppaal qui doivent être construits sur la grammaire :*

$$A ::= x \sim t |\ A \wedge A |\ True$$

où $\sim \in \{<, \leq\}$ sont clos par le passé.

Exemple 3.3.4 Si $X = (x, y)$, $Z_{(x-y<1)\vee(y\leq 2)}$ est close par le passé. Elle est représentée dans la figure 3.3.

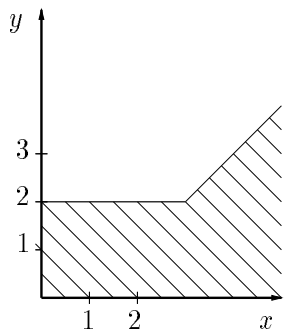


FIG. 3.3 – $Z_{(x-y<1)\vee(y\leq 2)}$

3.3.2 Convexité pour le temps

Définition 3.3.5
Soit X un ensemble d'horloges,

- *soit V un ensemble de valuation d'horloges, V est* **convexe pour le temps** *si :*

$$\forall v \in V,\ \forall d \geq 0,\ si\ v + d \in V,\ alors\ \forall t \in [0, d],\ v + t \in D.$$

- *Une contrainte d'horloges ϕ est convexe pour le temps si Z_ϕ,la zone associée à ϕ, est convexe pour le temps.*

Proposition 3.3.6 *Soit V un ensemble de valuation d'horloges, si V est clos par le passé, alors V est convexe pour le temps.*

Démonstration : On raisonne par l'absurde.
Supposons que V ne soit pas convexe paour le temps. Alors il existe une valuation v, deux réels d et $t \in [0, d]$ tels que v et $v + d$ appartiennent à V mais pas $v + t$. Or $t \leq d$ et $v + d$ est dans V, donc V n'est pas clos par le passé. □

Proposition 3.3.7 *Les contraintes d'horloges définies par la grammaire :*

$$A ::= x \sim t |\ x - y \sim t |\ A \wedge A |\ A \vee A |\ True$$

où $\sim \in \{<, \leq\}$ sont convexes pour le temps. En particulier, les invariants de l'outil de model-checking Uppaal et les contraintes d'horloges positives sont convexes pour le temps.

Démonstration : C'est une conséquence des propositions 3.3.2 et 3.3.7 □

Exemple 3.3.8 $Z_{(x-y<1)\vee(y\leq 2)}$, représentée dans la figure 3.3 est convexe pour le temps.

Corollaire 3.3.9 *Les invariants de l'outil de model-checking Uppaal et les contraintes d'horloges positives sont convexes pour le temps.*

3.3.3 Fermeture à droite

Définition 3.3.10 *Un ensemble de valuations V est* **fermé à droite** *si*

$$\forall v \in \mathbb{T}^X,\ \forall d > 0, ((\forall t \in [0, d[,\ v + t \in V) \Rightarrow (v + d \in V))$$

Définition 3.3.11 *Soit V un ensemble de valuations, la* **fermeture à droite** *de V, notée $F_d(V)$ est la plus petite des extensions de V fermées à droite.*

Exemple 3.3.12 $x < 1$ n'est pas fermée à droite.

Définition 3.3.13 *Soit V un ensemble de valuations, le* **complément à droite** *de V, noté $f_d(V)$ est $\{v + \eta,\ \eta > 0$ et $\forall t \in [0, \eta[,\ v + t \in V\}$.*

Proposition 3.3.14 *Soit ϕ une contrainte d'horloges. Soit $V = Z_\phi$. $F_d(V)$ est l'union de V et de $f_d(V)$.*

Démonstration : Soit W l'union de V et de $f_d(V)$. Soit U une extension de V fermée à droite, W est inclus dans U car $f_d(V)$ est inclus dans U. Donc W est inclus dans $F_d(V)$.
Montrons que W est fermé à droite. Comme W contient V, on aura que W est égal à $F_d(V)$.

Soit v une valuation et η un réel strictement positif tel que $\forall t \in [0, \eta[,\ v + t \in W$.

Si $v + t \notin V$, soit $\eta_1 = sup\{t \in [0, \eta[,\ v + t \in V\}$. Si $\eta_1 \neq \eta$, alors $\forall t \in]\eta_1, \eta[,\ v + t \notin V$. Alors $\forall t \in]\eta_1, \eta[,\ v + t \notin f_d(V)$, contradiction avec $v + t \in W$. Donc $\eta_1 = \eta$.

Soit $\mathbb{1}_V$ la fonction caractéristique de V. Comme V est la zone d'une contrainte d'horloge, la fonction : $t \mapsto \mathbb{1}_V(v + t)$ n'a qu'un nombre fini de points singuliers. Donc il existe $\eta' < \eta$

tel que pour tout t de $]\eta', \eta[$, $\mathbb{1}_V(v+t)$ a la même valeur ($t \mapsto \mathbb{1}_V(v+t)$ prends ses valeurs dans $\{0,1\}$). Si cette valeur est 0, alors $\forall t \in]\eta', \eta[,\ v+t \notin V$. Alors $\forall t \in]\eta', \eta[,\ v+t \notin f_d(V)$, contradiction avec $v+t \in W$. Donc $\forall t \in]\eta', \eta[,\ v+t \in V$, d'où $v+\eta \in f_d(V)$. Donc $v+\eta \in W$.

□

Tout le problème, maintenant, est de savoir si, étant donnée une contrainte sur les horloges ϕ, il existe une autre contrainte sur les horloges ψ telle que $Z_\psi = F_d(Z_\phi)$.

Proposition 3.3.15 *Soient V_1 et V_2 deux ensembles de valuations d'horloges.*

$$f_d(V_1 \cap V_2) = f_d(V_1) \cap f_d(V_2).$$

Démonstration :

- Soit $v \in f_d(V_1 \cap V_2)$,

  alors il existe $\eta > 0$ et $v' \in (V_1 \cap V_2)$ tels que, $v = v' + \eta$ et $\forall t \in [0, \eta[,\ v'+t \in (V_1 \cap V_2)$.

  Donc $v \in f_d(V_1)$ et $v \in f_d(V_2)$.

- Soit $v \in (f_d(V_1) \cap f_d(V_2))$. $v \in f_d(V_1)$ donc il existe $\eta' > 0$ et $v' \in V_1$ tels que, $\forall t \in [0, \eta'[,\ v'+t \in V_1$ et $v = v' + \eta'$. De même, il existe $\eta'' > 0$ et $v' \in V_1$ tels que, $\forall t \in [0, \eta''[,\ v''+t \in V_2$ et $v = v'' + \eta''$. Soit $\eta = inf(\eta', \eta'')$, par exemple $\eta = \eta'$, $\forall t \in [0, \eta'[,\ v'+t = v''+\eta''-\eta'+t$. Or $\eta''-\eta'+t \in [\eta''-\eta', \eta''[$, donc $v'+t \in (V_1 \cap V_2)$. Donc $v \in f_d(V_1 \cap V_2)$, $\eta > 0$.

□

Proposition 3.3.16 *Soient V_1 et V_2 deux ensembles de valuations d'horloges convexes pour le temps.*

$$f_d(V_1 \cup V_2) = f_d(V_1) \cup f_d(V_2).$$

Conséquence 3.3.17 *Soient V_1 et V_2 deux ensembles de valuations d'horloges convexes pour le temps.*

$$F_d(V_1 \cup V_2) = f_d(V_1) \cup f_d(V_2) \cup V_1 \cup V_2.$$

Démonstration :

- Soit $v \in f_d(V_1 \cup V_2)$,

  alors il existe $\eta > 0$ et $v' \in (V_1 \cup V_2)$ tels que, $\forall t \in [0, \eta[,\ v'+t \in (V_1 \cup V_2)$ et $v = v' + \eta$.

  Soit $t_1 = sup\{t \in [0, \eta[$ tel que $v'+t \in V_1\}$ et $t_2 = sup\{t \in [0, \eta[$ tel que $v'+t \in V_2\}$.

  Supposons que $t_1 \leq t_2$, alors $t_2 = \eta$. Sinon, soit $\alpha \in]t_2, \eta[\ v+\alpha \notin (V_1 \cup V_2)$ (contradiction).

  Soit $t_0 = inf\{t \in [0, \eta[$ tel que $v'+t \in V_2\} \cup \{\eta\}$.

  - si $t_0 = \eta$, alors $\forall t \in [0, \eta[,\ v'+t \in V_1$ donc $v \in f_d(V_1)$ car $\eta > 0$.
  - si $t_0 \neq \eta$, alors $t_0 < \eta$ et $\forall t \in [0, \eta - t_0[,\ v'+t+t_0 \in V_2$ donc $v \in f_d(V_2)$ car $\eta - t_0$.

Dans tous les cas, $v \in f_d(V_1) \cup f_d(V_2)$.

- Soit $v \in (f_d(V_1) \cup f_d(V_2))$.

  Si, par exemple, $v \in f_d(V_1)$ alors il existe η' et $v' \in V_1$ tels que, $\forall t \in [0, \eta'[,\ v' + t \in V_1$ et $v = v' + d$. $\forall t \in [0, \eta'[,\ v' + t \in (V_1 \cup V_2)$. Donc $v \in f_d(V_1 \cup V_2)$.

□

On obtient un résultat analogue en généralisant

Proposition 3.3.18 *Soient $V_1 ... V_k$ des ensembles de valuations d'horloges convexes pour le temps.*

$$f_d(V_1 \cup ... \cup V_k) = f_d(V_1) \cup ... \cup f_d(V_k).$$

$$F_d(V_1 \cup ... \cup V_k) = (V_1 \cup ... \cup V_k) \cup f_d(V_1) \cup ... \cup f_d(V_k).$$

Conséquence 3.3.19 *Soit ϕ une contrainte d'horloges, $\phi \equiv \phi_1 \vee \phi_2 \vee ... \vee \phi_k$ où les ϕ_j sont des contraintes d'horloges positives. Alors $F_d(Z_\phi) = Z_{\phi_1} \cup Z_{\phi_2} \cup ... \cup Z_{\phi_k} \cup f_d(Z_{\phi_1}) \cup f_d(Z_{\phi_2}) \cup ... \cup f_d(Z_{\phi_k})$.*

Démonstration : Les contraintes d'horloges peuvent s'écrire comme des disjonctions de contraintes d'horloges positives (cf proposition 2.2.11). Or les zones associées aux contraintes d'horloges positives sont convexes, donc convexes pour le temps. □

Ceci va nous permettre, **étant donnée une contrainte sur les horloges ϕ, de lui associer une autre contrainte sur les horloges ψ telle que $Z_\psi = F_d(Z_\phi)$.**

- On écrit ϕ comme disjonction de contraintes d'horloges positives $\phi \equiv \phi_1 \vee \phi_2 \vee ... \vee \phi_k$. Alors $F_d(Z_\phi) = Z_{\phi_1} \cup Z_{\phi_2} \cup ... \cup Z_{\phi_k} \cup f_d(Z_{\phi_1}) \cup f_d(Z_{\phi_2}) \cup ... \cup f_d(Z_{\phi_k})$.
- Pour chaque ϕ_j, on calcule $f_d(Z_{\phi_j})$, en appliquant les règles suivantes :
  - $f_d(Z_{x<t}) = Z_{0<x\leq t}$.
  - $f_d(Z_{x\leq t}) = Z_{0<x\leq t}$.
  - $f_d(Z_{x=t}) = Z_{false}$.
  - $f_d(Z_{x\geq t}) = Z_{x>t}$.
  - $f_d(Z_{x>t}) = Z_{x>t}$.
  - $f_d(Z_{x-y\sim t}) = Z_{(x-y\sim t)\wedge(x\neq 0)\wedge(y\neq 0)}$.
  - $f_d(Z_1 \cap Z_2) = f_d(Z_1) \cap f_d(Z_2)$.

  où $\sim \in \{<, \leq\},\ x \in X, y \in X$ et Z_1 et Z_2 sont des contraintes d'horloges.
- On regroupe ensuite les termes en utilisant les règles suivantes :
  - $Z_{(\phi_1 \vee \phi_2)} = Z_{\phi_1} \cup Z_{\phi_2}$
  - $Z_{(\phi_1 \wedge \phi_2)} = Z_{\phi_1} \cap Z_{\phi_2}$

Exemple 3.3.20
Si $\phi = (x \neq 3)$.

$$\begin{aligned} F_d(Z_\phi) &= F_d(Z_{(x>3)\vee(x<3)}) \\ &= F_d(Z_{(x>3)} \cup Z_{(x<3)}) \\ &= Z_{(x>3)} \cup Z_{(x<3))} \cup f_d(Z_{(x>3)}) \cup f_d(Z_{(x<3)}) \\ &= Z_{(x>3)} \cup Z_{(x<3)} \cup Z_{(x>3)} \cup Z_{(x\leq 3)} \\ &= Z_{(x>3)\vee(x<3)\vee(x>3)\vee(x\leq 3)}. \\ &= Z_{True}. \end{aligned}$$

Donc $\psi = True$ convient.

Définition 3.3.21 *ψ est appelée fermeture à droite de ϕ et notée $F_d(\phi)$.*

Lemme 3.3.22 *Soit ϕ une contrainte d'horloge, alors $\forall v \in \mathbb{T}^X$,*

$$v \vDash F_d(\phi) \text{ si et seulement si}$$

$$v \vDash \phi \text{ ou il existe } \eta > 0 \text{ et } v' \in \mathbb{T}^X \text{ tels que } v = v' + \eta \text{ et } \forall t \in [0, \eta[,\ v' + t \vDash \phi.$$

Démonstration : C'est une conséquence de $Z(F_d(\phi)) = F_d(Z_\phi)$. □

Lemme 3.3.23 *Soient ϕ une contrainte d'horloges et v une valuation d'horloges. Si $\forall t \in [0, d[,\ v + t \vDash \phi$, alors $\forall t \in [0, d],\ v + t \vDash F_d(\phi)$.*

Démonstration : $\forall t \in [0, d[,\ v + t \vDash \phi$, donc $\forall t \in [0, d[,\ v + t \in Z_\phi$. D'où $\forall t \in [0, d],\ v + t \in f_d(Z_\phi)$. Par conséquent, $\forall t \in [0, d],\ v + t \in F_d(Z_\phi)$. D'où, $\forall t \in [0, d],\ v + t \vDash F_d(\phi)$. □

Lemme 3.3.24 *Soient ϕ une contrainte d'horloges fermée par le passé et v une valuation d'horloges. Si $\forall t \in]0, d],\ v + t \vDash F_d(\phi)$, alors $\forall t \in [0, d[,\ v + t \vDash \phi$.*

Démonstration : (démonstration par l'absurde)

Soit $t_0 \in [0, d[$, tel que $v + t_0 \nvDash \phi$. Donc $v + t_0 \notin Z_\phi$. Alors $\forall t \geq t_0,\ v + t \notin Z_\phi$ (ϕ est close par le passé).

Soit $t_1 \in]t_0, d[$, $v + t_1 \notin Z_\phi$ et $v + t_1 \vDash F_d(\phi)$, donc $v + t_1 \in F_d(Z_\phi)$.

Donc il existe $\eta > 0$ et $v' \in \mathbb{T}^X$ tels que $v + t_1 = v' + \eta$, et $\forall t \in [0, \eta[,\ v' + t \in Z_\phi$. On peut supposer, sans perdre de généralité, que $\eta < t_1$.

Donc $\forall t \in [(t_1 - \eta), t_1[,\ v + t \in Z_\phi$. Soit $\alpha = sup((t_1 - \eta), t_0)$, $\alpha \in [(t_1 - \eta), t_1[$, donc $v + \alpha \in Z_\phi$. Contradiction avec $\forall t \geq t_0,\ v + t \notin Z_\phi$. □

3.4 Simulation des automates de Deadline par des automates de Uppaal, dans le cas où les invariants sont clos par le passé

3.4.1 Introduction

Etant donné un automate de Deadline A, nous souhaitons exhiber un automate de Uppaal B tel que leurs systèmes de transitions soient bisimilaires.

Si on regarde le dernier exemple de la minuterie, figure 3.2, une première idée, consiste à

- garder les états de A pour B,
- modifier les invariants afin que si $\forall t < d,\ v + t \vDash I_A(q)$, alors $\forall t \leq d,\ v + t \vDash I_B(q)$.

Ce qui ici renvient à remplacer l'invariant de `on` par $x \leq 2$.

De plus, on introduit une horloge supplémentaire y. Cette horloge va permettre de faire des transitions d'états telles que la valuation des horloges d'arrivée ne vérifie pas l'invariant I_A de l'état d'arrivée mais impose dans ce cas de ne pas rester dans l'état d'arrivée.

En fait, pour que cette construction convienne, nous devons restreindre les formules sur les invariants, ils doivent être clos par le passé (la justification est faite dans la partie suivante). Une construction ne demandant pas de restriction sur les invariants mais faisant croître de façon exponentielle le nombre d'états est proposée plus loin.

3.4.2 Automate de Uppaal associé

Soit $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, I_A)$ un automate de Deadline dont les invariants sont clos par le passé.

On considère l'automate de Uppaal suivant $B = (\Sigma, Q_B, q_{B,0}, X_B, T_B, I_B)$ définie par :

- $Q_B \stackrel{def}{=} Q_A$,
- $q_{B,0} \stackrel{def}{=} q_{0,A}$,
- $X_B \stackrel{def}{=} X_A \cup \{y\}$,
- T_B est défini de la façon suivante :

$$(q, \delta, a, \lambda \cup \{y\}, q') \in T_B \text{ si et seulement si } (q, \delta, a, \lambda, q') \in T_A$$

- I_B est défini de la façon suivante :

$$I_B(q) \stackrel{def}{=} F_d(I_A(q)) \vee (y = 0)$$

Remarque 3.4.1 On a bien $v_0 \vDash I_B(q_0)$.

Lemme 3.4.2 *Soit q appartenant à Q, si $I_A(q)$ est fermé à droite, alors $I_B(q)$ est fermé à droite.*

Démonstration : Soit q appartenant à Q, $I_A(q) \equiv \phi_1 \vee \phi_2 \vee ... \vee \phi_k$ où les ϕ_j sont des contraintes d'horloges positives. Donc $f_d(Z_{I_A(q)}) = f_d(Z_{\phi_1}) \cup f_d(Z_{\phi_2}) \cup ... \cup f_d(Z_{\phi_k})$.
Donc $(I_A(q) \vee (y = 0)) \equiv \phi_1 \vee \phi_2 \vee ... \vee \phi_k \vee (y = 0)$ et $f_d(Z_{I_A(q) \vee (y=0)}) = f_d(Z_{\phi_1}) \cup f_d(Z_{\phi_2}) \cup ... \cup f_d(Z_{\phi_k}) \cup f_d(Z_{y=0})$ car $Z_{y=0}$ est convexe.
Donc $f_d(Z_{I_A(q) \vee (y=0)}) = f_d(Z_{I_A(q)}) \cup f_d(Z_{y=0})$. D'où $f_d(Z_{I_A(q) \vee (y=0)}) = f_d(Z_{I_A(q)})$ car $f_d(Z_{y=0}) = \emptyset$. Par conséquent, $F_d(Z_{I_A(q) \vee (y=0)}) = F_d(Z_{I_A(q)}) \cup Z_{y=0}$.
Donc $F_d(Z_{I_A(q) \vee (y=0)}) = Z_{I_B(q)}$. Donc $I_B(q)$ est fermé à droite car la fermeture à droite d'un ensemble est fermée à droite. □

3.4.3 Les automates ont des systèmes de transitions bisimilaires

Théorème 3.4.3
Le système de transition de A, $\mathcal{S}_A = (S_A, s_{A,0}, \rightarrow_A)$ est bisimilaire au système de transition de B, $\mathcal{S}_B = (S_B, s_{B,0}, \rightarrow_B)$.

Démonstration : Soit $\mathcal{R}$ la relation incluse dans $S_A \times S_B$ définie par : $((q,v),(q',v')) \in \mathcal{R}$ si et seulement si :

1. $q = q'$,
2. $v'_{|X} = v$ i.e. $\forall x \in X,\ v'(x) = v(x)$,
3. $v' \vDash I_B(q)$ i.e. $v'(y) = 0$ ou $v \vDash F_d(I_A)$.

Dans ce cas, $(q,v') \in S_B$ car $v' \vDash I_B(q)$. Montrons que $\mathcal{R}$ est bien une bisimulation :
Tout d'abord, $(s_{A,0}, s_{B,0}) \in \mathcal{R}$.
Ensuite, soit $((q,v),(q,v')) \in \mathcal{R}$, $v' = v \cup \{y = \alpha\}$, $\alpha = 0$ ou $v \vDash I_A(q)$.

- Soient $(q_1, v_1) \in S_A$ et $\ell \in \mathcal{A}ct$ tels que $(q,v) \xrightarrow{\ell}_A (q_1, v_1)$,

  1. $\ell \in \mathbb{T}$, alors $q = q_1$, $v_1 = v + \ell$ et $\forall t \in [0, \ell[,\ v + t \vDash I_A(q)$.
     Soit $v'_1 = v' + \ell$, d'après le lemme 3.3.23, $\forall t \in [0,\ell],\ v + t \vDash F_d(I_A(q))$. Donc $\forall t \in [0,\ell],\ v' + t \vDash I_B(q)$ en particulier, $v'_1 \vDash I_B(q)$ donc $(q_1, v'_1) \in S_B$. De plus, $(q,v') \xrightarrow{\ell}_B (q_1, v'_1)$ et $((q_1,v_1),(q_1,v'_1)) \in \mathcal{R}$ car $(v'_1)_{|X} = v_1$.
  2. $\ell \in \Sigma$, alors $\exists (q, \delta, \ell, \lambda, q_1) \in T_A$ tel que $v \vDash \delta$ et $v_1 = v[0 \leftarrow \lambda]$.
     Soit $v'_1 = v' \cup \{y = 0\}$, $v'_1 \vDash I_B(q_1)$ donc $(q_1, v'_1) \in S_B$ et $((q_1,v_1),(q_1,v'_1)) \in \mathcal{R}$. D'autre part, $(q, \delta, \ell, \lambda \cup \{y\}, q_1) \in T_B$, $v' \vDash \delta$, $v'_1 = v'[0 \leftarrow \lambda \cup \{y\}]$ et $v'_1 \vDash I_B(q_1)$ donc $(q,v') \xrightarrow{\ell}_B (q_1, v'_1)$.

- Soient $(q_1, v'_1) \in S_B$ et $\ell \in \mathcal{A}ct$ tels que $(q,v') \xrightarrow{\ell}_B (q_1, v'_1)$,

  Soit $v_1 = (v'_1)_{|X}$, $((q_1,v_1),(q_1,v'_1)) \in \mathcal{R}$ donc reste à voir si $(q,v) \xrightarrow{\ell}_A (q_1, v_1)$.

  1. $\ell \in \mathbb{T}$, alors $q = q_1$, $v'_1 = v' + \ell$ et $\forall t \in [0,\ell],\ v' + t \vDash I_B(q)$.
     Si $t \neq 0$, $v' + t(y) \neq 0$, donc $v' + t \vDash F_d(I_A(q))$, d'où $\forall t \in]0,\ell],\ v + t \vDash F_d(I_A(q))$.
     D'après le lemme 3.3.24, $\forall t \in [0, \ell[,\ v + t \vDash (I_A(q))$. Donc $(q,v) \xrightarrow{\ell}_A (q_1, v_1)$.
  2. $\ell \in \Sigma$, alors $\exists (q, \delta, \ell, \lambda \cup \{y\}, q_1) \in T_A$ tel que $v' \vDash \delta$ et $v'_1 = v'[0 \leftarrow \lambda \cup \{y\}]$.
     Donc $(q, \delta, \ell, \lambda, q_1) \in T_A$, $v \vDash \delta$ et $v_1 = v[0 \leftarrow \lambda]$. D'où $(q,v) \xrightarrow{\ell}_A (q_1, v_1)$.

□

3.5 Ensembles particuliers de valuations d'horloges

3.5.1 Présentation du problème

Dans la partie précédente, nous avons limité les formules des invariants. On peut se demander ce qui se passe si on ne le fait pas et si l'on garde la même construction. Par exemple si on a un invariant $I(q) = x \neq 3$, alors la fermeture à droite de $I(q)$ est $F_d(x \neq 3) = True$. Si on interprète $I(q)$ comme un invariant de Deadline et $F_d(I(q))$ comme un invariant de Uppaal, dans les deux cas, on peut rester dans l'état q jusqu'à ce que $x = 3$. Mais avec $F_d(I(q))$, la barrière $x = 3$ a disparu, on peut rester dans l'état q de $x = 2,5$ à $x = 3,5$ par exemple, ce qui est impossible avec $I(q)$.

Pour remédier à ce problème, nous allons associer à chaque invariant I une famille d'invariants $I_1, .., I_k$ telle que $I = I_1 \vee \ .. \ \vee I_k$ et telle que les I_j soient convexes pour le temps. Par exemple, dans le cas précédent, $I(q) = (x < 3) \vee (x > 3)$. Et ce sont ces nouveaux invariants que nous allons fermer à droite.

D'autre part, dans la construction précédente, un état ayant initialement pour invariant $x > 3$, a comme nouvel invariant $(x > 3) \vee y = 0$. Supposons que l'on arrive dans cet état suite à une transition d'états avec $x = 3$, puisqu'on vient de tirer une transition, $y = 0$ et donc le nouvel invariant est vérifié. Et ensuite, si le temps passe $x > 3$, on peut rester dans cet état ce qui est impossible avec l'automate initial. C'est pour éviter ce problème que nous avons exigé que les invariants soient clos par le passé . On rajoute donc un état qui a pour unique invariant $y = 0$. On obtient alors l'automate de la figure 3.4

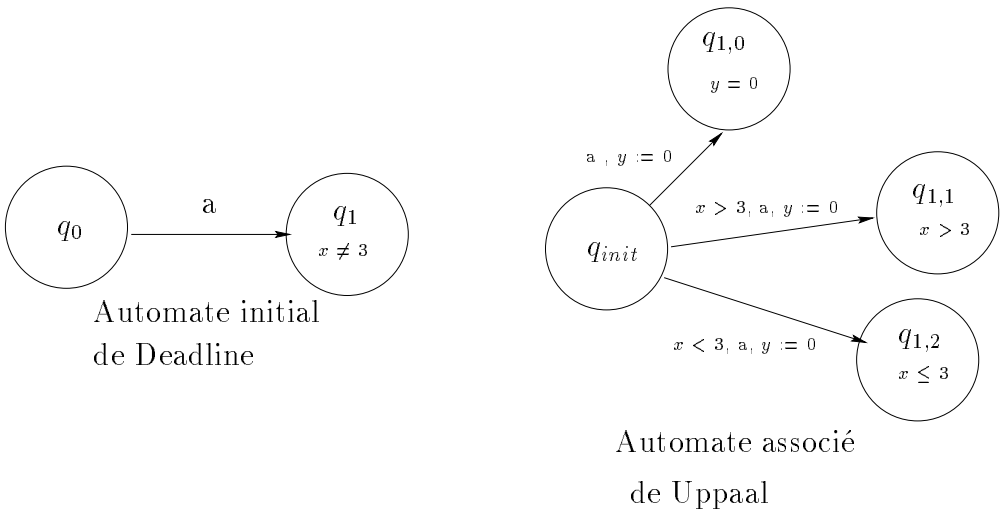


FIG. 3.4 – *construction d'un automate de Uppaal à partir d'un automate ayant comme invariant* $x \neq 3$

Dans le cas général, pour définir ces nouveaux invariants, nous allons définir des régions et ensuite des ensembles de régions appelés domaines.

3.5.2 Les régions d'horloges

Les régions d'horloges que nous présentons sont peu différentes de celles présentées dans [AD90, ACD93]. Pour simplifier la présentation, nous supposerons, dans toute cette partie, que toutes les constantes apparaissant dans les invariants sont entières (cela permet d'utiliser la partie entière et la partie fractionnaire des valeurs des horloges).

Les invariants sont donc des formules définies par la grammaire:

$$A ::= x \sim t |\ x - y \sim t |\ A \wedge A |\ \neg A |\ True$$

où $\sim \in \{<, \leq, =, \geq, >\}$, $x \in X, y \in X, t \in \mathbb{N}$.

Pour chaque $t \in \mathbb{R}$, nous noterons $fract(t) \in [0, 1[$ la partie fractionnaire de t et $E(t)$ sa partie entière : $t = fract(t) + E(t)$.

Définition 3.5.1 *Soit I un invariant. Soit m la plus grande constante apparaissant dans I. La* **relation d'équivalence** $\equiv_I$ *est définie sur l'ensemble des valuations d'horloges de la façon suivante, $v \equiv_I v'$ si toutes les conditions suivantes sont vérifiées :*

1. $\forall x \in X$, *on a soit* $E(v(x)) = E(v'(x))$, *soit* $(E(v(x)) > m) \wedge (E(v'(x)) > m)$,
2. $\forall (x, y) \in X^2$, *on a soit* $E(v(x) - v(y)) = E(v'(x) - v'(y))$, *soit* $(E(v(x)) - E(v(y)) > m) \wedge (E(v'(x)) - E(v'(y)) > m)$, *soit* $(E(v(x)) - E(v(y)) < -m) \wedge (E(v'(x)) - E(v'(y)) < -m)$,
3. $\forall x \in X$, *tel que* $v(x) < m$, $fract(v(x)) = 0 \Leftrightarrow fract(v'(x)) = 0$,
4. $\forall (x, y) \in X^2$, *tel que* $-m < v(x) - v(y) < m$, $fract(v(x) - v(y)) = 0 \Leftrightarrow fract(v'(x) - v'(y)) = 0$.

Définition 3.5.2 *Une* **région d'horloge** *pour I est une classe d'équivalence de valuations d'horloges pour $\equiv_I$.*

Conséquence 3.5.3 *Si v et v' sont deux valuations telles que $v \equiv_I v'$, v satisfait I si et seulement si v' satisfait I.*

Remarque 3.5.4 On note $[v]_I$ la région de la valuation v.

Exemple 3.5.5 La figure 3.5 présente les différentes régions existantes pour l'invariant $((x \neq 2) \vee ((x - y) > 1)) \wedge (y < 3)$.

Remarque 3.5.6 En fait, $\equiv_I$ dépend de m plus que de I.

Lemme 3.5.7 *On peut associer à chaque région r une contrainte d'horloges ϕ_r telle que $r = Z_{\phi_r}$ et qui est telle que :*

1. *pour chaque x de X, il y a exactement une contrainte de l'ensemble $\{x = t$ tel que $t = 0, 1, ..m\} \cup \{t - 1 < x < t$ tel que $t = 1, ..m\} \cup \{m < x\}$*
2. *pour chaque x et y de X, il y a exactement une contrainte de l'ensemble $\{x - y = t$ tel que $t = -m, -m + 1, ..m\} \cup \{t - 1 < x - y < t$ tel que $t = -m + 1, ..m\} \cup \{m < x - y\} \cup \{-m > x - y\}$*

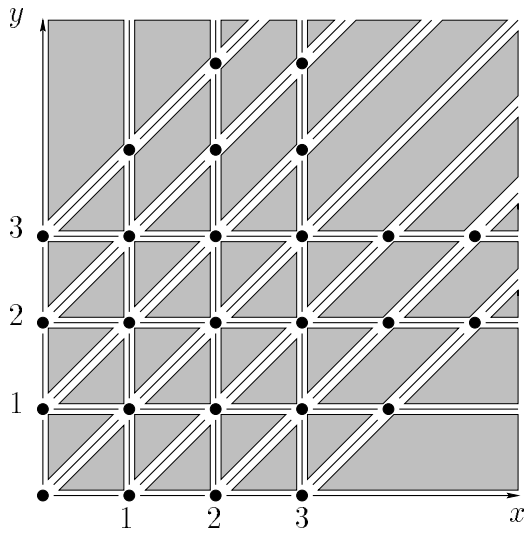


FIG. 3.5 – *les régions de* $((x \neq 2) \vee (x - y) > 1) \wedge (y < 3)$

Démonstration omise.

Conséquence 3.5.8 *Soit I un invariant, il y a un nombre fini de régions pour I.*

Conséquence 3.5.9 *Soit I un invariant, les régions de I sont convexes et donc convexes pour le temps.*

3.5.3 Les domaines d'horloges

Une façon de construire l'automate de Uppaal associé serait la suivante : étant donné un état q de l'automate de Deadline, on associe à $I(q)$ un certain nombre de régions en considérant la relation d'équivalence $\equiv_{I(q)}$. A chaque région r, on associe un état $q_{r,q}$ et un invariant ϕ_r. L'automate de Uppaal aurait pour état, les états $q_{r,q}$ et pour invariant, les invariants ϕ_r.

Dans ce cas, considérons l'invariant $((x \neq 2) \vee ((x - y) > 1)) \wedge (y < 3)$, $r_0 = \{(0,0)\}$ est une région associée à cet invariant (cf figure 3.5). Supposons que l'on est dans l'état q_{r_0}, donc la valuation est $(0,0)$. On ne peut pas y rester car si un délai s'écoule on viole l'invariant donc on est obligé de faire une transition d'états. Or on peut rester dans un état d'invariant $((x \neq 2) \vee ((x - y) > 1)) \wedge (y < 3)$, $r_0 = \{(0,0)\}$ avec une valuation $(0,0)$.

Pour contourner ce problème, on regroupe les régions. Nous définissons des domaines d'horloges qui n'existent pas, à notre connaissance, dans la littérature.

Une relation d'ordre sur les régions

Informellement, dans le cas de la dimension 2, les régions qui sont supérieures à une région r, sont les régions que l'on rencontre quand on est dans r et que l'on suit une diagonale $x - y = cste$, c'est à dire que le temps s'écoule.

Définition 3.5.10 *Une région r_1 est inférieure à une région r_2, noté $r_1 << r_2$, si :*

1. $\forall v \in r_1,\ \exists \eta \geq 0$ *tel que* $v + \eta \in r_2$ *et*

2. $\forall v \in r_2,\ \exists \eta \geq 0$ *tel que* $v - \eta \in r_1$.

Proposition 3.5.11
$<<$ *est une relation d'ordre.*

Démonstration : L'antisymétrie provient du fait que les régions sont convexes:
si $r_1 << r_2 << r_1$, soit $v \in r_1$, $\exists \eta_2 \geq 0$, tel que $v + \eta_2 \in r_2$. Donc, $\exists \eta_1 \geq 0$ tel que $v + \eta_2 + \eta_1 \in r_1$. Donc, d'après la conséquence 3.5.9, $v + \eta_2 \in r_1$. D'où $r_1 \cap r_2 \neq \emptyset$ et $r_1 = r_2$. □

Proposition 3.5.12

- *si* $r << r_1$ *et* $r << r_2$, *alors* $r_1 << r_2$ *ou* $r_2 << r_1$.
- *si* $r_1 << r$ *et* $r_2 << r$, *alors* $r_1 << r_2$ *ou* $r_2 << r_1$.

Proposition 3.5.13 *Soient* v *une valuation et* η *un réel positif,* $[v] << [v + \eta]$.

Démonstration : cette démonstration très calculatoire est en annexe. □

Les domaines d'horloges

On définit maintenant une relation d'équivalence sur les régions.

Définition 3.5.14 *La relation d'équivalence* $\equiv$ *est définie par* $r \equiv r'$ *si l'une des conditions suivantes est remplie:*

- $r << r'$ *et* $\forall r_0$ *telle que* $r << r_0 << r'$, $\forall v \in r_0$, $v \vDash I$,
- $r << r'$ *et* $\forall r_0$ *telle que* $r << r_0 << r'$, $\forall v \in r_0$, $v \nvDash I$,
- $r' << r$ *et* $\forall r_0$ *telle que* $r' << r_0 << r$, $\forall v \in r_0$, $v \vDash I$,
- $r' << r$ *et* $\forall r_0$ *telle que* $r' << r_0 << r$, $\forall v \in r_0$, $v \nvDash I$.

Remarque 3.5.15 C'est bien la définition d'une équivalence (c'est une conséquence de la propriété 3.5.12).

Définition 3.5.16 *Un* **domaine d'horloge** *pour* I *est une classe d'équivalence de régions d'horloges pour* $\equiv$.

Exemple 3.5.17 La figure 3.6 présente les différents domaines existants pour l'invariant $((x \neq 2) \vee ((x - y) > 1)) \wedge (y < 3)$.

Lemme 3.5.18 *Si* v *et* v' *sont deux valuations appartenant au même domaine,* v *satisfait* I *si et seulement si* v' *satisfait* I.

Proposition 3.5.19 *On peut associer à chaque domaine* D *une formule* I_D *construite sur la même grammaire que les invariants telle que* v *appartient à* D *si et seulement si* v *satisfait* I_D. $I_D = \bigvee_{r \in D} \phi_r$.

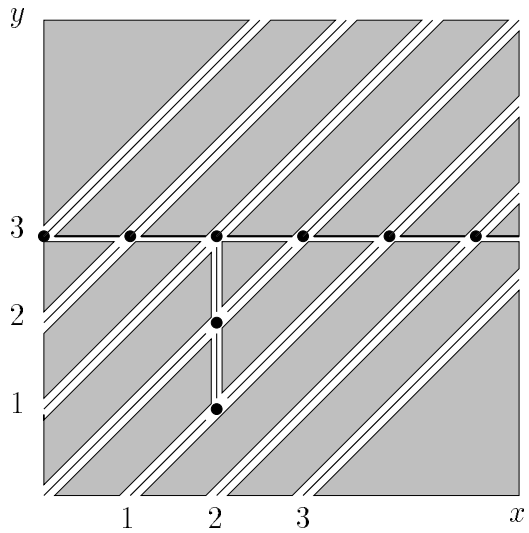


FIG. 3.6 – *Les domaines de* $((x \neq 2) \vee (x - y) > 1) \wedge (y < 3)$

Soit I un invariant, soit $\mathcal{D}$ l'ensemble fini de ses domaines. Soit $\mathcal{D}^+$ l'ensemble des domaines D qui sont tels que si v appartient à D, v satisfait I. L'ensemble $\mathcal{D}^+$ peut s'écrire $\{D_1, D_2, ..., D_n\}$. On a $I \equiv I_{D_1} \vee .. \vee I_{D_n}$.
On note $D_0 = \bigcup_{D \in \mathcal{D} \setminus \mathcal{D}^+} D$, D_0 est l'ensemble des valuations qui ne vérifient pas I.

Exemple 3.5.20
Sur la figure 3.7, les éléments de $\mathcal{D}^+$ apparaissent en gris foncé et D_0 en gris clair.

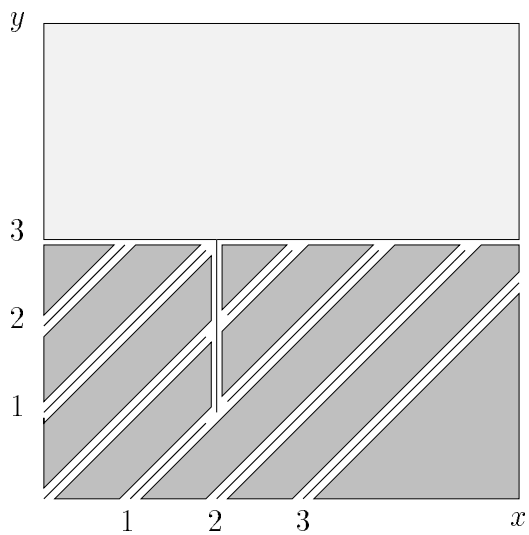


FIG. 3.7 – *Les domaines de* $((x \neq 2) \vee (x - y) > 1) \wedge (y < 3)$

3.6 Simulation des automates de Deadline par des automates de Uppaal, dans le cas général

Etant donné un automate de Deadline, nous souhaitons exhiber un automate de Uppaal tel que leurs systèmes de transitions soient bisimilaires. Pour cela, nous allons associer à chaque couple (état, invariant) un ensemble de couples (état,invariant).

3.6.1 Automate de Uppaal associé

Soit $A = (\Sigma, Q, q_0, X, T_A, I)$ un automate de Deadline, tel que $Q = \{q_0, .., q_N\}$.
A tout état q_j, on associe un ensemble $\mathcal{D}_j^+ = \{D_{j,1}, .., D_{j,n_j}\}$, un ensemble de valuation $D_{j,0}$, des états $q_{j,0}, .., q_{j,n_j}$ et des invariants $I_{j,0}, .., I_{j,n_j}$ tels que $I_{j,1} = I_{D_{j,1}}$ et $I_{j,0} = \neg I(q_j)$. On a $I(q_j) \equiv I_{j,1} \vee .. \vee I_{j,n_j}$ et $True \equiv I_{j,0} \vee .. \vee I_{j,n_j}$.

Définition 3.6.1 *$\{\phi\}^\lambda$ est la formule obtenue, à partir de ϕ, en remplaçant par zéro toutes les horloges contenues dans λ i.e. $\{\phi\}^\lambda = \phi[0 \leftarrow \lambda]$.*

On considère l'automate de Uppaal suivant $B = (\Sigma, Q_B, q_{B,0}, X_B, T_B, I_B)$ défini par :

- $Q_B = \{q_{j,k}\}_{0 \leq j \leq N,\ 0 \leq k \leq n_j}$
- $q_{B,0} = q_{0,k}$ tel que si $v_0 \not\models I_A(q_0),\ k = 0$ sinon $v_0 \models I_{0,k}$,
- $X_B = X \cup \{y\}$,
- T_B est défini de la façon suivante :

  $(q_{j_0,k_0}, \delta \wedge \{I_{j_1,k_1}\}^\lambda, a, \lambda \cup \{y\}, q_{j_1,k_1}) \in T_B$ si et seulement si $(q_{j_0}, \delta, a, \lambda, q_{j_1}) \in T_A$

- I_B est défini de la façon suivante :
  - $I_B(q_{j,k}) = F_d(I_{j,k})$ si $k \neq 0$,
  - $I_B(q_{j,0}) = (y = 0)$.

3.6.2 Les automates ont des systèmes de transitions bisimilaires

Définition 3.6.2 *Soit $\mathcal{R}$ la relation incluse dans $(Q \times \mathbb{T}^X) \times (Q_B \times \mathbb{T}^{X \cup \{y\}})$ définie par $((q_j, v), (q', v')) \in \mathcal{R}$ si et seulement si :*

- $v'_{|X} = v$,
- $v' - v'(y)$ *a un sens, i.e.* $\forall x \in X,\ v'(x) - v'(y) \geq 0$.
- *soit k tel que $v' - v'(y) \models I_{j,k}$, alors $q' = q_{j,k}$ et $v' \models I_B(q_{j,k})$.*

Remarque 3.6.3
$v' - v'(y)$ est la valuation correspondant à l'instant où l'automate est arrivé dans un nouvel état suite à une transition d'états.
Quelque soient, $v' \in \mathbb{T}^{X \cup \{y\}}$ et $j \in [\![0, N]\!]$, k existe toujours car $True \equiv I_{j,0} \vee .. \vee I_{j,n_j}$.

Lemme 3.6.4 *Soit $((q_j, v), (q_{j,k}, v')) \in \mathcal{R}$, si $k = 0$ alors $v \not\models I_A(q_j)$.*

Démonstration : Si $k = 0$ alors, comme $v' \vDash I_B(q_{j,0})$ et $I_B(q_{j,0}) = (y = 0)$, $v'(y) = 0$. Donc $v' - v'(y) = v'$. Or $v' - v'(y) \vDash I_{j,0}$ donc $v' \vDash I_{j,0}$ et $v \vDash I_{j,0}$. D'où $v \nvDash I_A(q_j)$. □

Proposition 3.6.5 *La relation $\mathcal{R}$ est incluse dans $S_A \times S_B$.*

Démonstration : Soit $((q_j, v), (q', v')) \in \mathcal{R}$, $(q_j, v) \in S_A$ car $S_A = Q \times \mathbb{T}^X$ et $(q', v') \in S_B$ car $v' \vDash I_B(q_{j,k})$ et $q' = q_{j,k}$. □

Proposition 3.6.6 *La relation $\mathcal{R}$ a la propriété de transfert.*

Démonstration : Cette démonstration est très longue et ne présente pas de difficulté majeure, elle est proposée en annexe. □

Théorème 3.6.7
Le système de transitions de A, $\mathcal{S}_A = (S_A, s_{A,0}, \rightarrow_A)$ est bisimilaire au système de transitions de B, $\mathcal{S}_B = (S_B, s_{B,0}, \rightarrow_B)$.

Démonstration : La relation $\mathcal{R}$ est incluse dans $S_A \times S_B$ et a la propriété de transfert. De plus, $(s_{A,0}, s_{B,0})$ appartient à $\mathcal{R}$. Donc, $\mathcal{R}$ est une bisimulation pour $\mathcal{S}_A$ et $\mathcal{S}_B$. □

3.7 L'horloge supplémentaire est-elle nécessaire?

Dans les deux constructions que nous proposons pour trouver un automate de Uppaal qui soit bisimilaire à un automate de Deadline, nous avons besoin d'ajouter une horloge supplémentaire. On peut se demander si elle est vraiment nécessaire.

Conjecture 3.7.1 *Pour tout entier n, il existe un automate de Deadline ayant n horloges tel que tout automate de Uppaal qui lui est bisimilaire a au moins $n + 1$ horloges.*

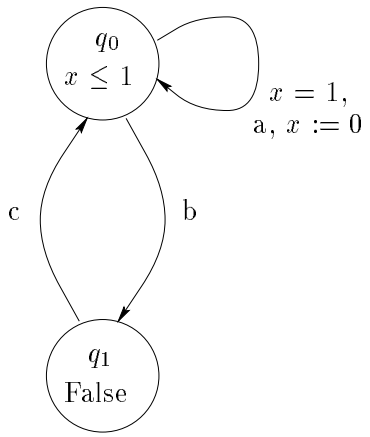


FIG. 3.8 – *Automate de Deadline Contre-exemple (C)*

Démonstration : **Voici une preuve de cette conjecture dans le cas particulier où** $n = 1$.

Considérons un automate de Uppaal A qui reconnait le langage de l'automate de Deadline Contre-exemple (figure 3.8) et qui a une unique horloge. Nous allons montrer que A et C n'ont pas le même langage.
$C = (\{a, b, c\},\ \{q_0, q_1\},\ q_0, \{x\},\ T_C,\ I_C)$ et $A = (\{a, b, c\},\ Q_A,\ q_{0,A},\ \{y\},\ T_A,\ I_A)$.

Pour tout $\eta \leq 1$,

$$(q_0, 0) \xrightarrow{\eta}_C (q_0, \eta) \xrightarrow{b}_C (q_1, \eta) \xrightarrow{c}_C (q_0, \eta) \xrightarrow{1-\eta}_C (q_0, 1) \xrightarrow{a}_C (q_0, 0)$$

est une exécution de l'automate C, donc "$\eta\ b\ c\ (1-\eta)\ a$" est une trace de A. Donc, pour tout $\eta \leq 1$, il existe une exécution de A de trace "$\eta\ b\ c\ (1-\eta)\ a$".

Nous allons déterminer un η tel que, pour toute exécution de A de trace "$\eta\ b\ c\ (1-\eta)\ a$", la transition de l'automate A d'étiquette b utilisée remette y à zéro.

Considérons l'ensemble des constantes comprises **strictement** entre 0 et 1 apparaissant dans les gardes des transitions de A. Si cet ensemble est vide, $M_1 = 0,5$, sinon M_1 est le maximum de cet ensemble fini. Dans tous les cas, $0 < M_1 < 1$.

Considérons l'ensemble des constantes comprises **strictement** entre 0 et 1 apparaissant dans les invariants des états de A. Si cet ensemble est vide, $M_2 = 0,5$, sinon M_2 est le maximum de cet ensemble. $0 < M_2 < 1$.

Soit $M = sup(M_1, M_2)$. Soit un réel η fixé tel que $0 < M < \eta < 1$. Considérons une exécécution de "$\eta\ b\ c\ (1-\eta)\ a$" pour l'automate A :

$$(q_{0,A}, 0) \xrightarrow{\eta}_A (q_{0,A}, \eta) \xrightarrow{b}_A (q_\alpha, v_\alpha) \xrightarrow{c}_A (q_\beta, v_\beta) \xrightarrow{1-\eta}_A (q_\gamma, v_\gamma) \xrightarrow{a}_A (q_\epsilon, v_\epsilon)$$

$(q_{0,A}, \eta) \xrightarrow{b}_A (q_\alpha, v_\alpha)$ donc il existe $(q_{0,A}, \delta, b, \lambda, q_\alpha) \in T_A$ tel que $\eta \vDash \delta$, $v_\alpha = \eta[0 \leftarrow \lambda]$ et $v_\alpha \vDash I_A(q_\alpha)$.

Supposons que $\lambda = \emptyset$, montrons que l'on aboutit à une contradiction. On a alors $v_\alpha = \eta$ et $\eta \vDash I_A(q_\alpha)$. Soit $\eta' \in]\eta, 1[$, soit $t \in [0, 1-\eta']$, $\eta + t \in [\eta, 1[$ et $M_2 < \eta < 1$ donc $\eta + t \vDash I_A(q_\alpha)$. Par conséquent, $(q_\alpha, \eta) \xrightarrow{\eta'-\eta}_A (q_\alpha, \eta')$ est une transition de A et "$\eta\ b\ (\eta' - \eta)$" est une trace de A avec $\eta' - \eta > 0$. Or "$\eta\ b\ (\eta' - \eta)$" n'est pas une trace de C, donc $\lambda \neq \emptyset$ et $v_\alpha = 0$.

Nous allons montrer que si A a pour trace "$\eta\ b\ c\ (1-\eta)\ a$" alors il existe d compris strictement entre 0 et $1-\eta$ tel que A a également pour trace "$\eta\ b\ c\ d\ a$". On aura alors montré que A et C ne reconnaissent pas le même langage.

Considérons l'ensemble des constantes comprises **strictement** entre 0 et 1 apparaissant dans les gardes des transitions de A. Si cet ensemble est vide, $m_1 = 0,5$, sinon m_1 est le minimum de cet ensemble fini. Dans tous les cas, $0 < m_1 < 1$.

Considérons l'ensemble des constantes comprises **strictement** entre 0 et 1 apparaissant dans les invariants des états de A. Si cet ensemble est vide, $m_2 = 0,5$, sinon m_2 est le maximum de cet ensemble. $0 < m_2 < 1$.

Soit $m = inf(m_1, m_2)$. Soit d un réel tel que $\eta < d < 1$ et $0 < 1 - d < m$.

$$(q_{0,A}, 0) \xrightarrow{\eta}_A (q_{0,A}, \eta) \xrightarrow{b}_A (q_\alpha, 0) \xrightarrow{c}_A (q_\beta, 0) \xrightarrow{1-\eta}_A (q_\gamma, 1-\eta) \xrightarrow{a}_A (q_\epsilon, v_\epsilon)$$

est une exécution de A. Donc $(q_\beta, 0) \xrightarrow{1-\eta}_A (q_\gamma, 1-\eta)$ est une transitions de temps de A, d'où $\forall t \in [0, 1-\eta],\ t \vDash I_A(q_{beta})$. En particulier, $\forall t \in [0, 1-d],\ t \vDash I_A(q_{beta})$ donc $(q_\beta, 0) \xrightarrow{1-\eta}_A (q_\gamma, 1-d)$ est une transition de temps de A.

De plus, $(q_\gamma, 1-\eta) \xrightarrow{a}_A (q_\epsilon, v_\epsilon)$ est une transition d'états de A donc $(q_\gamma, \delta, a, \lambda, q_\epsilon) \in T_A$ tel que $1-\eta \vDash \delta$, $v_\epsilon = \eta[0 \leftarrow \lambda]$ et $v_\epsilon \vDash I_A(q_\epsilon)$. $0 < 1-d < m_1$, donc $1-d \vDash \delta$. Soit $v'_\epsilon = d[0 \leftarrow \lambda]$. $0 < 1-d < m_2$ donc que λ soit vide ou égal à $\{x\}$, $v'_\epsilon \vDash I_A(q_\epsilon)$. Donc $(q_\gamma, 1-d) \xrightarrow{a}_A (q_\epsilon, v'_\epsilon)$.

On en conclu que

$$(q_{0,A}, 0) \xrightarrow{\eta}_A (q_{0,A}, \eta) \xrightarrow{b}_A (q_\alpha, 0) \xrightarrow{c}_A (q_\beta, 0) \xrightarrow{1-d}_A (q_\gamma, 1-d) \xrightarrow{a}_A (q_\epsilon, v'_\epsilon)$$

est une exécution de A. Ceci est impossible car $1-d \neq 1-\eta$.

□

3.8 Simulation des automates de Uppaal par des automates de Deadline

3.8.1 Situation

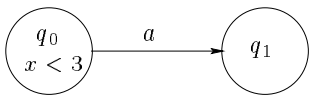


FIG. 3.9 – *contre-exemple*

Considérons l'automate de Uppaal A de la figure 3.9. Pour tout délai $d < 3$, $< q_0, 0 > \xrightarrow{d}_A < q_0, d >$ en est un comportement. Donc "d"est une trace de A. Par contre, "3" n'est pas une trace de A car 3 viole l'invariant de q_0.

Supposons maintenant qu'il existe un automate de Deadline qui reconnaisse le même langage que A. Alors, quelque soit $d < 3$, "d"est une trace de B. Par conséquent, soit q'_0 l'état initial de B, $< q'_0, 0 > \xrightarrow{d}_B < q'_0, d >$ est un comportement de B.

Soit $0 \leq t < 3$, soit $d \in]t, 3]$, $< q'_0, 0 > \xrightarrow{d}_B < q'_0, d >$ est une transition de temps de B donc $\forall t' \in [0, d[,\ t' \vDash I_B(q'_0)$, en particulier, $t \vDash I_B(q'_0)$. Donc $\forall t' \in [0, 3[,\ t \vDash I_B(q'_0)$ d'où $< q'_0, 0 > \xrightarrow{3}_B < q'_0, 3 >$ est une transition de temps de B. Donc "3"appartient au langage de B, il ne reconnaît donc pas le même langage que A.

Au vu de cet exemple, étant donné un automate de Uppaal, il ne sera pas toujours possible de trouver un automate de Deadline qui reconnaisse le même langage et donc, à fortiori, qui ait un système de transitions bisimilaire à celui de l'automate initial. Toutefois, il est possible de lui associer un automate de Deadline qui ait un langage "proche". Nous définissons donc une nouvelle notion : la bisimulation mixte entre deux systèmes de transitions.

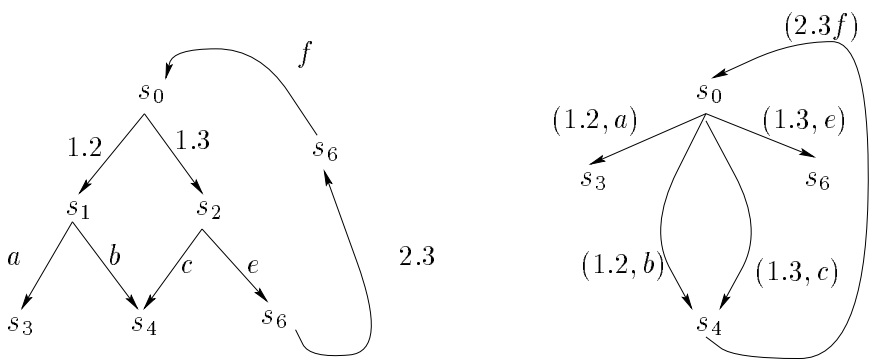
3.8.2 La bisimulation mixte

Définition 3.8.1 *Soit $\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ un système de transitions avec $\mathcal{A}ct = \Sigma \cup \mathbb{T}$. Soient*

$d \in \mathbb{T}$, $a \in \Sigma$ et $(s, s') \in S^2$, $s \stackrel{(d,a)}{\Longrightarrow} s'$ si, par définition, il existe $s'' \in S$ tel que $s \stackrel{d}{\longrightarrow} s''$ et $s'' \stackrel{a}{\longrightarrow} s'$.

Définition 3.8.2 *Soit $\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ un système de transitions. On lui associe le système de transition $\mathbb{S} \stackrel{def}{=} (S, s_0, \mathbb{T} \times \Sigma, \Rightarrow)$. On appelle $\mathbb{S}$ le* **système de transitions mixte associé à $\mathcal{S}$.**

Remarque 3.8.3 Le système de transition mixte est celui qui est choisi pour définir la sémantique des automates temporisés dans le modèle d'Alur et Dill, présentée dans [AD90, ACD93].



Système de transitions $\mathcal{S}$ Système de transitions $\mathbb{S}$

FIG. 3.10 – *Exemple de système de transitions mixte*

Définition 3.8.4 *Soient $\mathcal{S}$ et $\mathcal{S}'$ deux systèmes de transitions. $\mathcal{S}$ et $\mathcal{S}'$ sont* **bisimilaires mixtes** *si, par définition, $\mathbb{S}$ et $\mathbb{S}'$ sont bisimilaires.*

Remarque 3.8.5 Si $\mathcal{S}$ et $\mathcal{S}'$ sont bisimilaires mixtes, alors tout mot de $(\mathbb{T} \cdot \Sigma)^*$ qui est une trace de l'un des systèmes de transitions est une trace de l'autre.

Rappel 3.8.6
Soit $\mathbb{A}$ un alphabet, l'ensemble des mots finis formés sur $\mathbb{A}$ est noté $\mathbb{A}^*$, l'ensemble des mots finis non vide formés sur $\mathbb{A}$ est noté $\mathbb{A}^+$ et l'ensemble des mots infinis formés sur $\mathbb{A}$ est noté $\mathbb{A}^\omega$.

3.8.3 Automate de Deadline associé

Soit $A = (\Sigma, Q, q_0, X, T_A, I)$ un automate de Uppaal.
Soit $\mathcal{A}ct$ l'union de l'ensemble des actions Σ et l'ensemble des délais $\mathbb{T}$.

On considère l'automate de Deadline suivant $B = (\Sigma, Q_B, q_{B,0}, X_B, T_B, I_B)$ définie par :

- $Q_B = Q$,

- $q_{B,0} = q_0$,
- $X_B = X$,
- T_B est défini de la façon suivante :

$$(q, \delta \wedge I(q) \wedge \{I(q')\}^{\lambda}, a, \lambda, q') \in T_B \text{ si et seulement si } (q, \delta, a, \lambda, q') \in T_A,$$

- $I_B = I$.

Théorème 3.8.7
Le système de transition de A, $\mathcal{S}_A = (S_A, s_{A,0}, \rightarrow_A)$ est bisimilaire mixte au système de transition de B, $\mathcal{S}_B = (S_B, s_{B,0}, \rightarrow_B)$.

Démonstration : Soit $\mathcal{R}$ la relation incluse dans $S_A \times S_B$ définie par : $((q, v), (q', v')) \in \mathcal{R}$ si et seulement si :

1. $(q', v') = (q, v)$,
2. $(q, v) \in S_A$.

Montrons que $\mathcal{R}$ est bien une bisimulation mixte:
Tout d'abord, $(s_{A,0}, s_{B,0}) \in \mathcal{R}$.
Ensuite, soit $((q, v), (q, v)) \in \mathcal{R}$.

- Soient $(q_1, v_1) \in S_A$, $d \in \mathbb{T}$ et $a \in \Sigma$ tels que $(q, v) \stackrel{(d,a)}{\Longrightarrow}_A (q_1, v_1)$,

  alors il existe $(q_2, v_2) \in S_A$ tel que $(q, v) \stackrel{d}{\longrightarrow}_A (q_2, v_2)$ et $(q_2, v_2) \stackrel{a}{\longrightarrow}_A (q_1, v_1)$.

  $(q, v) \stackrel{d}{\longrightarrow}_A (q_2, v_2)$ donc $q = q_2$, $v_2 = v + d$ et $\forall t \in [0, d]$, $v + t \vDash I(q)$. Donc $(q, v) \stackrel{d}{\longrightarrow}_B (q_2, v_2)$.

  $(q_2, v_2) \stackrel{a}{\longrightarrow}_A (q_1, v_1)$ donc $\exists (q_2, \delta, a, \lambda, q_1) \in T_A$ tel que $v_2 \vDash \delta$, $v_1 = v_2[0 \leftarrow \lambda]$ et $v_1 \vDash I(q_1)$.

  Par conséquent, $(q_2, \delta \wedge I(q_2) \wedge \{I(q_1)\}^{\lambda}, a, \lambda, q_1) \in T_B$. $v_2 \vDash \delta$, $v_2 \vDash I(q_2)$ (en raison de la transition de temps précédente) et $v_2 \vDash \{I(q_1)\}^{\lambda}$ car $v_1 = v_2[0 \leftarrow \lambda]$ et $v_1 \vDash I(q_1)$. Donc $(q_2, v_2) \stackrel{a}{\longrightarrow}_B (q_1, v_1)$. D'où $(q, v) \stackrel{(d,a)}{\Longrightarrow}_B (q_1, v_1)$. De plus, $v_1 \vDash I(q_1)$ donc $((q_1, v_1), (q_1, v_1)) \in \mathcal{R}$.

- Soient $(q_1, v_1) \in S_B$, $d \in \mathbb{T}$ et $a \in \Sigma$ tels que $(q, v) \stackrel{(d,a)}{\Longrightarrow}_B (q_1, v_1)$,

  alors il existe $(q_2, v_2) \in S_B$ tel que $(q, v) \stackrel{d}{\longrightarrow}_B (q_2, v_2)$ et $(q_2, v_2) \stackrel{a}{\longrightarrow}_B (q_1, v_1)$.

  $(q, v) \stackrel{d}{\longrightarrow}_B (q_2, v_2)$ donc $q = q_2$, $v_2 = v + d$ et $\forall t \in [0, d[$, $v + t \vDash I(q)$. $(q_2, v_2) \stackrel{a}{\longrightarrow}_B (q_1, v_1)$ donc $\exists (q_2, \delta \wedge I(q_2) \wedge \{I(q_1)\}^{\lambda}, a, \lambda, q_1) \in T_B$, $v_2 \vDash \delta \wedge I(q_2) \wedge \{I(q_1)\}^{\lambda}$ et $v_1 = v_2[0 \leftarrow \lambda]$.

  $v_2 \vDash \delta \wedge I(q_2) \wedge \{I(q_1)\}^{\lambda}$ donc $v_2 \vDash I(q_2)$ donc $\forall t \in [0, d]$, $v + t \vDash I(q)$. Par conséquent, $(q, v) \stackrel{d}{\longrightarrow}_A (q_2, v_2)$.

  $(q_2, \delta \wedge I(q_2) \wedge \{I(q_1)\}^{\lambda}, a, \lambda, q_1) \in T_B$ donc $(q_2, \delta, a, \lambda, q_1) \in T_A$. De plus, $v_2 \vDash \delta$, $v_1 = v_2[0 \leftarrow \lambda]$ et $v_1 \vDash I(q_1)$ car $v_2 \vDash \{I(q_1)\}^{\lambda}$. Donc $(q_2, v_2) \stackrel{a}{\longrightarrow}_A (q_1, v_1)$.

  D'où $(q, v) \stackrel{(d,a)}{\Longrightarrow}_B (q_1, v_1)$. De plus, $v_1 \vDash I(q_1)$ donc $((q_1, v_1), (q_1, v_1)) \in \mathcal{R}$.

□

Conséquence 3.8.8 *Tout mot de* $(\mathbb{T} \cdot \Sigma)^*$ *qui appartient au langage de A, appartient au langage de B et vice-versa.*

Donc si on choisit la sémantique des automates temporisés présentée dans [AD90, ACD93], A et B reconnaissent le même langage.

En fait, on peut raffiner un peu ce résultat en observant quelques propriétés des systèmes de transitions. Ceci est l'objet de la sous-partie qui suit.

3.8.4 Propriétés des systèmes de transitions relatives au temps

L'additivité du temps

Définition 3.8.9 $\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ *a la propriété d'***additivité du temps** *si :*

$$\forall (s, s_1) \in S^2,\ \forall (d, d') \in \mathbb{T}^2,\ (\exists s_2 \in S,\ s \xrightarrow{d}_S s_2\ \wedge\ s_2 \xrightarrow{d'}_S s_1) \Leftrightarrow s \xrightarrow{d+d'}_S s_1.$$

Proposition 3.8.10 *Soit* $A = (\Sigma, Q, q_0, X, T, I)$ *un automate avec invariant, quelque soit l'interprétation choisie (Uppaal ou Deadline), le système de transition associé à A a la propriété d'additivité du temps.*

Démonstration :

- Dans le cas de l'interprétation de Deadline,
  - Soient $(d, d') \in \mathbb{T}^2$ tels que $(q, v) \xrightarrow{d+d'} (q_1, v_1)$ est une transition de temps de $\mathcal{S}_A$. Alors, $q_1 = q$, $v_1 = v + d + d'$ et $\forall t,\ 0 \leq t < d + d',\ v + t \vDash I(q)$. Soient $q_2 = q$ et $v_2 = v + d$. $\forall t,\ 0 \leq t < d,\ v + t \vDash I(q)$ donc $(q, v) \xrightarrow{d} (q_2, v_2)$ et $\forall t,\ 0 \leq t < d',\ v + d + t \vDash I(q)$ donc $(q_2, v_2) \xrightarrow{d'} (q_1, v_1)$.
  - Soient $(d, d') \in \mathbb{T}^2$ tels que $(q, v) \xrightarrow{d} (q_2, v_2)$ et $(q_2, v_2) \xrightarrow{d'} (q_1, v_1)$. Alors, $q_2 = q$, $q_1 = q_2$, $v_2 = v + d$, $v_1 = v_2 + d'$, $\forall t,\ 0 \leq t < d,\ v + t \vDash I(q)$ et $\forall t,\ 0 \leq t < d',\ v + d + t \vDash I(q)$. Donc $\forall t,\ 0 \leq t < d + d',\ v + t \vDash I(q)$ et $(q, v) \xrightarrow{d+d'} (q_1, v_1)$ est une transition de temps de $\mathcal{S}_A$.
- dans le cas de l'interprétation de Uppaal, la démonstration est analogue en remplaçant les $<$ par des $\leq$.

□

Lemme 3.8.11 *Soient* $\mathcal{S}$ *et* $\mathcal{S}'$ *deux systèmes de transitions ayant la propriété d'additivité du temps. Si* $\mathcal{S}$ *et* $\mathcal{S}'$ *sont bisimilaires mixtes, alors tout mot de* $((\mathbb{T})^+ \cdot \Sigma)^*$ *qui est une trace de l'un des systèmes de transitions est une trace de l'autre.*

La propriété de délai nul

Définition 3.8.12 *$\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ a la* **propriété de délai nul** *si :*

$$\forall s \in S,\ s \xrightarrow{0}_S s.$$

Proposition 3.8.13 *Soit $A = (\Sigma, Q, q_0, X, T, I)$ un automate avec invariant, quelque soit l'interprétation choisie (Uppaal ou Deadline), le système de transition associé à A a la propriété de délai nul.*

Démonstration :

- Dans le cas de l'interprétation de Deadline, cela provient du fait que, dans le cas où $d = 0, \forall t,\ 0 \leq t < d,\ v + t \vDash I(q)$ est toujours vérifiée.
- dans le cas de l'interprétation de Uppaal, cela provient du fait que $(q, v) \in S_A$ si et seulement si $v \vDash I(q)$.

□

La propriété de déterminisme du temps

Définition 3.8.14 *$\mathcal{S} = (S, s_0, \mathcal{A}ct, \rightarrow)$ a la* **propriété de déterminisme du temps** *si :*

$$\forall (s, s_1, s_2) \in S^3,\ \forall d \in \mathbb{T},\ (s \xrightarrow{d}_S s_1 \wedge s \xrightarrow{d}_S s_2) \Rightarrow s_1 = s_2.$$

Proposition 3.8.15 *Soit $A = (\Sigma, Q, q_0, X, T, I)$ un automate avec invariant, quelque soit l'interprétation choisie (Uppaal ou Deadline), le système de transition associé à A a la propriété de déterminisme du temps.*

Démonstration : Soient $d \in \mathbb{T}$, $(q, v) \in S_A$ et $(q', v') \in S_A$ tels que $(q, v) \xrightarrow{d} (q', v')$ est une transition de temps de $\mathcal{S}_A$. Alors, $q' = q$, $v' = v + d$, donc on n'a aucun choix possible pour (q', v'). □

Proposition 3.8.16 *Soit $\mathcal{S}$ un système de transitions ayant la propriété de délai nul et la propriété de déterminisme du temps. Soient $a \in \Sigma$ et $(s, s') \in S^2$, $s \overset{(0,a)}{\Longrightarrow} s$ si et seulement si $s \xrightarrow{a} s'$.*

Démonstration : Soient $a \in \Sigma$ et $(s, s') \in S^2$ tels que $s \overset{(0,a)}{\Longrightarrow} s$ alors il existe $s'' \in S$ tel que $s \xrightarrow{0} s''$ et $s'' \xrightarrow{a} s'$. Or $s \xrightarrow{0} s$ (propriété de délai nul) donc $s'' = s$ (propriété de déterminisme du temps).
Soient $a \in \Sigma$ et $(s, s') \in S^2$ tels que $s \xrightarrow{a} s'$, $s \xrightarrow{0} s$ (propriété de délai nul) donc $s \overset{(0,a)}{\Longrightarrow} s$. □

Lemme 3.8.17 *Soient $\mathcal{S}$ et $\mathcal{S}'$ deux systèmes de transitions ayant la propriété d'additivité du temps, la propriété de délai nul et la propriété de déterminisme du temps. Si $\mathcal{S}$ et $\mathcal{S}'$ sont bisimilaires mixtes, alors tout mot de $((\mathbb{T})^* \cdot \Sigma)^*$ qui est une trace de l'un des systèmes de transitions est une trace de l'autre.*

Proposition 3.8.18 *Soit $\mathcal{L}_A$ le langage reconnu par A inclus dans $((\mathbb{T})^* \cdot \Sigma)^*$ et $\mathcal{L}_B$ celui de B. $\mathcal{L}_A = \mathcal{L}_B$.*

Démonstration : C'est une conséquence du fait que $\mathcal{S}_A$ et $\mathcal{S}_B$ ont la propriété d'additivité du temps, la propriété de délai nul et la propriété de déterminisme du temps (propriétés 3.8.10, 3.8.13 et 3.8.15), du fait que $\mathcal{S}_A$ et $\mathcal{S}_B$ sont bisimilaires mixtes (théorème 3.8.7) et du lemme précédent. □

3.9 Les automates avec conditions limites

Une autre extension des automates temporisés sont les automates avec conditions limites (deadlines). C'est un concept dual des invariants quand on choisit l'interprétation dite des automates de Deadline [SY96, BS97] d'où le nom "automate de Deadline".

3.9.1 Définition d'un automate avec conditions limites

Les conditions limites sont des conditions sur les horloges qui s'interprètent de la façon suivante : l'automate peut "rester" dans un état q tant qu'aucune des conditions limites des transitions de l'automate qui ont pour état initial q n'est satisfaite.

Définition 3.9.1 *Un* **automate temporisé avec conditions limites** *A est un sextuplet $(\Sigma, Q, q_0, X, T, D)$, où :*

- *(Σ, Q, q_0, X, T) est un automate temporisé,*
- *D est une fonction de T dans $\mathcal{C}(X)$.*

A chaque état q de Q, on associe un invariant de la manière suivante :

$$D(q) \stackrel{\text{def}}{=} \bigvee_{t=(q,a,q') \in T} D(t) \quad \text{et} \quad I(q) \stackrel{\text{def}}{=} \neg D(q)$$

Cela définit une fonction $I : Q \longrightarrow \mathcal{C}(X)$. On peut alors considérer l'automate avec invariant A_I défini par:

$$A_I = (\Sigma, Q, q_0, X, T, I).$$

Définition 3.9.2 *Par définition, le* **système de transition associé** *de A est celui associé à A_I, interprété comme un automate de Deadline. De même, une exécution pour l'automate A est une exécution de l'automate A_I.*

On note désormais A_I l'automate avec invariants correspondant à un automate avec conditions limites A.

En fin de compte, ce qui a de l'importance pour le comportement de l'automate avec conditions limites est, pour chaque état q, l'ensemble de conditions limites associées aux transitions d'origine q. Il semble donc plus naturel d'associer directement des contraintes aux états, donc d'utiliser des invariants.

La dernière définition associe à un automate avec conditions limites un automate de Deadline qui a le même système de transitions. On peut se demander si, étant donné un automate de Deadline, on peut lui associer un automate avec conditions limites. C'est l'objet de la sous-partie qui suit.

3.9.2 Automate avec conditions limites associé à un automate de Deadline

Soit $A = (\Sigma, Q, q_0, X, T, I)$ un automate de Deadline. A chaque transition $t = (q, \delta, a, \lambda, q') \in T$, on associe une condition limite de la manière suivante :

$$D(t) \stackrel{\text{def}}{=} \neg I(q).$$

Cela définit une fonction $D : T \longrightarrow \mathcal{C}(X)$ et permet de considérer l'automate avec conditions limites A_D défini par:

$$A_D = (\Sigma, Q, q_0, X, T, D).$$

Définition 3.9.3 *A_D est l'automate avec conditions limites associé à A.*

Exemple 3.9.4 (fin de la minuterie)
Voici la minuterie avec conditions limites associée à celle de la figure 3.2. Les conditions limites sont encadrées.

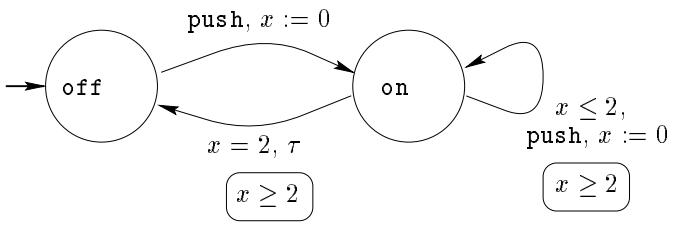


FIG. 3.11 – *une minuterie avec conditions limites*

Proposition 3.9.5 *Soit A un automate de Deadline, l'automate de Deadline associé à A_D est égal à A i.e. $(A_D)_I = A$.*

Conséquence 3.9.6 *Soit A un automate de Deadline, A et A_D ont le même système de transition.*

Démonstration : Soit $(A_D)_I$ l'automate de Deadline associé à A_D. A et $(A_D)_I$ ont le même alphabet, le même ensemble d'états, le même état initial, le même ensemble d'horloges et le même ensemble de transitions. Reste donc à voir que I la fonction d'invariants de A est égale à I' la fonction d'invariants de $(A_D)_I$.

Soit $q \in Q$,

$$\begin{aligned} I'(q) &= \neg D(q) \\ &= \neg\Big(\bigvee_{t=(q,a,q')\in T} D(t)\Big) \\ &= \neg\Big(\bigvee_{t=(q,a,q')\in T} \neg I(q)\Big) \\ &= I(q). \end{aligned}$$

□

3.9.3 Conclusion

L'ensemble des systèmes de transitions associés aux automates avec conditions limites est le même que celui des systèmes de transitions associés aux automates de Deadline. D'autre part, **étant donné un automate d'un type (de Deadline ou avec conditions limites), on peut lui associer un automate de l'autre type ayant le même système de transitions. Par conséquent, on peut associer à tout automate de Deadline, un automate avec conditions limites qui lui est bisimilaire et on peut associer à tout automate avec conditions limites, un automate de Deadline qui lui est bisimilaire mixte.**

Chapitre 4

Produits de synchronisation

Pour modéliser un système, on commence souvent par modéliser des sous-systèmes que l'on compose ensuite en les synchronisant suivant certaines actions. Dans ce chapitre, nous présentons le produit de synchronisation des systèmes de transitions. Nous définissons ensuite celui des automates temporisés avec invariants ou conditions limites afin que le système de transitions associés aux produit de deux automates et le produit des systèmes de transitions de chacun de ces automates soient bisimilaires.

4.1 Produit de synchronisation de deux systèmes de transitions

Dans toute cette partie, $\mathcal{A}ct$ désigne un ensemble d'étiquettes de transitions et $\mathcal{S}yncr \subset \mathcal{A}ct$ un ensemble d'étiquettes de synchronisation.

4.1.1 Définition de la composition de deux systèmes de transitions temporisés

Définition 4.1.1 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S, \mathcal{A}ct)$ et $\mathcal{S}' = (S', s'_0, \rightarrow_{S'}, \mathcal{A}ct)$ deux systèmes de transitions, on note :*

$$\mathcal{U} = \mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}'$$

le système de transitions obtenu en synchronisant les systèmes de transitions $\mathcal{S}$ et $\mathcal{S}'$ sur l'ensemble $\mathcal{S}yncr$. $\mathcal{U} = (U, u_0, \rightarrow_U, \mathcal{A}ct)$ est défini de la manière suivante :

- *$U = S \times S'$, on note $s \otimes_{\mathcal{S}yncr} s'$ pour $(s, s') \in U$,*
- *$u_0 = (s_0, s'_0)$ $(= s_0 \otimes_{\mathcal{S}yncr} s'_0)$.*
- *$\rightarrow_U$ est définie par les règles suivantes :*

$$\frac{s \xrightarrow{\ell}_S u}{s \otimes_{\mathcal{S}yncr} s' \xrightarrow{\ell}_U u \otimes_{\mathcal{S}yncr} s'} \ \ell \in \mathcal{A}ct \backslash \mathcal{S}yncr,\ s' \in S' \ (1)$$

$$\frac{s' \xrightarrow{\ell}_{S'} u'}{s \otimes_{\mathcal{S}yncr} s' \xrightarrow{\ell}_U s \otimes_{\mathcal{S}yncr} u'} \ \ell \in \mathcal{A}ct \backslash \mathcal{S}yncr,\ s \in S \ (2)$$

$$\frac{s \xrightarrow{\ell}_S u,\ s' \xrightarrow{\ell}_{S'} u'}{s \otimes_{\mathcal{S}yncr} s' \xrightarrow{\ell}_U u \otimes_{\mathcal{S}yncr} u'}\ \ell \in \mathcal{S}yncr \quad (3)$$

Nous allons maintenant voir un certain nombre de propriétés de ce produit que nous utiliserons quand nous nous intéresserons aux automates temporisés avec invariants.

4.1.2 Propriétés du produit de synchronisation de deux systèmes de transitions

Toutes les démonstrations de cette sous-partie sont calculatoires et ne posent aucune difficulté. Elles sont donc dans l'annexe.

L'associativité de l'opération de composition

Proposition 4.1.2 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S, \mathcal{A}ct)$, $\mathcal{S}' = (S', s'_0, \rightarrow_{S'}, \mathcal{A}ct)$ et $\mathcal{S}'' = (S'', s''_0, \rightarrow_{S''}, \mathcal{A}ct)$ trois systèmes de transitions, $(\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}') \otimes_{\mathcal{S}yncr} \mathcal{S}''$ est isomorphe (donc bisimilaire) à $\mathcal{S} \otimes_{\mathcal{S}yncr} (\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}'')$.*

La commutativité de l'opération de composition

Proposition 4.1.3 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S, \mathcal{A}ct)$ et $\mathcal{S}' = (S', s'_0, \rightarrow_{S'}, \mathcal{A}ct)$ deux systèmes de transitions, $\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}'$ est isomorphe (donc bisimilaire) à $\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}$.*

La bisimulation est une congruence par rapport à l'opération de composition

Proposition 4.1.4 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S)$ et $\mathcal{S}' = (S', s'_0, \rightarrow_{S'})$ deux systèmes de transitions bisimilaires, alors $\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}''$ et $\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}''$ sont bisimilaires.*

4.2 Produit de synchronisation pour les automates avec invariants

Dans cette partie, nous allons définir le produit de synchronisation de deux automates avec invariants. Notre définition est la même que celle de [SY96] et est très proche de celle de [DY95]. Ce qui est nouveau, c'est que nous montrons que, pour chaque interprétation, le système de transition associé au produit de deux automates temporisés avec invariants est bisimilaire au produit de synchronisation des systèmes de transitions de chaque automate.

4.2.1 Composition d'automates avec invariants

Définition 4.2.1 *Soient $\mathcal{A} = (\Sigma, Q_A, q_{0,A}, X_A, T_A, I_A)$ et $\mathcal{B} = (\Sigma, Q_B, q_{0,B}, X_B, T_B, I_B)$ deux automates avec invariants. Soit J un ensemble de synchronisation. On note :*

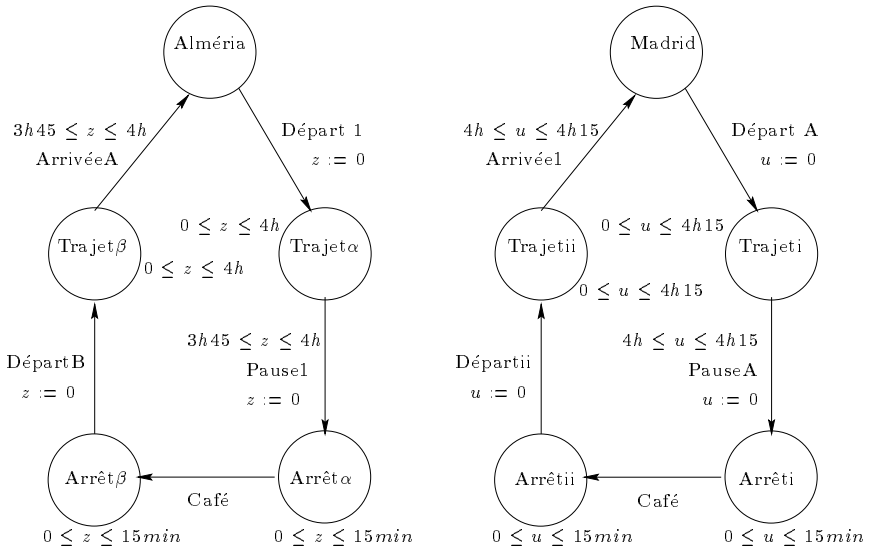
$$\mathcal{C} = \mathcal{A} \otimes_J \mathcal{B}$$

*l'***automate obtenu en synchronisant les automates avec invariants** *$\mathcal{A}$ et $\mathcal{B}$ sur l'ensemble J. $\mathcal{C} = (\Sigma, Q_C, q_{0,C}, X_C, T_C, I_C)$ est défini de la manière suivante :*

- $Q_C \stackrel{\text{def}}{=} Q_A \times Q_B$,

- $q_{0,C} \stackrel{\text{def}}{=} (q_{0,A}, q_{0,B})$,
- $X_C \stackrel{\text{def}}{=} X_A \sqcup X_B$,
- *on définit* I_C *de la manière suivante : Soit* $(q_A, q_B) \in Q_C$, $I_C((q_A, Q_B)) \stackrel{\text{def}}{=} I_A(q_A) \wedge I_B(q_B)$,
- *on définit* T_C *de la manière suivante :* $(q_A, q_B) \xrightarrow{\delta,a,\lambda} (q'_A, q'_B)$ *si*
  1. $a \in \Sigma \backslash J$,
     (a) $q_B = q'_B$, $\delta \in \mathcal{C}(X_A)$, $\lambda \in \mathcal{P}(X_A)$ *et* $q_A \xrightarrow{\delta,a,\lambda} q'_A \in T_A$ *ou*
     (b) $q_A = q'_A$, $\delta \in \mathcal{C}(X_B)$, $\lambda \in \mathcal{P}(X_B)$ *et* $q_B \xrightarrow{\delta,a,\lambda} q'_B \in T_B$,
  2. $a \in J$, $\delta = \delta_A \wedge \delta_B$ *avec* $\delta_A \in \mathcal{C}(X_A)$ *et* $\delta_B \in \mathcal{C}(X_B)$, $\lambda = \lambda_A \sqcup \lambda_B$ *avec* $\lambda_A \in \mathcal{P}(X_A)$ *et* $\lambda_B \in \mathcal{P}(X_B)$, $q_A \xrightarrow{\delta_A,a,\lambda_A} q'_A \in T_A$ *et* $q_B \xrightarrow{\delta_B,a,\lambda_B} q'_B \in T_B$.

Exemple 4.2.2 (chauffeurs et cars)
Sur la liaison Alméria-Madrid, un car circule dans chaque sens chaque jour. Le voyage est long, environ huit heures. Chaque chauffeur ne fait donc qu'un seul trajet par jour. Afin de rentrer tous les jours chez eux, les deux chauffeurs ont convenu d'un rendez-vous dans un café situé approximativement à mi-distance, pour échanger leurs cars et donc repartir d'où ils viennent.



Chauffeur 1 Chauffeur 2

FIG. 4.1 – *Les chauffeurs*

Les automates de la figure 4.2 représentent le trajet de chaque car. La partie Alméria-le café prend entre $3h45$ et $4h$, la partie le café-Madrid prend entre $4h$ et $4h15$ et les chauffeurs ont droit à au plus $15min$ de pause.

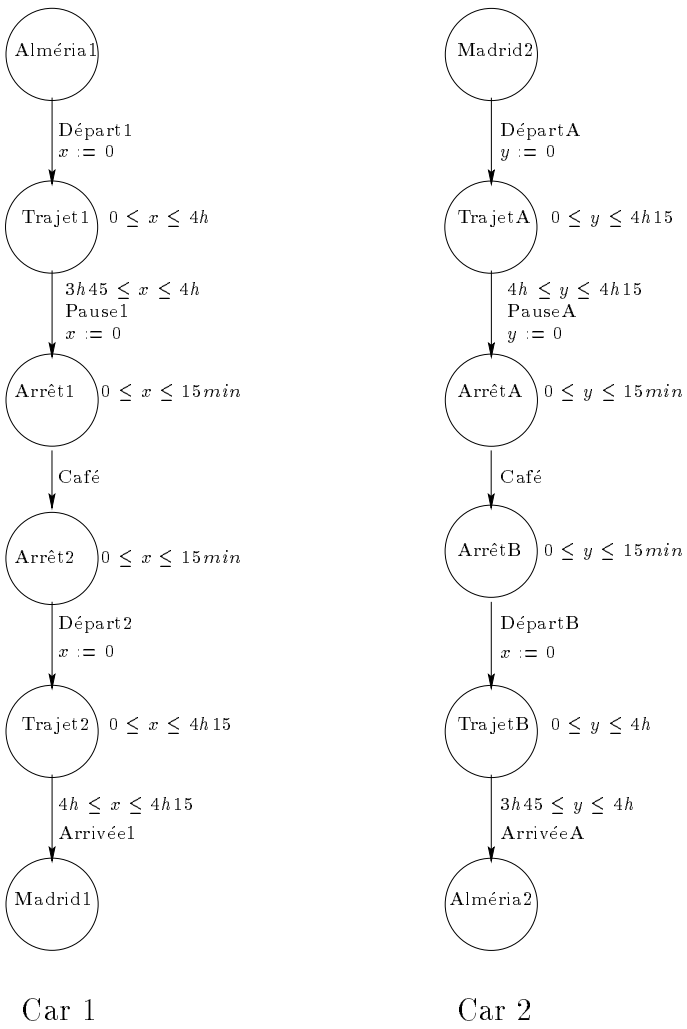


FIG. 4.2 – *Les cars*

Les automates de la figure 4.1 représentent le trajet de chaque conducteur. L'automate obtenu en synchronisant les quatre précédents sur les actions : Départ1, Départ2, DépartA, DépartB, Pause1, PauseA, Café, Arrivée1 et Arrivée2 a été coupé en deux pour pouvoir apparaître de façon lisible sur des figures. Il est donc obtenu en superposant les figures 4.3 et 4.4 sur l'action Café et les états Arrêt1 et Arrêt2. Dans la figure 4.3, l'horloge X représente à la fois l'horloge x et l'horloge z. Elles ne sont distinguées qu'après les arrêts car alors le chauffeur1 et le car1 agissent séparément. X représente alors x et X', z. De même pour Y, y, u.

Dans les deux sous-parties suivantes, nous montrons que quelle que soit l'interprétation choisie (Uppaal ou Deadline), étant donnés deux automates temporisés avec invariants A et B, $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$ et $\mathcal{S}_{A \otimes_J B}$ sont isomorphes où $\mathcal{A}ct = \Sigma \cup \mathbb{T}$ et $\mathcal{S}yncr = J \cup \mathbb{T}$.

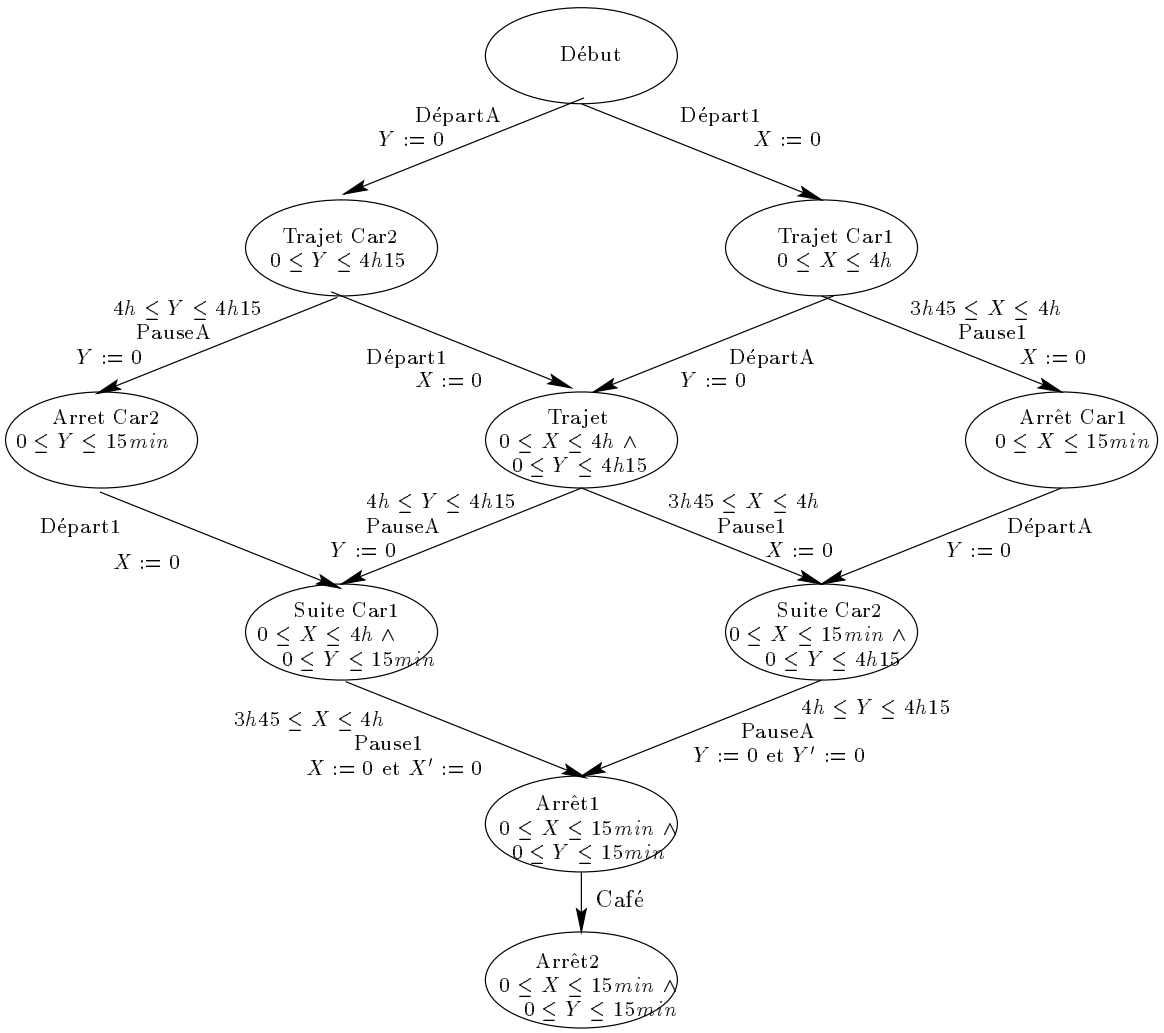


FIG. 4.3 – *L'automate de synchronisation (partie haute)*

4.2.2 Théorème de correction pour les automates de Uppaal

Théorème 4.2.3 *Soient* $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, I_A)$ *et* $B = (\Sigma, Q_B, q_{0,B}, X_B, T_B, I_B)$ *deux automates de Uppaal. Soient* $\mathcal{A}ct = \Sigma \cup \mathbb{T}$ *et* $\mathcal{S}yncr = J \cup \mathbb{T}$. $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$ *et* $\mathcal{S}_{A \otimes_J B}$ *sont isomorphes donc bisimilaires.*

On a le diagramme suivant :

$$\begin{array}{ccccc} (A,B) & & \longrightarrow & & (\mathcal{S}_A, \mathcal{S}_B) \\ \downarrow & & & & \downarrow \\ A \otimes_J B & \longrightarrow & \mathcal{S}_{A \otimes_J B} & = & \mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B \end{array}$$

Démonstration : Soit $h : \mathcal{S}_{A \otimes_J B} \to \mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$ tel que $((q_A, q_B), [v_A v_B]) \to (q_A, v_A) \otimes_{\mathcal{S}yncr} (q_B, v_B)$. h est un isomorphisme entre $\mathcal{S}_{A \otimes_J B}$ et $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$. La démonstration est longue mais ne présente aucune difficulté, elle est proposée en annexe. □

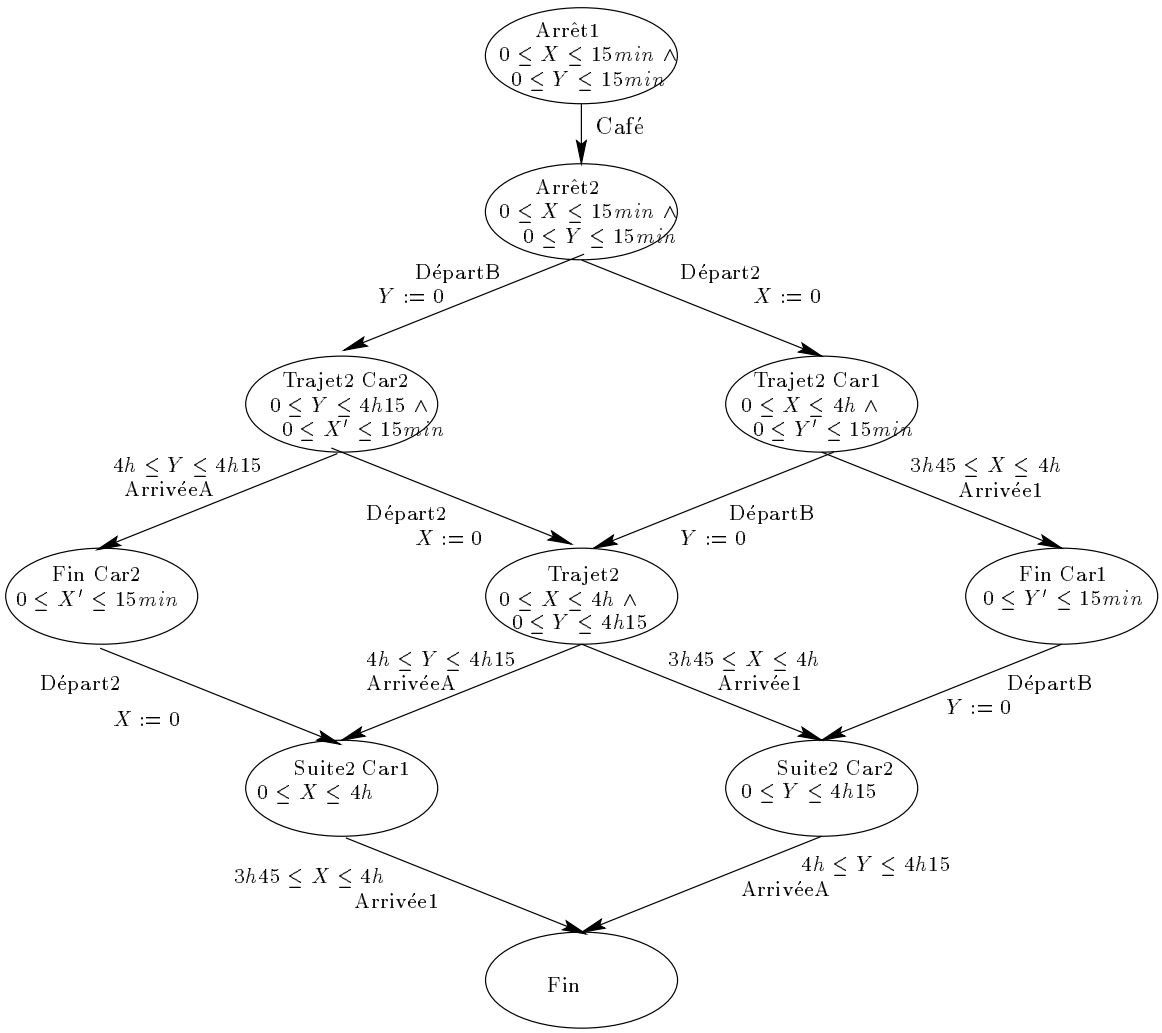


FIG. 4.4 – *L'automate de synchronisation (partie basse)*

4.2.3 Théorème de correction pour les automates de Deadline

Théorème 4.2.4 *Soient* $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, I_A)$ *et* $B = (\Sigma, Q_B, q_{0,B}, X_B, T_B, I_B)$ *deux automates de Deadline. Soient* $\mathcal{A}ct = \Sigma \cup \mathbb{T}$ *et* $\mathcal{S}yncr = J \cup \mathbb{T}$. $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$ *et* $\mathcal{S}_{A \otimes_J B}$ *sont isomorphes donc bisimilaires.*

On a le diagramme suivant:

$$\begin{array}{ccccc} (A,B) & & \longrightarrow & & (\mathcal{S}_A, \mathcal{S}_B) \\ \downarrow & & & & \downarrow \\ A \otimes_J B & \longrightarrow & \mathcal{S}_{A \otimes_J B} & = & \mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B \end{array}$$

Démonstration : La démonstration est quasi-identique à celle du théorème de correction pour les automates de Uppaal. Les différences sont les suivantes :

- on n'a plus besoin de vérifier que $((q_A, q_B), [v_A v_B]) \in S$, $(q_A, v_A) \otimes_{\mathcal{S}yncr} (q_B, v_B) \in U$, $(q_A, v_A) \in S_A$ ou $(q_B, v_B) \in S_B$. En effet, les invariants n'entraînent aucune restriction sur l'ensemble des configurations donc $S = (Q_A \times Q_B) \times \mathbb{T}^{X_A \cup X_B}$, $U = (Q_A \times \mathbb{T}^{X_A}) \times (Q_b \times \mathbb{T}^{X_B})$, $S_A = Q_A \times \mathbb{T}^{X_A}$ et $S_B = Q_B \times \mathbb{T}^{X_B}$.
- lorsque $\ell \in \mathbb{T}$, il faut remplacer, $\forall t \leq \ell$ par $\forall t < \ell$.
- lorsque $\ell \in \Sigma$, il n'y a pas besoin de vérifier que $v' \vDash I(q')$ pour avoir $(q, v) \xrightarrow{\ell} (q', v')$.

□

4.2.4 Conséquences des théorèmes de correction

Conséquence 4.2.5 *Le produit d'automates (de Uppaal ou de Deadline) est associatif et commutatif. Soient A, B et C trois automates temporisés avec invariants, alors quelle que soit l'interprétation choisie (Uppaal ou Deadline),*

$$A \otimes_J B \text{ et } B \otimes_J A \text{ sont bisimilaires.}$$

$$(A \otimes_J B) \otimes_J C \text{ et } A \otimes_J (B \otimes_J C) \text{ sont bisimilaires.}$$

Démonstration : Etant donné que l'opération de composition sur les systèmes de transitions est commutative (cf proposition 4.1.3), on a les équations suivantes :

$$\begin{aligned} \mathcal{S}_{A \otimes_J B} &\leftrightarrow \mathcal{S}_A \otimes_{\mathcal{A}ct} \mathcal{S}_B \\ &\leftrightarrow \mathcal{S}_B \otimes_{\mathcal{A}ct} \mathcal{S}_A \\ &\leftrightarrow \mathcal{S}_{B \otimes_J A}. \end{aligned}$$

Etant donné que l'opération de composition sur les systèmes de transitions est associative (cf proposition 4.1.2), on a les équations suivantes :

$$\begin{aligned} \mathcal{S}_{(A \otimes_J B) \otimes_J C} &\leftrightarrow (\mathcal{S}_A \otimes_{\mathcal{A}ct} \mathcal{S}_B) \otimes_{\mathcal{A}ct} \mathcal{S}_C \\ &\leftrightarrow \mathcal{S}_A \otimes_{\mathcal{A}ct} (\mathcal{S}_B \otimes_{\mathcal{A}ct} \mathcal{S}_C) \\ &\leftrightarrow \mathcal{S}_{A \otimes_J (B \otimes_J C)}. \end{aligned}$$

□

Conséquence 4.2.6 *Les traductions d'automates de X en Y, qui sont correctes par rapport à la bisimulation, sont compatibles avec les produits (de X, de Y). En particulier, les traductions d'automates de Deadline en automates de Uppaal, sont compatibles avec les produits (de Deadline, de Uppaal). Soient A_1 et A_2 deux automates de Deadline, soient B_1 et B_2 les automates de Deadline associés, alors*

$$A_1 \otimes_J A_2 \text{ et } B_1 \otimes_J B_2 \text{ sont bisimilaires.}$$

Démonstration : Etant donné que A_1 et B_1 sont bisimilaires, que A_2 et B_2 sont bisimilaires, on a

$$\mathcal{S}_{A_1} \leftrightarrow \mathcal{S}_{B_1} \text{ et } \mathcal{S}_{A_2} \leftrightarrow \mathcal{S}_{B_2}$$

Comme la bisimulation est une congruence par rapport à l'opération de composition (proposition 4.1.4),

$$\mathcal{S}_{A_1} \otimes_{\mathcal{A}ct} \mathcal{S}_{A_2} \leftrightarrow \mathcal{S}_{B_1} \otimes_{\mathcal{A}ct} \mathcal{S}_{B_2}$$
$$\text{et } \mathcal{S}_{A_1 \otimes_J A_2} \leftrightarrow \mathcal{S}_{B_1 \otimes_J B_2}.$$

□

4.3 Produit de synchronisation pour les automates avec conditions limites

4.3.1 Composition d'automates avec conditions limites

Définition 4.3.1 *Soient* $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, D_A)$ *et* $B = (\Sigma, Q_B, q_{0,B}, X_B, T_B, D_B)$ *deux automates avec conditions limites. Soit* $J \subset \Sigma$ *un ensemble d'actions de synchronisation. On note :*

$$\mathcal{C} = \mathcal{A} \otimes_J \mathcal{B}$$

l'automate obtenu en synchronisant les automates $\mathcal{A}$ *et* $\mathcal{B}$ *sur l'ensemble* J.

$C = (\Sigma, Q_C, q_{0,C}, X_C, T_C, D_C)$ *est défini de la manière suivante :*

- $Q_C \stackrel{\text{def}}{=} Q_A \times Q_B$
- $q_{0,C} \stackrel{\text{def}}{=} (q_{0,A}, q_{0,B})$
- $X_C \stackrel{\text{def}}{=} X_A \sqcup X_B$
- *on définit* T_C *et* D_C *par les règles suivantes :*

$$\frac{(t_A = q_A \stackrel{(\delta,a,\lambda)}{\longrightarrow} q'_A) \in T_A}{(t_C = (q_A, q_B) \stackrel{(\delta,a,\lambda)}{\longrightarrow} (q'_A, q_B)) \in T_C,\ D_C(t_C) = D_A(t_A) \vee D_B(q_B)} \; q_B \in Q_B,\ a \in \Sigma \backslash J \quad (1)$$

$$\frac{(t_B = q_B \stackrel{(\delta,a,\lambda)}{\longrightarrow} q'_B) \in T_B}{(t_C = (q_A, q_B) \stackrel{(\delta,a,\lambda)}{\longrightarrow} (q'_A, q_B)) \in T_C,\ D_C(t_C) = D_A(q_A) \vee D_B(t_B)} \; q_A \in Q_A,\ a \in \Sigma \backslash J \quad (2)$$

$$\frac{(t_A = q_A \stackrel{(\delta_A,a,\lambda_A)}{\longrightarrow} q'_A) \in T_A,\ (t_B = q_B \stackrel{(\delta_B,a,\lambda_B)}{\longrightarrow} q'_B) \in T_B}{(t_C = (q_A, q_B) \stackrel{(\delta_A \wedge \delta_B, a, \lambda_A \cup \lambda_B)}{\longrightarrow} (q'_A, q'_B)) \in T_C,\ D_C(t) = D_A(t_A) \vee D_B(t_B)} \; a \in J \quad (3)$$

4.3.2 Théorème de correction pour les automates avec conditions limites

Proposition 4.3.2 *Soient $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, D_A)$ et $B = (\Sigma, Q_B, q_{0,B}, X_B, T_B, D_B)$ deux automates avec conditions limites. Soit $J \subset \Sigma$ un ensemble d'actions de synchronisation.*

$C = A \otimes_J B$ et $D = A_I \otimes_J B_I$ se comportent de la même manière i.e. $C_I = D$.

Démonstration : Soient $C = (\Sigma, Q_C, q_{0,C}, X_C, T_C, D_C)$ et $D = (\Sigma, Q_D, q_{0,D}, X_D, T_D, I_D)$.

$Q_C = Q_D$, $q_{0,C} = q_{0,D}$, $X_C = T_D$ et $T_C = T_D$. Montrons que $I_{C_I} = I_D$.

Soient $(q_A, q_B) \in Q_C$.

$$
\begin{aligned}
D_C((q_A,q_B)) &= \bigvee_{t=((q_A,q_B),a,q')\in T_C} D_C(t) \\
&= \left(\bigvee_{\substack{t=((q_A,qB),a,(q'A,q_B))\in T_C \\ a\in\Sigma\setminus J}} D_C(t) \right) \vee \left(\bigvee_{\substack{t=((q_A,qB),a,(qA,q'_B))\in T_C \\ a\in\Sigma\setminus J}} D_C(t) \right) \vee \left(\bigvee_{\substack{t=((q_A,qB),a,(qA',q'_B))\in T_C \\ a\in J}} D_C(t) \right) \\
&= \left(\bigvee_{\substack{t_A=(q_A,a,q'A)\in T_A \\ a\in\Sigma\setminus J}} D_A(t_A) \vee D_B(q_B) \right) \vee \left(\bigvee_{\substack{t_B=((qB,a,q'_B)\in T_B \\ a\in\Sigma\setminus J}} D_A(q_A) \vee D_B(t_B) \right) \\
&\vee \left(\bigvee_{\substack{t_A=(q_A,a,q'A)\in T_A \\ (qB,a,q'_B)\in T_B,\ a\in J}} D_A(t_A) \vee D_B(t_B) \right) \\
&= D_A(q_A) \vee D_B(q_B) \vee \left(\bigvee_{t_A=(q_A,a,q'A)\in T_A} D_A(t_A) \right) \vee \left(\bigvee_{t_B=((qB,a,q'_B)\in T_B} D_B(t_B) \right) \\
&= D_A(q_A) \vee D_B(q_B).
\end{aligned}
$$

Donc,

$$
\begin{aligned}
I_C((q_A,q_B)) &= \neg(D_C((q_A,q_B))) \\
&= \neg(D_A(q_A) \vee D_B(q_B)) \\
&= I_A(q_A) \wedge I_B(q_B) \\
&= I_D((q_A,q_B)).
\end{aligned}
$$

□

D'où le théorème de correction :

Théorème 4.3.3 *Soient A et B deux automates avec conditions limites. $\mathcal{S}_A \otimes_J \mathcal{S}_B$ et $\mathcal{S}_{A \otimes_{Syncr} B}$ sont isomorphes donc bisimilaires.*

On a le diagramme suivant :

$$
\begin{array}{ccccc}
(A,B) & & \longrightarrow & & (\mathcal{S}_A, \mathcal{S}_B) \\
\downarrow & & & & \downarrow \\
A \otimes_J B & \longrightarrow & \mathcal{S}_{A\otimes_J B} & = & \mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B
\end{array}
$$

Chapitre 5

Conclusion

L'objet de ce travail était d'étudier la sémantique des automates temporisés avec invariants ou conditions limites, afin de leur associer un produit de composition ayant de bonnes propriétés.

Nous avons commencé par une étude bibliographique lors de laquelle nous avons constaté l'existence de peu de travaux théoriques concernant les différentes constructions de vivacité pour les automates temporisés (deadlines, invariants ou actions urgentes). Nous avons également remarqué que plusieurs définitions (plus ou moins précises) des constructions de vivacité sont proposées, qu'aucune définition standard n'émerge et qu'aucun article ne tente de les comparer.

Nous avons donc dû isoler et formaliser les différentes notions de construction de vivacité. Cela nous a amené à distinguer deux sémantiques d'automates temporisés avec invariants : les automates de Uppaal et les automates de Deadline, et une sémantique d'automates avec conditions limites. Nous nous sommes demandé comment les comparer.

Nous nous sommes alors posés différentes questions : peut-on trouver, étant donné un automate d'un type X, un automate de type Y qui ait un "comportement semblable" ? Quelle est pour un type d'automate donné une bonne définition du produit de synchronisation ? Enfin, est-ce que la traduction d'un automate de type X en un automate de type Y est compatible avec les produits de synchronisation de X et de Y ?

Afin de donner des réponses à ces questions, nous avons dû définir une notion de "comportement semblable", nous avons choisi la bisimulation classique, temporisée de deux systèmes de transitions et en avons défini une variante, la *bisimulation mixte*. Par ailleurs, nous avons été amenés à définir plusieurs nouveaux outils tels que des propriétés sur les ensembles de valuations d'horloges : *clôture par le passé*, *convexité pour le temps* et *fermeture à droite* et de nouveaux ensembles de valuations d'horloges : les *domaines d'horloges*.

Nos résultats : Nous avons défini un produit de synchronisation sur les systèmes de transitions qui sert de critère de correction pour nos définitions de produits d'automates. En particulier, nous avons vérifié que, étant donné deux automates temporisés du même type (de Deadline, de Uppaal ou avec conditions limites) A_1 et A_2, le produit de synchronisation des systèmes de transitions de A_1 et A_2 est égal au système de transitions associé à l'automate égal au produit de synchronisation de A_1 et A_2. Nous avons également montré que ces produits sont commutatifs, associatifs. Les autres résultats sont illustrés par le tableau ci-dessous :

	Traduction de Uppaal en Deadline	**Traduction de Deadline en Uppaal** invariants clos par le passé	**Traduction de Deadline en Uppaal** invariants quelconques
Comportement préservé	Bisimulation mixte –	Bisimulation mixte Bisimulation	Bisimulation mixte Bisimulation
Complexité	même nbre d'états même nbre de transitions même nbre d'horloges	même nbre d'états même nbre de transitions 1 horloge de plus	bcp plus d'états bcp plus de transitions 1 horloge de plus
Compositionnalité	–	compatible avec le produit de synchronisation	compatible avec le produit de synchronisation

La conclusion qui s'impose est que, contrairement à ce qui est suggéré dans [SY96], les automates de Uppaal permettent de modéliser plus de systèmes de temps-réel que les automates de Deadline. Cependant, pour un système donné, la complexité en état, en transition et en horloge de la modélisation par un automate de Uppaal sera plus grande que celle de la modélisation par un automate de Deadline.

Perspectives : D'ores et déjà, trois grandes directions apparaissent pour la suite de notre travail.

(1) Il serait intéressant de compléter nos comparaisons en s'intéressant à d'autres constructions de vivacité comme les actions urgentes et à d'autres opérations telles que le raffinement.

(2) Ce travail devrait nous amener à proposer une définition standard des constructions pour la vivacité des automates temporisés, ce qui n'existe pas actuellement.

(3) Une des motivations de ce travail est l'utilisation des automates avec conditions de vivacité pour la **spécification** et la **vérification**. Le second point conduit tout naturellement à étudier les questions de décidabilité soulevées par notre travail telles que : peut-on décider s'il existe un automate de Deadline qui soit bisimilaire à un automate de Uppaal donné?

Annexe A

Démonstrations Techniques

A.1 A propos de la relation d'ordre sur les régions

Proposition A.1.1 *Soient v une valuation et η un réel positif,* $[v] << [v+\eta]$.

Démonstration : On suppose que η est strictement positif sinon la proposition est évidente.

Soit $v' \in [v]$, montrons que il existe $\eta' \geq 0$ tel que $v'+\eta' \in [v+\eta]$ i.e. $v'+\eta' \equiv_I v+\eta$.

- si pour tout x de X, $v(x) > m$, alors pour tout x de X, $v(x)+\eta > m$. Donc $[v] = [v+\eta]$ et $\eta' = 0$ convient.
- sinon soit $X' = \{x, v(x) \leq m\}$.

  1. s'il existe $x \in X'$ tel que $fract(v(x)+\eta) = 0$. Soit $\eta' = (v(x)+\eta) - v'(x)$. Montrons que η' convient.
     $fract(v(x)+\eta) = 0$ et $\eta > 0$ donc $E(v(x)) < v(x)+\eta$. $v(x) \leq m$ donc $E(v'(x)) = E(v(x))$. D'où $E(v'(x)) < v(x)+\eta$ or $v(x)+\eta$ est un entier donc $v'(x) < v(x)+\eta$. Par conséquent, $\eta' > 0$.
     Les conditions 2 et 4 de la définition de $\equiv_I$ sont vérifiées car pour tout $(y,z) \in X^2$, $v(y)-v(z) = (v(y)+\eta)-(v(z)+\eta)$ et $v'(y)-v'(z) = (v'(y)+\eta)-(v'(z)+\eta)$. Reste donc à voir les conditions 1 et 3.
     Soit $y \in X$,
     (a) Vérifions la condition 1,
     - si $v(y) > m$ alors $v'(y) > m$ (condition 1). Par conséquent, $v'(y)+\eta' > m$, or $v(y)+\eta > m$ donc la condition 1 est vérifiée.
     - si $v(y) \leq m$, $E(v'(y)+\eta') = E(v'(y)+v(x)+\eta-v'(x))$. $fract(v(x)+\eta) = 0$ donc $E(v'(y)+\eta') = E(v'(y)-v'(x)) + E(v(x)+\eta)$. Or $E(v'(y)-v'(x)) = E(v(y)-v(x))$ (condition 2 car $-m \leq v(y)-v(x) \leq m$ car $v(x) \leq m$ et $v(x) \leq m$) donc $E(v'(y)+\eta') = E(v(y)-v(x)) + E(v(x)+\eta)$. D'où, $E(v'(y)+\eta') = E(v(y)+\eta)$ car $fract(v(x)+\eta) = 0$. Donc la condition 1 est vérifié.

     (b) Vérifions la condition 3.
     Si $v(y)+\eta \leq m$ et $fract(v(y)+\eta) = 0$, $fract(v(y)+\eta) = fract(v(y)-v(x)+v(x)+\eta) = fract(v(y)-v(x))$ car $fract(v(x)+\eta) = 0$. Donc $fract(v(y)-$

$v(x)) = 0$, d'où $fract(v'(y) - v'(x)) = 0$ (condition 4 car $-m \leq v(y) - v(x) \leq m$ car $v(x) \leq m$ et $v(x) \leq m$). Or $fract(v'(y) + \eta') = fract(v'(y) + v(x) + \eta - v'(x)) = fract(v'(y) - v'(x))$ car $fract(v(x) + \eta) = 0$. Donc $fract(v'(y) + \eta') = fract(v'(y) - v'(x)) = 0$. La condition 3 est vérifiée.

2. s'il n'existe pas de x dans X' tel que $fract(v(x) + \eta) = 0$.

   S'il existe soit x_0 appartenant à X' tel que $fract(v(x_0)) + fract(\eta)$ est le maximum de l'ensemble $\{(fract(v(x)) + fract(\eta))$ tel que $(fract(v(x)) + fract(\eta)) < 1$ et $x \in X'\}$.

   S'il existe soit x_1 appartenant à X' tel que $fract(v(x_1)) + fract(\eta)$ est le maximum de l'ensemble $\{(fract(v(x)) + fract(\eta))$ tel que $(fract(v(x)) + fract(\eta)) > 1$ et $x \in X'\}$.

   (a) si x_0 et x_1 existent, alors $fract(v(x_0)) < fract(v(x_1))$ Par conséquent, $0 < fract(v(x_1)) - fract(v(x_0)) < 1$ et $E(v(x_1)) - E(v(x_0)) < v(x_1) - v(x_0) < E(v(x_1)) - E(v(x_0)) + 1$. Alors comme $x_0 \in X'$ et $x_1 \in X'$, d'après la condition 2, $E(v(x_1)) - E(v(x_0)) < v'(x_1) - v'(x_0) < E(v(x_1)) - E(v(x_0)) + 1$ car $E(v(x_1) - v(x_0)) = E(v'(x_1) - v'(x_0))$. D'après la condition 1, $E(v(x_0)) = E(v'(x_0))$ et $E(v(x_1)) = E(v'(x_1))$ donc $E(v'(x_1)) - E(v'(x_0)) < v'(x_1) - v'(x_0) < E(v'(x_1)) - E(v'(x_0)) + 1$. Donc $0 < fract(v'(x_1)) - fract(v'(x_0)) < 1$ et donc $fract(v'(x_0)) < fract(v'(x_1))$.
   Soit α tel que $(1 - fract(v'(x_1))) < \alpha < (1 - fract(v'(x_0)))$. Soit $\eta' = E(\eta) + \alpha$, montrons que η' convient.
   Les conditions 2 et 4 de la définition de $\equiv_I$ sont vérifiées car pour tout $(y, z) \in X^2$, $v(y) - v(z) = (v(y) + \eta) - (v(z) + \eta)$ et $v'(y) - v'(z) = (v'(y) + \eta) - (v'(z) + \eta)$. D'autre part, il n'existe pas de x dans X' tel que $fract(v(x) + \eta) = 0$. Donc la condition 3 est également vérifiée. Reste donc à voir la condition 1.
   Soit $y \in X$,

      i. si $v(y) > m$ alors $v'(y) > m$ (condition 1). Par conséquent, $v'(y) + \eta' > m$, or $v(y) + \eta > m$ donc la condition 1 est vérifiée.

      ii. si $v(y) \leq m$,

         A. si $fract(v(y)) + fract(\eta) < 1$, alors $fract(v(y)) < fract(v(x_0))$. Par la même démonstration que 2a, $fract(v'(y)) < fract(v'(x_0))$. Donc $fract(v'(y)) + fract(\eta') < 1$. Donc $E(v'(y) + \eta') = E(v'(y) + E(\eta')$. Or $E(\eta') = E(\eta)$ (par définition de η') et $E(v'(y)) = E(v(y))$ (condition 1), donc $E(v'(y) + \eta') = E(v(y) + E(\eta) = EE(v(y) + \eta)$ car $fract(v(y)) + fract(\eta) < 1$. D'où la condition 1 est vérifiée.

         B. si $fract(v(y)) + fract(\eta) > 1$, une démonstration analogue au cas 2(a)iiA montre que la condition 1 est vérifiée.

   (b) si seul x_0 existe, soit α tel que $0 < \alpha < (1 - fract(v'(x_0)))$. Soit $\eta' = E(\eta) + \alpha$, une démonstration analogue au cas 2a montre que η' convient.

   (c) si seul x_1 existe, soit α tel que $(1 - fract(v'(x_1))) < \alpha < 0$. Soit $\eta' = E(\eta) + \alpha$, une démonstration analogue au cas 2a montre que η' convient.

Soit $v' \in [v + \eta]$, une démonstration analogue à celle qui précède montre que il existe $\eta' \geq 0$ tel que $v' - \eta' \in [v]$ i.e. $v' - \eta' \equiv_I v$.

□

A.2 Simulation des automates de Deadline par des automates de Uppaal dans le cas général

A.2.1 Résultats intermédiaires pour la démonstration du fait que $\mathcal{R}$ a la propriété de transfert

Lemme A.2.1 *Soient v une valuation et η un réel tel que pour tout t de $[0,\eta]$, $v+t$ satisfait I, alors il existe j un entier de $[\![1,n]\!]$ tel que pour tout t de $[0,\eta]$, $v+t$ satisfait I_{D_j}.*

Démonstration : Il suffit de montrer que $[v]_I$ et $[v+\eta]_I$ sont équivalents. Or $[v]_I << [v+\eta]_I$ (cf proposition 3.5.13), soit r une région telle que $[v]_I << r << [v+\eta]_I$ et $r \neq [v]$, il existe $\alpha \geq 0$ tel que $v+\alpha$ appartienne à r.
$\alpha < \eta$, sinon $r << [v+\eta]_I$ donc il existe $\beta \geq 0$ tel que $v+\eta-\beta \in r$. Donc $v+\eta-\beta \leq v+\eta \leq v+\alpha$ et $v+\eta-\beta$ et $v+\alpha$ appartiennent à r. Donc d'après la conséquence 3.5.9, $v+\eta$ appartienne à r et donc $r=[v]$ ce qui est impossible.
Par conséquent $\alpha \in [0,\eta]$, donc $v+\alpha$ satisfait I. D'où pour toute valuation v_0 de r, v_0 satisfait I (cf conséquence 3.5.3). Donc $[v]_I$ et $[v+\eta]_I$ sont équivalents. □

Lemme A.2.2 *Soient v une valuation et j un entier de $[\![1,n]\!]$, si v vérifie I_{D_j} et pour tout t de $[0,\eta[$, $v+t$ satisfait I, alors pour tout t de $[0,\eta[$, $v+t$ satisfait I_{D_j}.*

Démonstration : Soit $\eta_0 \in [0,\eta[$, pour tout t de $[0,\eta_0]$, $v+t$ satisfait I. Donc d'après le lemme A.2.1, il existe k un entier de $[\![1,n]\!]$ tel que pour tout t de $[0,\eta_0]$, $v+t$ satisfait I_{D_k}. v satisfait I_{D_k} et I_{D_j}. Or les I_i forment une partition donc $k=j$. □

Lemme A.2.3 *Soient v une valuation et j un entier de $[\![1,n]\!]$, si v vérifie I et $F_d(I_{D_j})$, v vérifie I_{D_j}.*

Démonstration : $v \vDash F_d(I_{D_j})$ donc $v \vDash I_{D_j}$ ou il existe $\eta > 0$ et $v' \in \mathbb{T}^X$ tels que $v = v' + \eta$ et $\forall t \in [0,\eta[,\ v'+t \vDash I_{D_j}$.
S'il existe $\eta > 0$ et $v' \in \mathbb{T}^X$ tels que $v = v'+\eta$ et $\forall t \in [0,\eta[,\ v'+t \vDash I_{D_j}$, alors $\forall t \in [0,\eta]$, $v'+t \vDash I$. Donc, d'après le lemme A.2.1, il existe j' un entier de $[\![1,n]\!]$ tel que pour tout t de $[0,\eta]$, $v'+t$ satisfait $I_{D_{j'}}$. Soit $\alpha = \eta \div 2$, $v+\alpha \vDash I_{D_j} \wedge I_{D_{j'}}$. Or les domaines forment une partition de l'ensemble des valuations, donc $j=j'$. D'où $v \vDash I_{D_j}$. □

Lemme A.2.4 *Soit k un entier non nul, soient v une valuation et η un réel tels que v et $v+\eta$ appartiennent à D_k. Alors pour tout t de $[0,\eta]$, $v+t$ appartient à D_k.*

Démonstration : v et $v+\eta$ appartiennent à D_k avec$k>0$. Donc $[v]$ et $[v+\eta]$ sont équivalents et v et $v+\eta$ satisfont I. Soit t dans $[0,\eta]$, $[v] << [v+t] << [0,\eta]$ (cf proposition 3.5.13) donc $v+t$ satisfait I (cf la définition de l'équivalence de deux régions. D'où pour tout t de $[0,\eta[$, $v+t$ satisfait I et v satisfait I_k. D'après le lemme A.2.2, pour tout t de $[0,\eta[$, $v+t$ satisfait I_k. □

Lemme A.2.5 *Soient v une valuation et j un entier de $[\![1,n]\!]$, si pour tout t de $[0,\eta]$, $v+t$ satisfait $F_d(I_{D_j})$, pour tout t de $[0,\eta[$, $v+t$ satisfait I_{D_j}.*

Démonstration : Soit η_0 appartenant à $[0,\eta[$, $v+\eta_0 \vDash F_d(I_{D_j})$ donc $v+\eta_0 \vDash I_{D_j}$ ou il existe $\eta_1 > 0$ et $v' \in \mathbb{T}^X$ tels que $v+\eta_0 = v'+\eta_1$ et $\forall t \in [0,\eta_1[,\ v'+t \vDash I_{D_j}$.
Si $v+\eta_0 \nvDash I_{D_j}$, soit $\alpha = \eta_0 - \eta_1$. $v' \vDash I_{D_j}$ donc, $v+\alpha \vDash I_{D_j}$. De plus, $\alpha \leq \eta_0$ car η_1 est positif.

$v+\eta \vDash F_d(I_{D_j})$ donc $v+\eta \vDash I_{D_j}$ ou il existe $\eta_2 > 0$ et $v'' \in \mathbb{T}^X$ tels que $v+\eta = v''+\eta_2$ et $\forall t \in [0,\eta_2[,\ v''+t \vDash I_{D_j}$.
Si $v+\eta \vDash I_{D_j}$, soit $\beta = \eta$ sinon $\beta = sup(\eta_0, (\eta-\eta_2))$. $\beta \geq \eta - \eta_2$ donc $\beta + \eta_2 - \eta \geq 0$. $\eta_0 \leq \eta$ et $\eta - \eta_2 \leq \eta$, donc $\beta + \eta_2 - \eta \leq \eta_2$. D'où $v'' + \beta + \eta_2 - \eta \vDash I_{D_j}$, par conséquent, $v+\beta \vDash I_{D_j}$.
$\alpha \leq \eta_0 \leq \beta$ et $v+\alpha$ et $v+\beta$ satisfont I_{D_j} alors d'après le lemme A.2.4, $v+\eta_0$ satisfait I_{D_j}. □

A.2.2 Démonstration que $\mathcal{R}$ a la propriété de transfert

Proposition A.2.6 *La relation $\mathcal{R}$ a la propriété de transfert.*

Démonstration : Soit $((q_j,v),(q_{j,k},v')) \in \mathcal{R}$.

- Soient $(q_{j_1},v_1) \in S_A$ et $\ell \in \mathcal{A}ct$ tels que $(q_j,v) \xrightarrow{\ell}_A (q_{j_1},v_1)$,
  1. $\ell \in \mathbb{T}$, alors $((q_j,v) = (q_{j_1},v_1)$.
     (a) si $k = 0$, alors d'après le lemme 3.6.4, $v \nvDash I_A(q_j)$ donc $\ell = 0$ et $v_1 = v$. $v' \vDash I_B(q_{j,k})$ car $(q',v') \in S_B$. Par conséquent, $(q_{j,k},v') \xrightarrow{\ell}_B (q_{j,k},v')$. De plus, $((q_j,v_1),(q_{j,k},v')) \in \mathcal{R}$.
     (b) si $k \neq 0$,
        i. si $\ell = 0$, alors $v_1 = v$. $v' \vDash I_B(q_{j,k})$ car $(q',v') \in S_B$. Par conséquent, $(q_{j,k},v') \xrightarrow{\ell}_B (q_{j,k},v')$. De plus, $((q_j,v_1),(q_{j,k},v')) \in \mathcal{R}$.
        ii. si $\ell \neq 0$, $(q_j,v) \xrightarrow{\ell}_A (q_{j_1},v_1)$ donc $v_1 = v+\ell$ et $\forall t \in [0,\ell[,\ v+t \vDash I_A(q_j)$. En particulier, $v \vDash I_A(q_j)$ car $\ell \neq 0$.
        D'autre part, $((q_j,v),(q_{j,k},v')) \in \mathcal{R}$ donc $v' \vDash I_B(q_{j,k})$ i.e. $v \vDash F_d(I_{j,k})$. D'où, d'après le lemme A.2.3, $v \vDash I_{j,k}$.
        De plus, $\forall t \in [0,\ell[,\ v+t \vDash I_A(q_j)$ donc, d'après le lemme A.2.2, $\forall t \in [0,\ell[,\ v+t \vDash I_{j,k}$. Donc, d'après le lemme 3.3.23, $\forall t \in [0,\ell]$, $v+t \vDash F_d(I_{j,k})$. D'où, $\forall t \in [0,\ell]$, $v+t \vDash I_B(q_{j,k})$.
        Conclusion soit $v'_1 = v'+\ell$, $(q_{j,k},v') \xrightarrow{\ell}_B (q_{j,k},v'_1)$.
        De plus, $((q_j,v_1),(q_{j,k},v'_1)) \in \mathcal{R}$
  2. $\ell \in \Sigma$, alors $\exists (q_j,\delta,\ell,\lambda,q_{j_1}) \in T_A$ tel que $v \vDash \delta$ et $v_1 = v[0 \leftarrow \lambda]$. Soit $v'_1 = v' \cup \{y = 0\}$.
     (a) si $v_1 \nvDash I_A(q_{j_1})$, soit $q'_1 = q_{j_1,0}$. $(q_{j,k}, \delta \wedge \{\neg I_A(q_{j_1})\}^\lambda, \ell, \lambda \cup \{y\}, q_{j_1,0}) \in T_B$, $v' \vDash \delta \wedge [\neg I_A(q_{j_1})]^\lambda$ (c'est pour ça qu'on à choisi $q'_1 = q_{j_1,0}$), $v'_1 = v'[0 \leftarrow (\lambda \cup \{y\})]$ et $v'_1 \vDash I_B(q_{j_1,0})$ $(v'_1(y) = 0)$, donc $(q_{j,k},v') \xrightarrow{\ell}_B (q'_1,v'_1)$. De plus, $v'_1 \vDash I_{j_1,0}$, donc $((q_{j_1},v_1),(q'_1,v'_1)) \in \mathcal{R}$.
     (b) si $v_1 \vDash I_A(q_{j_1})$, il existe $k_1 > 0$ tel que $v_1 \vDash I_{j_1,k_1}$. Soit $q'_1 = q_{j_1,k_1}$, $(q_{j,k}, \delta \wedge [I_{j_1,k_1}]^\lambda, \ell, \lambda \cup \{y\}, q_{j_1,0}) \in T_B$, $v' \vDash \delta \wedge [I_{j_1,k_1}]^\lambda$ (c'est pour ça qu'on a choisi k_1), $v'_1 = v'[0 \leftarrow (\lambda \cup \{y\})]$ et $v'_1 \vDash I_B(q_{j_1,k_1})$ $(v'_1(y) = 0)$, donc $(q_{j,k},v') \xrightarrow{\ell}_B (q'_1,v'_1)$. De plus, $v'_1 \vDash I_{j_1,k_1}$(choix de k_1), donc $((q_{j_1},v_1),(q'_1,v'_1)) \in \mathcal{R}$.

- Soient $(q'_1, v'_1) \in S_B$ et $\ell \in \mathcal{A}ct$ tels que $(q_{j,k}, v') \xrightarrow{\ell}_B (q'_1, v'_1)$. $q'_1 = q_{j_1,k_1}$. Soit $v_1 = \{v'_1\}_{|X}$.

  1. $\ell \in \mathbb{T}$,
     alors $q'_1 = q_{j,k}$, $v'_1 = v' + \ell$ et $\forall t \in [0, \ell],\ v' + t \vDash I_B(q_{j,k})$.

     (a) si $I_B(q_{j,k}) = (y = 0)$, alors $\ell = 0$ et $(q'_1, v'_1) = (q_{j,k}, v')$. On a alors $(q_j, v) \xrightarrow{\ell}_A (q_j, v)$ et $((q_j, v), (q_{j,k}, v')) \in \mathcal{R}$. De plus, $((q_j, v_1), (q_{j,k}, v'_1)) \in \mathcal{R}$.

     (b) si $I_B(q_{j,k}) = F_d(I_{j,k})$ avec $k > 0$, $\forall t \in [0, \ell],\ v + t \vDash F_d(I_{j,k})$. D'après le lemme A.2.5, $\forall t \in [0, \ell[,\ v + t \vDash I_{j,k}$. Donc $\forall t \in [0, \ell[,\ v + t \vDash I_A(q_j)$. Par conséquent, $(q_j, v) \xrightarrow{\ell}_A (q_j, v_1)$.
     $((q_j, v), (q', v')) \in \mathcal{R}$ donc $v' - v'(y) \vDash I_{j,k}$. D'où $v'_1 - v'_1(y) = v' + t - v'(y) - t \vDash I_{j,k}$. Donc, $((q_j, v_1), (q_{j,k}, v'_1)) \in \mathcal{R}$.

  2. $\ell \in \Sigma$, alors il existe $(q_{j,k}, \delta \wedge \{I_{j_1,k_1}\}^\lambda, \ell, \lambda \cup \{y\}, q_{j_1,0}) \in T_B$ telle que $v' \vDash \delta \wedge [I_{j_1,k_1}]^\lambda$, $v'_1 = v'[0 \leftarrow (\lambda \cup \{y\})]$ et $v'_1 \vDash I_B(q_{j_1,k_1})$. Donc $(q_j, \delta, \ell, \lambda, q_{j_1}) \in T_A$, $v \vDash \delta$ et $v_1 = v[0 \leftarrow \lambda]$. D'où $(q_j, v) \xrightarrow{\ell}_A (q_{j_1}, v_1)$ et $((q_{j_1}, v_1), (q'_1, v'_1)) \in \mathcal{R}$.

□

A.3 Démonstrations de plusieurs propriétés du produit de synchronisation de deux systèmes de transitions

A.3.1 L'associativité de l'opération de composition

Proposition A.3.1 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S, \mathcal{A}ct)$, $\mathcal{S}' = (S', s'_0, \rightarrow_{S'}, \mathcal{A}ct)$ et $\mathcal{S}'' = (S'', s''_0, \rightarrow_{S''}, \mathcal{A}ct)$ trois systèmes de transitions, $(\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}') \otimes_{\mathcal{S}yncr} \mathcal{S}''$ est isomorphe (donc bisimilaire) à $\mathcal{S} \otimes_{\mathcal{S}yncr} (\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}'')$.*

Démonstration : Soient $\mathcal{T} = (T, \rightarrow_T) = \mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}'$, $\mathcal{T}' = (T', \rightarrow_{T'}) = \mathcal{T} \otimes_{\mathcal{S}yncr} \mathcal{S}''$, $\mathcal{U} = (U, \rightarrow_U) = \mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}''$ et $\mathcal{U}' = (U', \rightarrow_{U'}) = \mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{U}$.

- $T' \equiv U'$, en effet soit $h : T' \longrightarrow U'$ telle que $(s \otimes_{\mathcal{S}yncr} s') \otimes_{\mathcal{S}yncr} s'' \longrightarrow s \otimes_{\mathcal{S}yncr} (s' \otimes_{\mathcal{S}yncr} s'')$. h est une bijection.

- L'image de l'état initial de $\mathcal{T}'$ par h et bien l'état initial de $\mathcal{U}'$ i.e. $h((s_0 \otimes_{\mathcal{S}yncr} s'_0) \otimes_{\mathcal{S}yncr} s''_0) = s_0 \otimes_{\mathcal{S}yncr} (s'_0 \otimes_{\mathcal{S}yncr} s''_0)$.

- Montrons que $t' \xrightarrow{\ell}_{T'} t'_1$ entraîne $h(t') \xrightarrow{\ell}_{U'} h(t'_1)$.
  $t' = (s \otimes_{\mathcal{S}yncr} s') \otimes_{\mathcal{S}yncr} s''$, $t'_1 = (s_1 \otimes_{\mathcal{S}yncr} s'_1) \otimes_{\mathcal{S}yncr} s''_1$, $h(s') = s \otimes_{\mathcal{S}yncr} (s' \otimes_{\mathcal{S}yncr} s'')$ et $h(t'_1) = s_1 \otimes_{\mathcal{S}yncr} (s'_1 \otimes_{\mathcal{S}yncr} s''_1)$.

  1. $\ell \in \mathcal{A}ct \backslash \mathcal{S}yncr$,

     (a) $s \otimes_{\mathcal{S}yncr} s' \xrightarrow{\ell}_T s_1 \otimes_{\mathcal{S}yncr} s'_1$ et $s'' = s''_1$ (règle 1).

        i. $s \xrightarrow{\ell}_S s_1$ et $s' = s'_1$ (règle 1).
           Alors $s' \otimes_{\mathcal{S}yncr} s'' = s'_1 \otimes_{\mathcal{S}yncr} s''_1$ et donc $h(t') \xrightarrow{\ell}_{U'} h(t'_1)$ (règle 1).

ii. $s = s_1$ et $s' \xrightarrow{\ell}_{S'} s'_1$ (règle 2).

Alors $s' \otimes_{\mathcal{S}yncr} s'' \xrightarrow{\ell}_{U} s'_1 \otimes_{\mathcal{S}yncr} s''_1$ (règle 1) et $h(t') \xrightarrow{\ell}_{U'} h(t'_1)$ (règle 2).

(b) $s \otimes_{\mathcal{S}yncr} s' = s_1 \otimes_{\mathcal{S}yncr} s'_1$ et $s'' \xrightarrow{\ell}_{S''} s''_1$ (règle 2).

Donc $s = s_1$ et $s' = s'_1$. On a alors $s' \otimes_{\mathcal{S}yncr} s'' \xrightarrow{\ell}_{U} s'_1 \otimes_{\mathcal{S}yncr} s''_1$ (règle 2) et $h(t') \xrightarrow{\ell}_{U'} h(t'_1)$ (règle 2).

2. $\ell \in \mathcal{S}yncr$,

alors $s \otimes_{\mathcal{S}yncr} s' \xrightarrow{\ell}_{T} s_1 \otimes_{\mathcal{S}yncr} s'_1$ et $s'' \xrightarrow{\ell}_{S''} s''_1$ (règle 3). Donc $s \xrightarrow{\ell}_{S} s_1$ et $s' \xrightarrow{\ell}_{S'} s'_1$ (règle 3).

D'où $s' \otimes_{\mathcal{S}yncr} s'' \xrightarrow{\ell}_{U} s'_1 \otimes_{\mathcal{S}yncr} s''_1$ (règle 3) et $h(t') \xrightarrow{\ell}_{U'} h(t'_1)$ (règle 3).

- De même, $u' \xrightarrow{\ell}_{U'} u'_1$ entraîne $h^{-1}(u') \xrightarrow{\ell}_{T'} h^{-1}(u'_1)$.

□

A.3.2 La commutativité de l'opération de composition

Proposition A.3.2 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S, \mathcal{A}ct)$ et $\mathcal{S}' = (S', s'_0, \rightarrow_{S'}, \mathcal{A}ct)$ deux systèmes de transitions, $\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}'$ est isomorphe (donc bisimilaire) à $\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}$.*

Démonstration : Soient $\mathcal{T} = (T, \rightarrow_T) = \mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}'$ et $\mathcal{U} = (U, \rightarrow_U) = \mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}$.

- $T \equiv U$, en effet soit $h : T \longrightarrow U$ telle que $s \otimes_{\mathcal{S}yncr} s' \longrightarrow s' \otimes_{\mathcal{S}yncr} s$. h est une bijection.
- L'image de l'état initial de $\mathcal{T}$ par h et bien l'état initial de $\mathcal{U}$ i.e. $h(s_0 \otimes_{\mathcal{S}yncr} s'_0) = s'_0 \otimes_{\mathcal{S}yncr} s_0$.
- Montrons que $t \xrightarrow{\ell}_{T} t_1$ entraîne $h(t) \xrightarrow{\ell}_{U} h(t_1)$.

  $t = s \otimes_{\mathcal{S}yncr} s'$, $t_1 = s_1 \otimes_{\mathcal{S}yncr} s'_1$, $h(s) = s' \otimes_{\mathcal{S}yncr} s$ et $h(t_1) = s'_1 \otimes_{\mathcal{S}yncr} s_1$.

  1. $\ell \in \mathcal{A}ct \backslash \mathcal{S}yncr$,

     (a) $s \xrightarrow{\ell}_{S} s_1$ et $s' = s'_1$ (règle 1).

     Alors $h(t) \xrightarrow{\ell}_{U} h(t_1)$ (règle 2).

     (b) $s = s_1$ et $s' \xrightarrow{\ell}_{S'} s'_1$ (règle 2).

     Alors $h(t) \xrightarrow{\ell}_{U} h(t_1)$ (règle 1).

  2. $\ell \in \mathcal{S}yncr$,

     Alors $s \xrightarrow{\ell}_{S} s_1$ et $s' \xrightarrow{\ell}_{S'} s'_1$ (règle 3). D'où $h(t) \xrightarrow{\ell}_{S'} h(t_1)$ (règle 3).

- De même, $u \xrightarrow{\ell}_{U} u_1$ entraîne $h^{-1}(u) \xrightarrow{\ell}_{T} h^{-1}(u_1)$.

□

A.3.3 La bisimulation est une congruence par rapport à l'opération de composition

Proposition A.3.3 *Soient $\mathcal{S} = (S, s_0, \rightarrow_S)$ et $\mathcal{S}' = (S', s'_0, \rightarrow_{S'})$ deux systèmes de transitions bisimilaires, alors $\mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}''$ et $\mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}''$ sont bisimilaires.*

Démonstration : Soient $\mathcal{T} = (T, \rightarrow_T) = \mathcal{S} \otimes_{\mathcal{S}yncr} \mathcal{S}''$ et $\mathcal{U} = (U, \rightarrow_U) = \mathcal{S}' \otimes_{\mathcal{S}yncr} \mathcal{S}''$.
Soit $\mathcal{R}$: $\mathcal{S} \leftrightarrow \mathcal{S}'$ et $(s_0, s'_0) \in \mathcal{R}$.
Soit $\mathcal{R}'$ incluse dans $T \times U$ définie par :
$(s \otimes_{\mathcal{S}yncr} s'', s' \otimes_{\mathcal{S}yncr} s''_1) \in \mathcal{R}'$ si et seulement si $(s, s') \in \mathcal{R}$ et $s'' = s''_1$.

Montrons que $\mathcal{R}'$ est une bisimulation de $\mathcal{T}$ et $\mathcal{U}$.
L'état initial de $\mathcal{T}$ et celui de $\mathcal{U}$ sont en relation par $\mathcal{R}'$ i.e. $(s_0 \otimes_{\mathcal{S}yncr} s''_0, s'_0 \otimes_{\mathcal{S}yncr} s''_0) \in \mathcal{R}'$.
Ensuite, soient $(t, u) \in \mathcal{R}'$ et $\ell \in \mathcal{A}ct$,

1. Soit $t_1 \in T$ tel que $t \xrightarrow{\ell}_T t_1$.
   $t = s \otimes_{\mathcal{S}yncr} s''$ et $t_1 = s_1 \otimes_{\mathcal{S}yncr} s''_1$.
   (a) $\ell \in \mathcal{A}ct \backslash \mathcal{S}yncr$
      i. $s \xrightarrow{\ell}_S s_1$ et $s'' = s''_1$ (règle 1).
      $(t, u) \in \mathcal{R}'$ donc $u = s' \otimes_{\mathcal{S}yncr} s''$ avec $(s, s') \in \mathcal{R}$.
      $(s, s') \in \mathcal{R}$, donc $\exists s'_1 \in S'$ tel que $s' \xrightarrow{\ell}_{S'} s'_1$ et $(s_1, s'_1) \in \mathcal{R}$.
      D'où : $s' \otimes_{\mathcal{S}yncr} s'' \xrightarrow{\ell}_U s'_1 \otimes_{\mathcal{S}yncr} s''_1$ (règle 1) et $(s_1 \otimes_{\mathcal{S}yncr} s'', s'_1 \otimes_{\mathcal{S}yncr} s''_1) \in \mathcal{R}'$.
      Donc $\exists u_1 \in U (u_1 = s'_1 \otimes_{\mathcal{S}yncr} s''_1)$, tel que : $u \xrightarrow{\ell}_U u_1$ et $(t_1, u_1) \in \mathcal{R}'$.
      ii. $s = s_1$ et $s'' \xrightarrow{\ell}_{S''} s''_1$ (règle 2).
      $(t, u) \in \mathcal{R}'$ donc $u = s' \otimes_{\mathcal{S}yncr} s''$ avec $(s, s') \in \mathcal{R}$.
      Soit $u_1 = s' \otimes_{\mathcal{S}yncr} s''_1$, $u \xrightarrow{\ell}_U u_1$ (règle 2) et $(t_1, u_1) \in \mathcal{R}'$ ($(s, s') \in \mathcal{R}$).
   (b) $\ell \in \mathcal{S}yncr$,
      alors $s \xrightarrow{\ell}_S s_1$ et $s'' \xrightarrow{\ell}_{S''} s''_1$ (règle 3).
      $(s, s') \in \mathcal{R}$, donc $\exists s'_1 \in S'$ tel que $s' \xrightarrow{\ell}_{S'} s'_1$ et $(s_1, s'_1) \in \mathcal{R}$. D'où $s' \otimes_{\mathcal{S}yncr} s'' \xrightarrow{\ell}_U s'_1 \otimes_{\mathcal{S}yncr} s''_1$ (règle 3) et $(s_1 \otimes_{\mathcal{S}yncr} s''_1, s'_1 \otimes_{\mathcal{S}yncr} s''_1) \in \mathcal{R}'$.
      Donc $\exists u_1 \in U (u_1 = s'_1 \otimes_{\mathcal{S}yncr} s''_1)$, tel que : $u \xrightarrow{\ell}_U u_1$ et $(t_1, u_1) \in \mathcal{R}'$.
2. Soit $u_1 \in U$ tel que $u \xrightarrow{\ell}_U u_1$.
   De manière analogue, on montre qu'il existe $t_1 \in T$, tel que : $t \xrightarrow{\ell}_T t_1$ et $(t_1, u_1) \in \mathcal{R}'$

□

A.4 Démonstration du théorème de correction pour les automates de Uppaal

Théorème A.4.1 *Soient $A = (\Sigma, Q_A, q_{0,A}, X_A, T_A, I_A)$ et $B = (\Sigma, Q_B, q_{0,B}, X_B, T_B, I_B)$ deux automates de Uppaal. Soient $\mathcal{A}ct = \Sigma \cup \mathbb{T}$ et $\mathcal{S}yncr = J \cup \mathbb{T}$. $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B$ et $\mathcal{S}_{A \otimes_J B}$ sont isomorphes donc bisimilaires.*

Démonstration : On note $\mathcal{S}_{A\otimes_J B} = (S, \rightarrow)$ et $\mathcal{S}_A \otimes_{\mathcal{S}yncr} \mathcal{S}_B = (U, \rightsquigarrow)$

- Cherchons une bijection de S dans U.

  Soit $h : S \rightarrow U$ tel que $((q_A, q_B), [v_A v_B]) \rightarrow (q_A, v_A) \otimes_{\mathcal{S}yncr} (q_B, v_B)$. C'est une bijection car $((q_A, q_B), [v_A v_B]) \in S$ si et seulement si $(q_A, v_A) \otimes_{\mathcal{S}yncr} (q_B, v_B) \in U$, en effet, $[v_A v_B] \vDash I_{A\otimes_J B}((q_A, q_B))$ si et seulement si $v_A \vDash I_A(q_A) \wedge v_B \vDash I_B(q_B)$.

- $h(s_{0,S}) = h(((q_{0,A}, q_{0,B}), v_0)) = s_{0,A} \otimes_{\mathcal{S}yncr} s_{0,B}$, donc l'image par h de l'état initial de $\mathcal{S}$ est celui de $\mathcal{U}$.

- Soient $s \in S$, $s' \in S$ et $\ell \in \mathcal{A}ct$, montrons que $s \xrightarrow{\ell} s'$ entraîne $h(s) \overset{\ell}{\rightsquigarrow} h(s')$.

  Soit $s = ((q_A, q_B), [v_A v_B])$ et $s' = ((q'_A, q'_B), [v'_A v'_B])$ alors $h(s) = ((q_A, v_A), (q_B, v_B))$ et $h(s') = ((q'_A, v'_A), (q'_B, v'_B))$

  1. $\ell \in \mathbb{T}$, $(q_A, q_B) = (q'_A, q'_B)$, $[v_A v_B] = [v'_A v'_B] + \ell$ et $\forall t,\ 0 \leq t \leq \ell,\ [v_A v_B] + t \vDash I_{A\otimes_J B}((q_A, q_B))$(cas 1 de la définition du système de transition associé à un automate avec invariants).

     Donc $q_A = q'_A, v'_A = v_A + \ell$ et $\forall t,\ 0 \leq t \leq \ell,\ v_A + t \vDash I_A(q_A)$ car $I_{A\otimes_J B}((q_A, q_B)) = I_A(q_A) \wedge I_B(q_B)$. D'où $(q_A, v_A) \xrightarrow{\ell}_A (q'_A, v'_A)$ (cas 1 de la définition d'un automate temporisé).

     De même $(q_B, v_B) \xrightarrow{\ell}_B (q'_B, v'_B)$.

     D'où $h(s) \overset{\ell}{\rightsquigarrow} h(s')$ (règle 3 de la composition des systèmes de transition).

  2. $\ell \in \Sigma$, $\exists\ ((q_A, q_B), \delta, \ell, \lambda, (q'_A, q'_B)) \in T_{A\otimes_J B}$, tel que $[v_A v_B] \vDash \delta$ et $[v'_A v'_B] = [v_A v_B][\lambda \leftarrow 0]$ et $[v'_A v'_B] \vDash I_{A\otimes_J B}((q'_A, q'_B)) = I_A(q'_A) \wedge I_B(q'_B)$(cas 2 de la définition du système de transition associé à un automate avec invariants).

     (a) $\ell \in J$, alors $\delta = \delta_A \wedge \delta_B$, $\lambda = \lambda_A \sqcup \lambda_B$, $(q_A, \delta_A, \ell, \lambda_A, q'_A) \in T_A$ et $(q_B, \delta_B, \ell, \lambda_B, q'_B) \in T_B$(cas 2 de la composition d'automates avec invariants).

     De plus $v_A \vDash \delta_A$, $v'_A = v_A[\lambda_A \leftarrow 0]$ et $v'_A \vDash I_A(q'_A)$. Donc $(q_A, v_A) \xrightarrow{\ell}_A (q'_A, v'_A)$ (cas 2 de la définition d'un automate temporisé avec invariants).

     De même $(q_B, v_B) \xrightarrow{\ell}_B (q'_B, v'_B)$.

     D'où $h(s) \overset{\ell}{\rightsquigarrow} h(s')$ (règle 3 de la composition des systèmes de transition).

     (b) $\ell \in \Sigma \backslash J$

     i. $(q_A, \delta, \ell, \lambda, q'_A) \in T_A$, et $q_B = q'_B$, $v_B = v'_B$ (cas 1.a de la composition d'automates avec invariants).

     De plus $v_A \vDash \delta$, $v'_A = v_A[\lambda \leftarrow 0]$ et $v'_A \vDash I_A(q'_A)$. Donc $(q_A, v_A) \xrightarrow{\ell}_A (q'_A, v'_A)$ (cas 2 de la définition d'un automate temporisé avec invariants)

     D'où $h(s) \overset{\ell}{\rightsquigarrow} h(s')$ (règle 1 de la composition des systèmes de transition).

     ii. $(q_B, \delta, \ell, \lambda, q'_B) \in T_B$ et $q_A = q'_A$, $v_A = v'_A$ (cas 1.b de la composition d'automates avec invariants). C'est un cas analogue au cas précédent, donc $h(s) \overset{\ell}{\rightsquigarrow} h(s')$.

- Soient $s \in U$, $s' \in U$ et $\ell \in \mathcal{A}ct$, montrons que $s \overset{\ell}{\rightsquigarrow} s'$ entraîne $h^{-1}(s) \xrightarrow{\ell} h^{-1}(s')$

Soit $s = ((q_A, v_A), (q_B, v_B))$ et $s' = ((q'_A, v'_A), (q'_B, v'_B))$ alors $h(^{-1}s) = ((q_A, q_B), [v_A v_B])$ et $h^{-1}(s') = ((q'_A, q'_B), [v'_A v'_B])$

1. $\ell \in \mathcal{S}yncr$ alors $(q_A, v_A) \xrightarrow{\ell}_A (q'_A, v'_A)$ et $(q_B, v_B) \xrightarrow{\ell}_B (q'_B, v'_B)$ (règle 3 de la composition des systèmes de transition).
   (a) $\ell \in J$ alors $\exists\ (q_A, \delta_A, \ell, \lambda_A, q'_A) \in T_A$, tel que $v_A \vDash \delta_A$, $v'_A = v_A[\lambda_A \leftarrow 0]$ et $v'_A \vDash I_A(q'_A)$ (cas 2 de la définition du système de transition associé à un automate). De même $\exists\ (q_B, \delta_B, \ell, \lambda_B, q'_B) \in T_B$, tel que $v_B \vDash \delta_B$, $v'_B = v_B[\lambda_B \leftarrow 0]$ et $v'_B \vDash I_B(q'_B)$.
   Donc $((q_A, q_B), \delta_A \wedge \delta_B, \ell, \lambda_A \sqcup \lambda_B, (q'_A, q'_B)) \in T_{A \otimes_J B}$ (cas 2 de la composition de deux automates).
   De plus, $[v_A v_B] \vDash \delta_A \wedge \delta_B$, $[v'_A v'_B] = [v_A v_B][\lambda_A \sqcup \lambda_B \leftarrow 0]$ et $[v'_A v'_B] \vDash I_{A \otimes_J B}((q'_A, q'_B)) = I_A(q'_A) \wedge I_B(q'_B)$ donc $h^{-1}(s) \xrightarrow{\ell} h^{-1}(s')$ (cas 2 de la définition du système de transition associé à un automate).
   (b) $\ell \in \mathbb{T}$ alors $q_A = q'_A$, $v_A = v'_A + \ell$ et $\forall t,\ 0 \leq t \leq \ell,\ v_A + t \vDash I_A(q_A)$ (cas 1 de la définition du système de transition associé à un automate). De même $q_B = q'_B$, $v_B = v'_B + \ell$ et $\forall t,\ 0 \leq t \leq \ell,\ v_B + t \vDash I_B(q_B)$.
   Donc $(q_A, q_B) = (q'_A, q'_B)$, $[v_A v_B] = [v'_A v'_B] + \ell$ et $\forall t,\ 0 \leq t \leq \ell,\ [v_A v_B] + t \vDash I_{A \otimes_J B}((q_A, q_B)) = I_A(q_A) \wedge I_B(q_B)$. D'où $h^{-1}(s) \xrightarrow{\ell} h(^{-1}s')$ (cas 1 de la définition du système de transition associé à un automate).
2. $\ell \in \Sigma \backslash J$,
   (a) $(q_A, v_A) \xrightarrow{a}_A (q'_A, v'_A)$, $(q_B, v_B) = (q'_B, v'_B)$ (règle 1 de la composition des systèmes de transition).
   Donc $\exists\ (q_A, \delta_A, \ell, \lambda_A, q'_A) \in T_A$, tel que $v_A \vDash \delta_A$, $v'_A = v_A[\lambda_A \leftarrow 0]$ et $v'_A \vDash I_A(q'_A)$(cas 2 de la définition du système de transition associé à un automate). On en déduit que $((q_A, q_B), \delta_A, \ell, \lambda_A, (q'_A, q'_B)) \in T_{A \otimes_J B}$ (cas 1.a de la composition de deux automates).
   De plus $[v_A v_B] \vDash \delta_A$, $[v'_A v'_B] = [v_A v_B][\lambda_A \leftarrow 0]$ et $[v'_A v'_B] \vDash I_{A \otimes_J B}((q'_A, q'_B))$ car $(q'_B, v'_B) \in S_B$ donc $v'_B \vDash I(q'_B)$, donc $h^{-1}(s) \xrightarrow{\ell} h^{-1}(s')$ (cas 2 de la définition du système de transition associé à un automate avec invariants).
   (b) $(q_B, v_B) \xrightarrow{\ell}_B (q'_B, v'_B)$, $(q_A, v_A) = (q'_A, v'_A)$ (règle 1 de la composition des systèmes de transition). C'est un cas analogue au cas précédent , donc $h^{-1}(s) \xrightarrow{\ell} h^{-1}(s')$.

□

Bibliographie

[ACD93] R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1):2–34, 1993.

[AD90] R. Alur and D. Dill. Automata for modeling real-time systems. In *Proc. 17th Int. Coll. Automata, Languages, and Programming (ICALP'90), Warwick, GB, July 1990*, volume 443 of *Lecture Notes in Computer Science*, pages 322–335. Springer-Verlag, 1990.

[Alu98] R. Alur. Timed automata. In *Verification of Digital and Hybrid Systems, Proc. NATO-ASI Summer School, Antalya, Turkey, 26 May - 6 June 1997*. Springer-Verlag, 1998. To appear.

[BCJ+] J. Bengtsson, P. Christensen, J. Jensen, K. Larsen, F. Larsson, P . Pettersson, T. Sorensen, and W. Yi. UPPAAL - a tool suite for validation and verification of real-time systems. User guide of UPPAAL version 2.02.

[BGP96] B. Berard, P. Gastin, and A. Petit. On the power of non-observable actions in timed automata. In *Proc. 13th Ann. Symp. Theoretical Aspects of Computer Science (STACS'96), Grenoble, France, Feb. 1996*, volume 1046 of *Lecture Notes in Computer Science*, pages 257–268. Springer-Verlag, 1996.

[BGP97] B. Bérard, P. Gastin, and A. Petit. Refinement and abstraction for timed languages. Research Report LSV-97-3, Lab. Specification and Verification, ENS de Cachan, Cachan, France, April 1997.

[BS97] S. Bornot and J. Sifakis. Relating time progress and deadlines in hybrid systems. In *Proc. 16th Conf. Found. of Software Technology and Theor. Comp. Sci. (FST&TCS'96), Hyderabad, India, Dec. 1996*, volume 1201 of *Lecture Notes in Computer Science*, pages 286–300. Springer-Verlag, 1997.

[DOTY96] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool KRONOS. In *Proc. Workshop Hybrid Systems III: Verification and Control, New Brunswick, NJ, USA, Oct. 1995*, volume 1066 of *Lecture Notes in Computer Science*, pages 208–219. Springer-Verlag, 1996.

[DY95] C. Daws and S. Yovine. Two examples of verification of multirate timed automata with KRONOS. In *Proc. 1995 IEEE Real-Time Systems Symposium (RTSS'95), Pisa, Italy, Dec. 1995*, pages 66–75. IEEE Computer Society Press, 1995.

[HHW95] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. HYTECH: the next generation. In *Proc. 16th IEEE Real-Time Systems Symposium, Dec. 1995, Pisa, Italy*, pages 56–65, 1995.

[HKW95] T. A. Henzinger, P. W. Kopke, and H. Wong-Toi. The expressive power of clocks. In *Proc. 22nd Int. Coll. Automata, Languages, and Programming (ICALP'95), Szeged, Hungary, July 1995*, Lecture Notes in Computer Science, pages 335–346. Springer-Verlag, 1995.

[HSLL97] K. Havelund, A. Skou, K. G. Larsen, and K. Lund. Formal modelling and analysis of an audio/video protocol: An industrial case study using UPPAAL. In *Proc. 18th IEEE Real-Time Systems Symposium (RTSS'97), San Francisco, CA, USA, Dec. 1997*, pages 2–13, 1997.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice Hall Int., 1989.

[SY96] J. Sifakis and S. Yovine. Compositional specification of timed systems. In *Proc. 13th Ann. Symp. Theoretical Aspects of Computer Science (STACS'96), Grenoble, France, Feb. 1996*, volume 1046 of *Lecture Notes in Computer Science*, pages 347–359. Springer-Verlag, 1996.

[Yov97] S. Yovine. Kronos: A verification tool for real-time systems. *Journal of Software Tools for Technology Transfer*, 1(1/2):??, 1997.