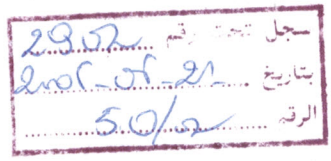
Université ABOU-BEKR BELKAÏD Tlemcen
Faculté des sciences de l'ingénieur

Mémoire
Pour obtenir le diplôme de magister

Dans le cadre du magister API
Option : Productique

Présenté par

Latéfa GHOMRI



Modélisation structurelle utilisant les automates hybrides et les réseaux de Petri hybrides en vue de la synthèse de contrôleur des systèmes dynamiques hybrides

Co-encadré par M. Zaki SARI
M. Hassane ALLA

Jury

Président	M. Brahim CHERKI	Maître de conférences à l'université de Tlemcen
Examinateur	M. Noureddine GHOUALI	Professeur à l'université de Tlemcen
Examinateur	M. Hafid HAFFAF	Maître de conférences à l'université d'Oran
Examinateur	Mme Rachida GHOUL	Maître de conférences à l'université d'Oran
Encadreur	M. Zaki SARI	Maître de conférences à l'université de Tlemcen
Encadreur	M. Hassane ALLA	Professeur à l'université Joseph Fourier Grenoble

Sommaire

Introduction générale

Chronologiquement, les systèmes continus ont été les premiers à être étudiés, leur modélisation s'effectue généralement au moyen des équations différentielles. Ces systèmes traitent des grandeurs continues, comme la température, la pression, le flux, etc.,

Le progrès de l'informatique et l'utilisation des ordinateurs pour la commande des systèmes ont créé des situations dans lesquelles la connaissance exacte des variations des grandeurs continues n'est pas intéressante et seules certaines valeurs seuils de ces variables continues présentent un intérêt. Ces systèmes sont les systèmes à événements discrets.

Pendant longtemps l'automatique a traité séparément les systèmes continus et les systèmes à événement discrets, pour chacune de ces deux classes de systèmes existent une théorie, des méthodes et des outils pour résoudre les problèmes qui se posent à elle. Cependant, la séparation entre le monde des systèmes continus et celui des systèmes à événements discrets, n'est pas aussi nette et la plupart des systèmes réels présentent à la fois des aspects continus et discrets. La prise de conscience de ce fait rapproche de plus en plus les deux communautés, ce qui a permis de favoriser les échanges entre les différents spécialistes, qu'ils soient issus du monde de l'automatique continue ou de celui de l'automatique événementielle. Les résultats de travaux dans ce domaine portent essentiellement sur les problèmes de modélisation et de commande des systèmes dynamiques hybrides. Au monde des automaticiens s'est associée la communauté informatique qui, en s'intéressant au problème de la vérification s'est approchée de la modélisation des systèmes à événements discrets puis à celle des systèmes hybrides.

Dans ce travail nous nous sommes intéressés à une classe particulière des systèmes dynamiques hybrides, à savoir la classe des systèmes à flux continus supervisés par des systèmes à événements discrets. Cette classe comporte des systèmes hybrides positifs et linéaires par morceaux. Un système hybride est dit positif si ses variables d'état prennent des valeurs positives dans le temps. Et il est dit linéaire par morceaux si les lois décrivant son évolution continue sont formulées au moyen d'équations différentielles linéaires. Un effort particulier a été apporté à

l'étude de cette classe pour deux raisons principales. D'abord, elle est suffisamment riche pour permettre une modélisation réaliste de nombreux problèmes. Ensuite, sa simplicité relative permet une conception facile d'outils et de modèles pour sa description et son analyse.

Pour intégrer les aspects continu et événementiel au sein d'un même formalisme, trois approches sont possibles. Cela dépend du modèle dominant, à partir duquel s'effectue l'extension. Ces approches sont l'approche continue, l'approche discrète et l'approche mixte.

La première approche consiste à intégrer l'aspect discret dans un formalisme continu, et ceci par l'introduction de variables booléenne ou entière dans un système d'équations.

L'approche discrète à pour principe d'intégrer l'aspect continu dans un formalisme pour les systèmes à événements discrets comme les réseaux de Petri. Le champ d'application de ces derniers a été étendu pour prendre en compte les systèmes dynamiques hybrides. Le premier pas dans cette direction a été pris en 1987 par David et Alla [DA87], en introduisant les réseaux de Petri continus et puis les réseaux de Petri hybrides. Depuis, plusieurs extensions des réseaux de Petri ont été présentées pour permettre la modélisation de l'aspect hybride.

Le modèle réseaux de Petri hybride présente l'avantage d'être un modèle puissant pour la description des systèmes hybrides, dans le sens ou le modèle est conçu d'une manière intuitive. De plus, ce modèle hérite de tous les avantages des réseaux de Petri. Des exemples de ces avantages sont : ce modèle n'exige pas une énumération exhaustive de l'espace d'état et peut représenter d'une manière finie un système dont l'espace d'états atteignable est infini. Il permet une modélisation modulaire, où la structure de chaque module est conservée dans le modèle composé. Et enfin les réseaux de Petri permettent une représentation explicite du parallélisme, de la synchronisation et des conflits.

La troisième approche pour l'intégration des aspects continu et discret dans un même modèle est dite approche mixte. Cette approche repose sur l'utilisation d'un modèle à événements discrets comme moniteur de système d'équations. A un instant de temps donné, seul un des modèles est actif, soit il y a modification de la structure des équations formant le modèle continu et le temps n'évolue pas, soit le temps évolue mais la structure des équations ne change pas. Cette caractéristique se trouve dans le modèle automates hybrides, qui constitue le modèle le plus général pour la modélisation des systèmes dynamiques hybrides puisqu'ils peuvent modéliser la plus grande variété de dynamiques continues. Ce modèle est bien adapté pour l'analyse des systèmes dynamiques hybrides.

Dans ce mémoire nous nous sommes intéressés à la modélisation en vue du contrôle des systèmes à flux continu contrôlé par des systèmes à événements discrets. Un système à flux continu peut traiter soit un flux de matière soit un flot important de produits. Nous avons introduit une variété des réseaux de Petri hybrides, dite réseaux de Petri hybrides D-élémentaires, pour la modélisation de cette classe de système. Dans ce modèle, c'est la partie discrète qui pilote l'ensemble du système. Le modèle considéré combine un réseau de Petri T-temporel et un réseau de Petri continu. Les intervalles de temps associés aux transitions du RdP T-temporel introduisent un indéterminisme quand au comportement du modèle hybride. Cet indéterminisme permet d'utiliser les transitions comme point de commande pour le contrôle des systèmes à flux continu.

La synthèse de contrôleur pour les systèmes dynamiques (autonomes, temporisé ou hybride) passe en général par les trois étapes suivantes :

1. La représentation du comportement du système par un modèle ;
2. La définition des spécifications exigées sur le comportement de ce système ;
3. La synthèse du contrôleur qui restreint le comportement du système au comportement exigé, en utilisant un algorithme de synthèse de contrôleur ;

Les algorithmes utilisent, traditionnellement, les automates à cause de leur facilité de manipulation formelle. Cependant ce formalisme n'est pas le plus adapté pour la description du système. Un des formalismes les plus adaptés pour cette première étape est le réseau de Petri. Une technique consiste à modéliser le système par un réseau de Petri, de traduire le modèle obtenu en automate hybride et d'utiliser ce dernier pour la synthèse du contrôleur.

Un algorithme permettant la traduction des réseaux de Petri hybrides D-élémentaire est présenté dans ce manuscrit. La traduction des réseaux de Petri en automates permet d'associer la puissance de modélisation des réseaux de Petri à la puissance d'analyse des automates.

Ce mémoire consacré à la modélisation des systèmes à flux continu, est composé de cinq parties :

Dans la première partie, nous introduisons les notions fondamentales relatives aux systèmes dynamiques hybrides. Leur définition ainsi que les principaux phénomènes hybrides sont présentés. Nous passons en revue les principaux formalismes de modélisation des systèmes hybrides.

La deuxième partie est dédiée aux deux formalismes qui nous ont particulièrement intéressés dans ce travail, à savoir les automates hybrides et les réseaux de Petri hybrides. En premier lieu, nous définissons les modèles à

événements discrets à partir desquels les modèles hybrides ont été définis. Il s'agit des automates à états finis et des réseaux de Petri. La temporisation de ces deux modèles est ensuite présentée. Et enfin les modèles hybrides sont présentés.

Dans la troisième partie nous définissons les réseaux de Petri hybrides D-élémentaires, ce formalisme permet la modélisation des systèmes à flux continu commandés par des systèmes à événements discrets. Nous présentons une procédure pour la traduction de ce modèle en automates hybrides, la procédure s'effectue d'une manière hiérarchique, d'abord la partie discrète est traduite en automate temporisé, ensuite nous définissons un automate hybride pour chaque sommet de la partie discrète.

Dans la quatrième partie nous utilisons l'application HyTech pour l'analyse de l'automate hybride résultant de l'application de la procédure de traduction sur les réseaux de Petri hybride D-élémentaires. L'application de la procédure de traduction donne une sur-approximation de l'automate hybride, l'application HyTech permet de calculer l'automate hybride atteignable exact.

La cinquième partie aborde la problématique générale de la synthèse de contrôleurs. Les notions et concepts de base donnés par Ramadge et Wonham [RW87] [RW89] pour la synthèse de contrôleur des systèmes à évènements discrets sont présentés. Notre perspective immédiate de ce travail est l'utilisation du modèle RdPH D-élémentaire pour la synthèse de contrôleur des systèmes dynamiques hybrides.

Les Systèmes Dynamiques Hybrides

Les systèmes dynamiques hybrides (SDH) sont généralement définis comme étant des systèmes dynamiques combinant une partie continue et une autre événementielle. La modélisation l'analyse et la commande des SDH n'a jamais attiré autant d'attention qu'actuellement. Ce qui a résulté en la conception de plusieurs formalismes pour leur modélisation. Dans ce chapitre nous présentons les SDH ainsi que leurs caractéristiques. Nous passons en revue la classe des formalismes, conçus pour leur modélisation, qui nous a particulièrement intéressés dans ce travail.

Les systèmes dynamiques hybrides sont des systèmes comportant un système continu interagissant avec et/ou supervisé par un système à événements discrets. Un système se caractérise par la nature de ses variables d'états, qui peuvent être continues ou discrètes. Le premier type est défini sur l'ensemble des réel $\Re$, alors que le second prend ses valeurs dans un ensemble dénombrable (entiers, booléens,...). Pour définir un système dynamique hybride, nous allons d'abord introduire les notions de systèmes continus et de système à événements discrets.

1.1. Classification des systèmes dynamiques

Suivant la nature des variables d'état et de la variable d'état indépendante qui est le temps, on peut classer les systèmes dynamiques comme suit :

1.1.1. Les systèmes dynamiques continus

Dans cette catégorie la variable temps est toujours présente, elle peut être continue ou discrète [Cas69]. Le comportement de ce type de système est décrit par des variables d'état continues, le temps lui peut être continu, le système est dit à temps continu (*figure 1.1.a*), ou discret, c'est le cas des systèmes discrétisés, dont l'exemple le plus répandu sont les systèmes échantillonnés (*figure 1.1.b*). La représentation de l'évolution de ces grandeurs, traduisant le comportement dynamique du procédé, est alors une représentation mathématique à l'aide de modèles continus, tels que les équations différentielles, les équations récurrentes, les équations aux différences, les fonctions de transfert, ou encore les équations d'état.

Un système continu est formellement défini comme suit :

Définition 1.1 (Système Dynamique Continu) : Un système dynamique continu est un triplet (X, X_0, f) tel que :

- $X \subseteq \Re^n$ est l'espace d'état continu ;
- $X_0 \subseteq X$ est l'espace d'état initial du système ;
- $f : X \rightarrow \Re^n$ est le champ de vecteur continu ;

❑

Le comportement d'un système continu est le plus souvent décrit par une équation différentielle de la forme

$$\dot{x} = f(x)$$

Où $x \in X$ est l'état du système, le comportement d'un système continu est caractérisé par la solution de l'équation différentielle $\dot{x} = f(x)$ à partir de l'état initial $x_0 \in X_0$.

Un système continu est dit linéaire s'il est modélisé par une équation différentielle de la forme

$$\dot{x} = A.x$$

Où $A \in \Re^{nxn}$ est une matrice constante.

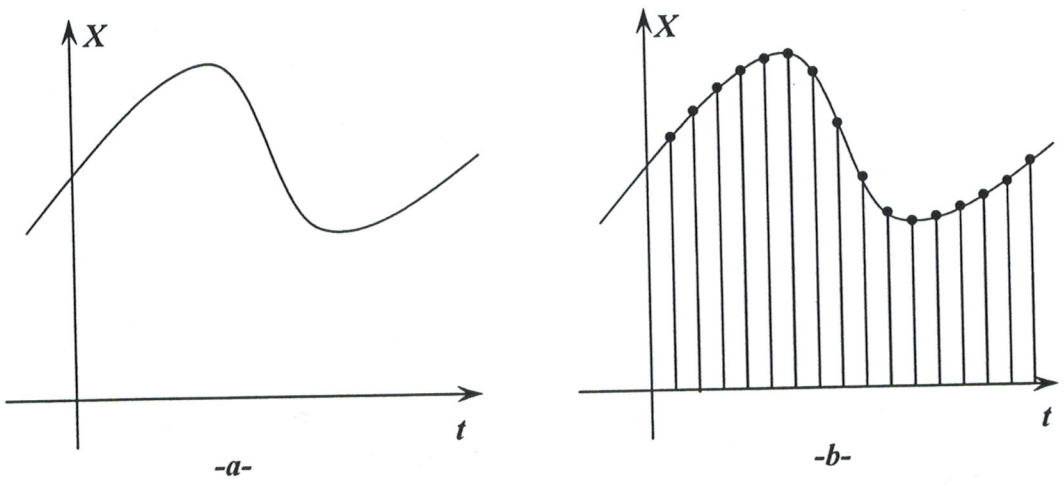


Figure 1.1. Système dynamique continu, -a- Système à temps continu, -b- Système échantillonné.

1.1.2. Les systèmes dynamiques à événement discrets

Un système à événements discrets SED est décrit par des variables d'état discrètes [Cas96], qui prennent leurs valeurs dans un ensemble dénombrable. Ce type de système peut être autonome (non temporisé) ou temporisé. Dans le cas d'un SED autonome, la variable temps est symbolique, c'est-à-dire qu'elle est utilisée pour définir une chronologie entre les événements. Dans le cas des SED temporisés, le temps peut être continu (dense) ou discret. Dans le premier cas (*figure 1.2.a*), à chaque événement est rattaché l'instant de son occurrence qui prend ses valeurs dans $\Re$, cas d'un automate temporisé par exemple. Dans le second cas (SED temporisés avec

temps discret) le temps n'est défini qu'à des instants de $\Re$ (*figure 1.2.b*). L'exécution d'une séquence d'instructions sur un processeur rentre dans cette dernière catégorie, puisque ces exécutions ne peuvent s'effectuer qu'à chaque top d'horloge du processeur. Les SED peuvent être modélisés par les automates à états, les réseaux de Petri, les chaînes de Markov, l'algèbre (Max, +), … etc.

Un système à évènements discrets est formellement défini comme suit :

Définition 1.2 (Système Dynamique à Evènements Discrets) : Un système dynamique à évènements discrets est un triplet (Q, Q_0, δ) tel que :

- Q est un ensemble fini ou infini d'états ;
- $Q_0 \subseteq Q$ est l'ensemble des états initiaux ;
- δ: $Q \times Q$ est la fonction de transition entre états, elle détermine le comportement du système ;

❑

Notons que le comportement d'un même système physique peut être décrit par des variables d'état continues ou discrètes, suivant les phénomènes auxquels on s'intéresse, comme le montre l'exemple 1.1 ci-dessous.

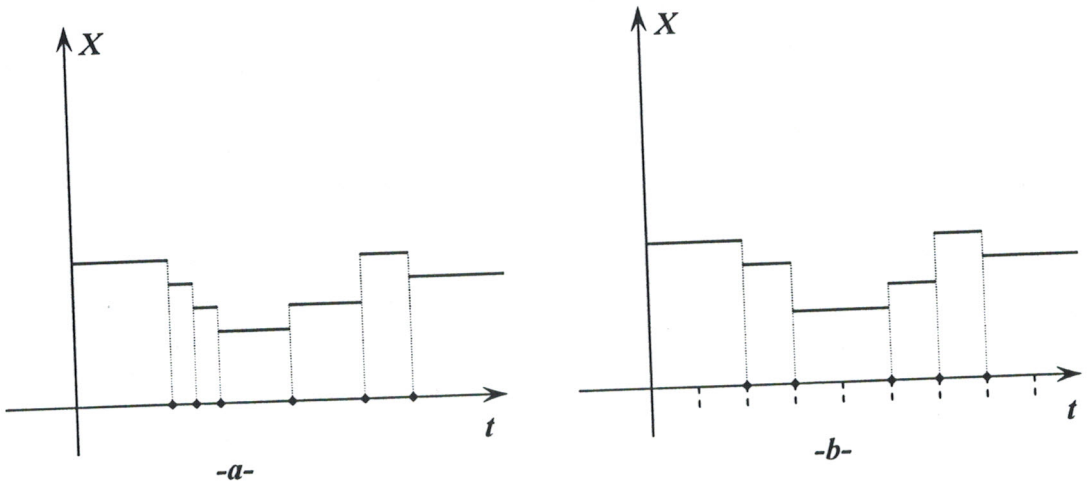


Figure 1.2. Système à événements discrets, -a- la variable temps est dense ou continue, -b- la variable temps est discrète.

Exemple 1.1 : Considérons un bac qui peut être rempli et vidé, comme montré en figure 1.3 ci-dessous. Le comportement de ce système peut être décrit par la variable continue h, si on s'intéresse au niveau de remplissage du bac. Dans ce cas on peut modéliser le système par l'équation différentielle

$$\dot{h} = d_1. - d_2.a$$

Avec d_1 le débit d'alimentation, d_2 le débit de soustraction, a est une variable booléenne qui décrit l'états de la vanne, elle prend 0 si la vanne est fermées et 1 si elle est ouverte.

Ce même système peut être décrit par une variable discrète qui indique si le bac est en état soutirage ou en état remplissage, si c'est ce qui nous intéresse et le système peut être modélisé par l'automate à état en figure 1.3.b.

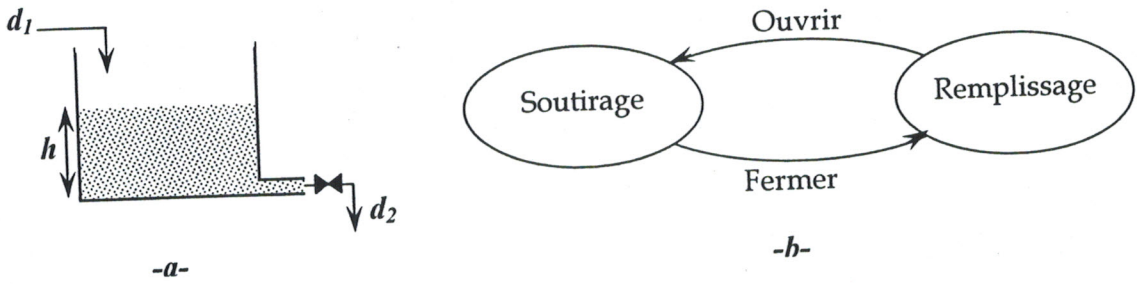


Figure 1.3. -a- exemple d'un système physique qui peut être considéré comme continu ou discret. -b- modèle automate correspondant.

1.1.3. *Les systèmes dynamiques hybrides*

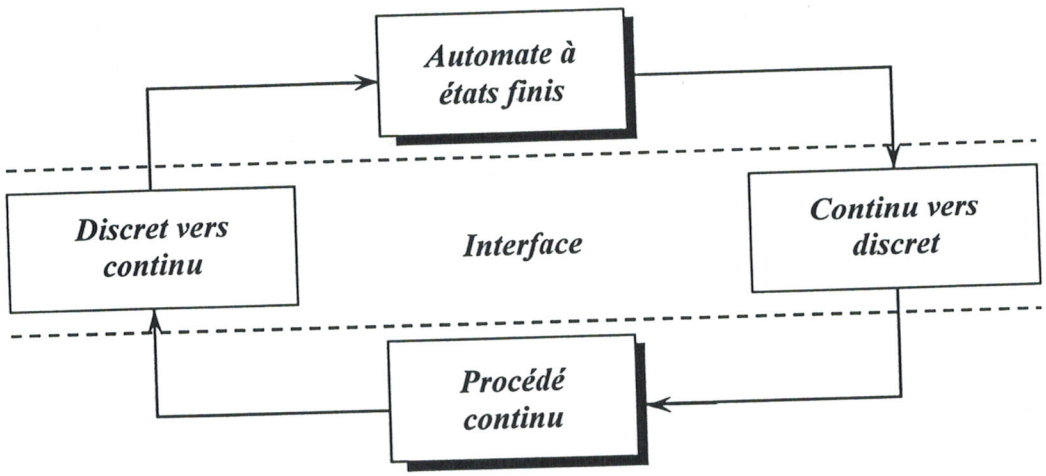


Figure 1.4. Système dynamique hybride

Dans certains cas et pour des objectifs spécifiques, il est possible de choisir toutes les variables d'état dans l'une des deux catégories, pour la description d'un système physique donnée, mais ce n'est pas toujours le cas. En effet, la plupart des systèmes physiques ne peuvent pas être classés dans l'une des deux catégories précédentes homogènes des systèmes dynamiques, et les variables d'états auxquelles on s'intéresse contiennent des variables discrètes et d'autres continues. Dans ce cas les systèmes sont dits systèmes dynamiques hybrides, ce sont des systèmes hétérogènes caractérisés par l'interaction d'une dynamique de nature discrète et une autre de nature événementielle. L'essor de ces systèmes est relativement nouveau, il date des années 1990 [AKZ98]. Flaus [Fla98] définit un SDH comme une combinaison d'un système à événement continue et un système à événements discrets, en insistant sur l'importance des interfaces qui assurent les connections entre les dynamiques continue et discrète, comme le montre la figure 1.4.

La recherche sur les systèmes dynamiques hybrides s'articule autour des trois axes complémentaires suivant :

1. La modélisation : Formaliser des modèles précis qui peuvent décrire le comportement riche et complexe des SDH.
2. L'analyse : développer des outils de simulation, de validation et de vérification des SDH.
3. La commande : effectuer la synthèse d'un contrôleur discret (ou hybride) conformément à certains objectifs de performances et de sûreté de fonctionnement du procédé hybride commandé.

1.1.4. Les systèmes dynamiques positifs

Les systèmes positifs sont une autre classe des systèmes dynamique relative aux variables d'état. Cette terminologie a été présentée pour la première fois par David Luenberger dans son livre « Introduction to dynamic systems » [Lue79]. Un système dynamique est dit positif si toutes ses variables d'état sont toujours positives. Cette positivité est souvent une conséquence de la nature du phénomène. Des exemples de tels phénomène peuvent être issus de disciplines aussi différentes telles que l'économie (les variables descriptives peuvent être des prix de marchandises, des quantités de stocks), les sciences sociales (nombres d'individus, des taux de satisfaction, de consommation), la physique (problèmes de réservoirs d'eau, systèmes d'irrigation), la chimie (réacteurs chimiques, colonnes à distiller), les télécommunications (quantité d'informations à faire transiter par un réseau). Dans ce document, nous nous intéressons aux systèmes positifs impliquant un flux de matière (continu), ou un flot de produits (discret). Des exemples de tels systèmes sont donnés dans le paragraphe suivant.

1.1.5. Exemples illustratifs

Comme mentionné précédemment, un système est qualifié d'hybride s'il implique des processus continus et des phénomènes événementiels. Par extension, nous pouvons qualifier d'hybrides des systèmes physiques dont certaines grandeurs varient très rapidement (quasi-instantanément) par rapport aux autres. Une modélisation hybride pour cette catégorie de systèmes physiques est envisageable et donne souvent de meilleurs résultats par rapport à une modélisation événementielle. Nous allons présenter ici deux exemples de systèmes hybrides, le premier est un système de bacs impliquant un flux de liquide (continu) et le second est un système manufacturier traitant un flot de produits (discret approché par une description continue).

Exemple 1.2 : Soit le système de bacs schématisé en figure 1.5. Ce système comporte deux bacs qui sont vidés en permanence (sauf dans le cas où les bacs sont vides) à un débit de 5 et 7 litres/seconde respectivement. Les bacs sont aussi alimentés à tour de rôle, avec une vanne dont le débit est 12 litres/seconde. Cette dernière à deux positions, quand elle est en position A, elle alimente le bac1 et elle

alimente le bac 2 si elle est en position B. Pour commuter entre les positions et B la vanne à besoin de 0.5 secondes. Pendant lesquelles, la vanne se comporte comme si elle est à son ancien position.

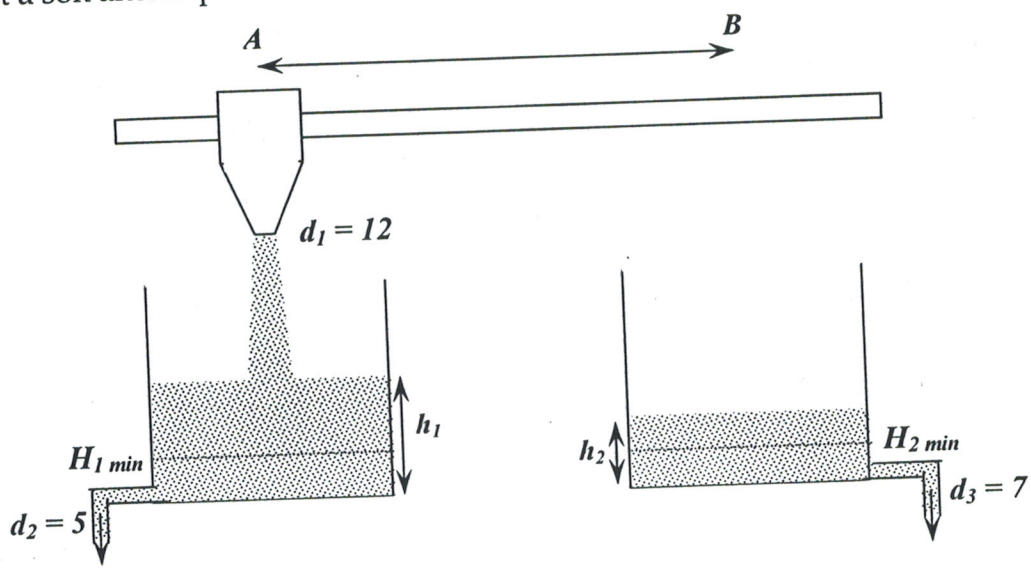


Figure 1.5 système de bacs.

Exemple 1.3 : La figure 1.6 représente un système manufacturier comportant 3 machines et 2 stocks tampons. Ce système est utilisé pour satisfaire une demande périodique, dont la période est de 20 unités de temps. Les machines 1 et 2 restent opératoires en permanence, tandis que la machine 3 peut être arrêtée pour la régulation du taux de fabrication. Les actions de mise en arrêt et en fonctionnement de la machine 3 prennent 5 u.t. Les machines ont des taux de fabrication de 10, 7, et 22 pièces/unité de temps, respectivement. Dans ce système le flux des pièces est supposé être un processus continu, tandis que l'état de la machine 3 ainsi que l'état de la demande sont des variables discrètes.

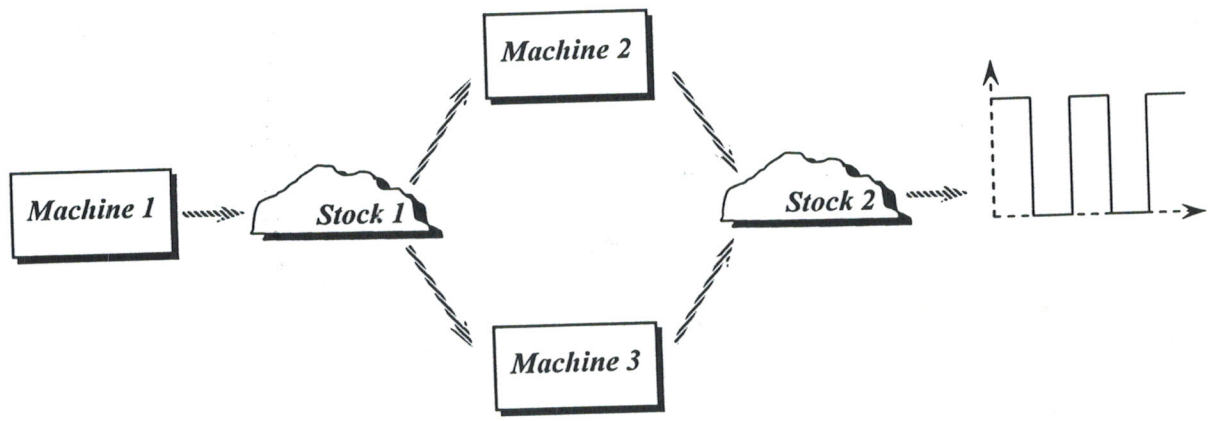


Figure 1.6 Système manufacturier

1.2. Caractéristiques des systèmes dynamiques hybrides

1.2.1. *Les phénomènes hybrides*

Branicky, Borkar et Mitters [BBM94] ont identifié les principaux phénomènes physiques pouvant être présents dans un système hybride et précisent qu'ils peuvent être autonomes ou contrôlés. Ainsi, ils définissent :

- Les *commutations* ("*switching*"). Ce phénomène est observé quand le champ vectoriel $f(x)$ change d'une façon discontinue lorsque l'état continu x atteint certains seuils, comme schématisé sur la figure I.7.a pour une seule variable d'état.
- Les *sauts* ("*jump*"). Il s'agit ici d'un saut discontinu de l'état continu x lorsqu'il atteint une région donnée de l'espace d'état. Comme le montre schématiquement la figure I.7.b pour une variable d'état.

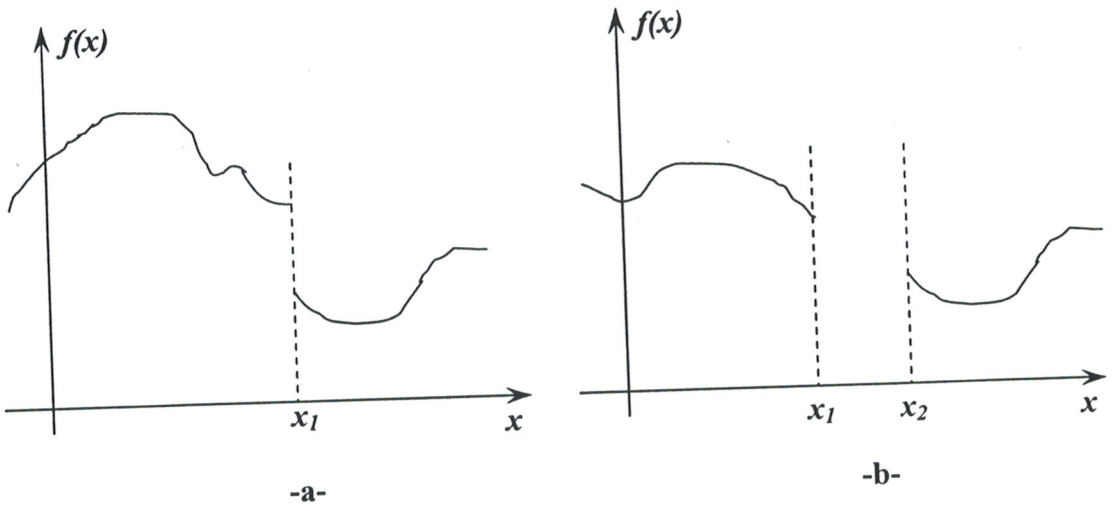


Figure I.7.a. Commutation à la valeur seuil x_1. b. Saut de la région $(x_2\text{-}x_1)$

1.2.2. *Les difficultés liées aux systèmes dynamiques hybrides*

Van der Schaft et Schumacher énumèrent dans [VS00] les principaux comportements types qu'il est possible de rencontrer en étudiant la trajectoire d'un système dynamique hybride :

- Le système arrive dans un état où il n'y a plus de trajectoires continues définies et où il n'y a pas de transitions vers un autre état. Le système est dit « bloqué » ;
- Les durées des trajectoires continues (*i.e.*, le temps pendant lequel le système évolue entre deux sauts/commutations) deviennent de plus en plus petites. Le système est dit «Zeno».
- L'évolution du système rencontre plusieurs événements simultanés. Nous avons plusieurs changements d'état au même instant.

- Le système commute indéfiniment entre deux états, il est alors appelé "*livelock*" ;
- L'ensemble des trajectoires est vaste et nous n'avons donc pas une solution unique ;
- Un état du système commute sur lui-même indéfiniment ;
- La partie continue tend vers l'infini dans un temps fini, (*par exemple* $\dot{x} = 1 + x^2$ *et* $x_0 = 0$ qui a pour solution $x(t) = \tan t$.

En fait, les problèmes rencontrés lors de l'étude des trajectoires d'un système dynamique hybride sont à la fois les problèmes rencontrés lors de l'étude d'un système continu et d'un système discret avec en plus des phénomènes qualifiés d'hybrides.

1.3. Formalismes pour les systèmes dynamiques hybrides

Dans la démarche d'intégration des deux aspects au sein d'un même modèle, trois approches ont été présentées. Elles dépendent du modèle dominant, c'est à dire du modèle à partir duquel est effectuée l'extension : approche continue, approche discrète et approche mixte sont les distinctions faites dans la littérature. Des états de l'art sur ces formalismes peuvent être trouvés dans [AKZ98], [Ant00], [LBR96] [VS00] et [Zay01].

1. *L'approche* continue : Cette approche consiste à intégrer l'aspect événementielle au sein d'un formalisme continu. C'est une extension des formalismes des systèmes continus. L'introduction des éléments de commutation dans le formalisme du Bond graph [STS93], [QBC96] en est un exemple.
2. *L'approche discrète* : Cette approche consiste à intégrer l'aspect continu au sein d'un modèle à évènements discrets. L'intégration de l'aspect continu au sein d'un modèle à évènements discrets comme les réseaux de Petri est un exemple de ce type d'approche. Le champ d'application des RdP a été étendu aux systèmes continus par l'introduction de nouvelles classes de RdP dans lesquelles la notion de marquage qui prend ses valeurs dans $\Re$ est apparue [DA87]. Cette extension a permis la définition des réseaux de Petri hybrides, intégrant un RdP Continu et un RdP discret, permettant alors de représenter explicitement l'interaction entre la partie continue et la partie discrète par l'intermédiaire d'arcs. Les RdP différentiels ont introduit par la suite les places et transitions différentielles [DK96].
3. *L'approche mixte* : Cette approche combine explicitement des modèles des parties continues et discrètes dans une même structure, l'aspect hybride étant pris en charge dans l'interface entre les deux parties. Comme exemple de tels formalismes on peut citer les automates hybrides [ACH+95].

Par la suite, nous allons nous focaliser sur les approches issues de la modélisation discrète après avoir introduit le modèle le plus général : les automates hybrides.

1.3.1. *Le modèle général de l'automate hybride*

Les automates hybrides [ACH+95] sont le formalisme de modélisation le plus général pour décrire et analyser les SDH, dans le sens où ils peuvent modéliser la plus grande variété des dynamiques hybrides. Un automate hybride est un automate à états finis étendu par un ensemble fini de variables réelles $X = \{x_1, x_2, \ldots, x_n\}$. De façon informelle on peut définir un automate hybride comme étant une combinaison d'un ensemble de variables prenant leurs valeurs dans $\Re$ et d'un graphe orienté et étiqueté A = (S, T), avec S étant l'ensemble des nœuds dits sommets dont on affecte à chacun :

1. Une *activité f* qui est une équation différentielle en fonction des variables de X.
1. Un invariant qui est une condition sur les valeurs des variables de X, L'automate hybride peut rester dans un sommet seulement si son invariant est vérifié.

T est l'ensemble des arcs dits transitions. A chaque transition on affecte :

2. Une *garde* qui est une condition sur les valeurs des variables de l'ensemble X, cette condition doit être vérifiée pour que la transition correspondante (sortie du sommet) soit franchissable.
3. Une *Affectation* dont le rôle est d'affecter de nouvelles valeurs aux variables de X après le franchissement de la transition.

L'état d'un automate hybride à tout instant t est donné par un couple $E = (q, v)$ ou q est un sommet de l'automate et v est une valuation qui donne les valeurs des variables de X à l'instant t. L'état peut changer de deux manières, soit par franchissement instantané de transition discrète, soit par l'écoulement du temps dans un même sommet. Franchir une transition change le sommet actif et les valeurs des variables réelles, tandis que l'écoulement du temps ne change que les valeurs des variables réelles selon l'activité du sommet actif.

Une exécution d'un automate hybride est une séquence fini ou infini de la forme :

$$E_0 \xrightarrow[f0]{t0} E_1 \xrightarrow[f0]{t0} E_2 \xrightarrow[f0]{t0} \cdots$$

Où les $E_i = (q_i, v_i)$ sont les états de l'automate hybride, les f_i sont les activités des sommets q_i tel que $f_i(0) = v_i$ et $f_i(t)$ satisfait l'invariant du sommet q_i quelque soit $t \in \Re$, et $0 \leq t \leq t_i$.

A titre d'illustration, considérons l'automate hybride de la figure 1.8. Cet automate décrit le comportement du système de bacs présenté en exemple 1.2, si on veut garder le niveau de liquides dans chaque réservoir aux dessus des seuils H_{1min} et H_{2min}. L('automate hybride comporte quatre sommets S_1, S_2, S_3 et S_4 et trois

variables d'état continues h_1, h_2 et x. Les deux premières modélisent les niveaux de liquide dans les deux réservoirs et la troisième est une horloge qui modélise le délai entre l'instant de décision de commutation de la vanne et sa commutation effective.

Lorsque la vanne est dans la position *A*, la hauteur du liquide dans les réservoirs varie d'une manière linéaire suivant les fonctions

$$h_1(t) = h_{10} + 7.t$$

$$h_2(t) = h_{20} - 7.t$$

De la même manière, si la position de la vanne est B, les équations deviennent :

$$h_1(t) = h_{10} - 5.t$$

$$h_2(t) = h_{20} + 5\,t$$

Avec h_{10} et h_{20} sont les hauteurs initiales du liquide dans les réservoirs. L'horloge x évolue uniformément avec le passage du temps dans les sommets S_2 et S_3, et a une dynamique nulle dans les autres sommets.

Les transitions entre sommets sont franchies si l'un des seuils est atteint.

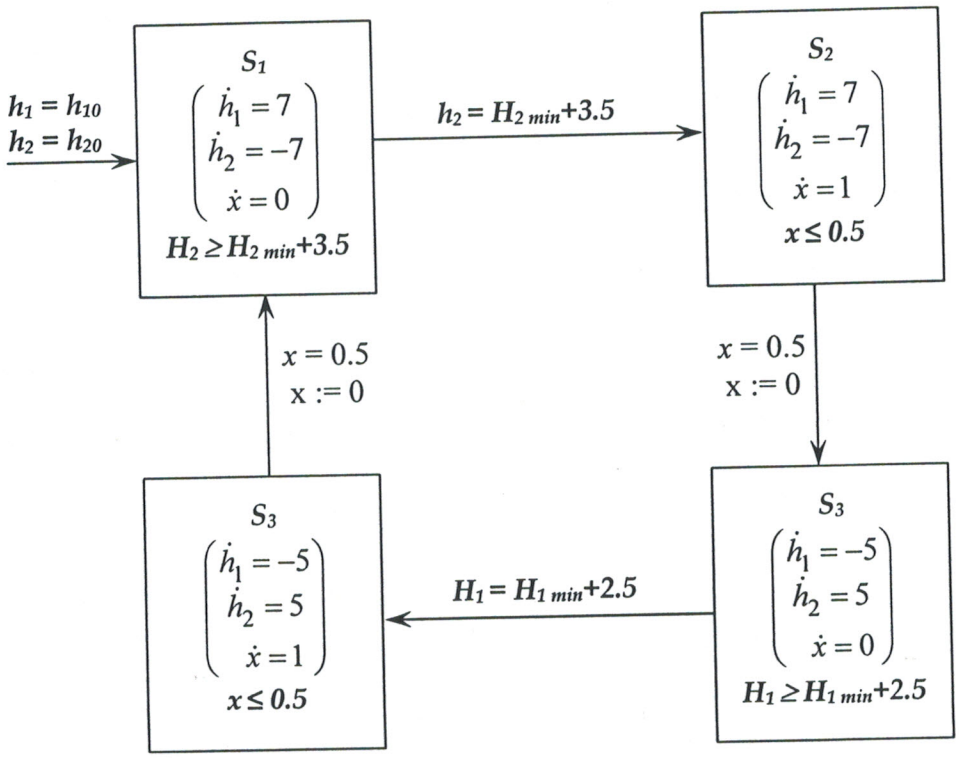


Figure 1.8 Automate hybride modèle du système de bacs

Les automates hybrides présentent l'avantage d'avoir une représentation graphique claire. Les parties continues et discrètes sont en effet bien identifiées. La clarté de ce modèle fait que plusieurs auteurs l'utilisent pour définir un système hybride général [VS00].

1.3.2. *Extensions des réseaux de Petri pour modéliser les systèmes hybrides*

De la même façon que les automates ont été étendus afin de représenter les systèmes dynamiques hybrides, plusieurs modèles construits à partir du formalisme des RdP ont été définis. Le premier pas en cette direction a été pris par David et Alla en définissant les réseaux de Petri continus [DA87]. Depuis plusieurs travaux ont été consacrés à étendre les réseaux de Petri pour pendre en charge les systèmes hybrides. Une liste de ces travaux peut être trouvé sur la page Web [1].

a. Les réseaux de Petri continus

Les RdP continus [DA87] ont été développés pour représenter des flots continus et ainsi approximer un RdP discret afin de réduire les temps de calcul. Les jetons ne sont plus considérés comme des entiers mais comme des réels. Ainsi, lors du franchissement d'une transition, une quantité infinitésimale est prélevée des places en amont et arrive dans les places en aval. Le franchissement des transitions n'est plus instantané mais fonction d'une vitesse que nous pouvons assimiler à un débit. L'équation fondamentale d'un RdP est donc remplacée par :

$$\dot{M} = W.v(t)$$

Où W est la matrice d'incidence du RdP et $v(t)$ est le vecteur des vitesse instantanée de franchissement à l'instant t.

Plusieurs modèles de RdP continus ont été développés : le *RdP continu à vitesses constantes* [DA87], le *RdP continu à vitesses variables* [DA87], où les vitesses de franchissement des transitions dépendent de l'état du marquage des places en amont, le *RdP continu asymptotique* [LeB92], où les vitesses de franchissement sont constantes par morceaux et le *RdP continu à vitesses fonction du temps* [DAD94], les vitesses sont soit une fonction continue par rapport au temps, soit une fonction continue par morceaux.

b. Extensions des réseaux de Petri à la modélisation des systèmes hybrides

Le Bail, Alla et David [LAD91], [LeB92] ont développé à partir des RdP continus les *RdP hybrides*. Ces réseaux contiennent des places et des transitions discrètes (représentées graphiquement avec un contour simple) et continues (représentées avec un contour double). Ainsi, la partie continue permet de modéliser les flux continus et la partie discrète les fonctionnements logiques (vanne ouverte/fermée). Au niveau de l'interface continu/discret et afin de garantir que le marquage des places discrètes reste entier, tout arc connectant une place discrète et une transition continue doit avoir son arc réciproque.

Le modèle de la figure 1.9 représente le système décrit en exemple 1.2. Les places P_3 et P_4 représentent respectivement les niveaux de liquides dans les bacs 1 et 2. La transition T_3, par exemple, est commandée par la place P_1 (Position de la vanne), quand le niveau de liquide dans le bac 2 atteint le niveau H_{2min} la vanne

change de position (marquage de P_2). Ceci est un exemple de commande discrète d'un processus continu.

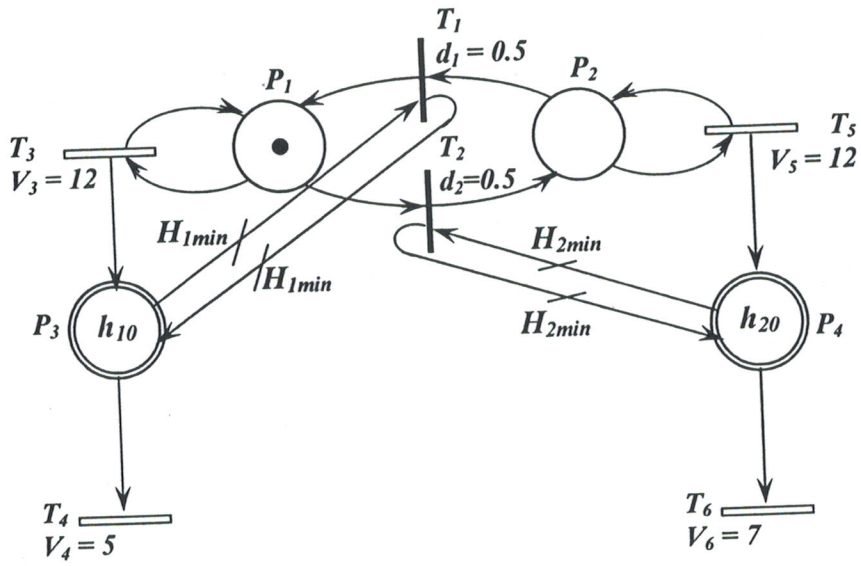


Figure 1.9 Réseau de Petri hybride, exemple du système des bacs.

Ce formalisme a, par exemple, été utilisé pour modéliser un système d'approvisionnement en eau [AD98], un système de production hydraulique d'électricité [Dav91] et un système de fabrication de transistors [LeB92]. Cependant, ces modèles ont principalement servi à évaluer les performances des procédés et n'ont donc pas été utilisés pour définir une politique de contrôle.

c. *Extension des RdP hybrides : les RdP lots*

Demongodin, Aubry et Prunet [DAP93] ont étendu les RdP hybrides afin d'introduire des retards et des accumulations, ce qui permet de modéliser les caractéristiques des lots et leurs transformations. Ceci s'effectue par l'attribution à des places (appelées places-lots) des paramètres et des lois d'évolution temporelle du système pendant la durée du marquage. D'autre part, les vitesses de transitions appelées transitions-lots sont fonction des informations liées aux places-lots en amont et en aval. Ce type de modèle est bien adapté à la modélisation de lignes de productions où il y a des convoyeurs.

d. *Les RdP différentiels*

Demongodin et Koussoulas ont défini les RdP différentiels [DK98] où deux types de places et de transitions sont définies : discrets et différentiels. Ceci permet d'introduire un marquage réel (positif, négatif ou nul). Des vitesses de franchissement et des temporisations sont d'autre part attribuées aux transitions différentielles, ce qui permet de discrétiser le temps. Il est ainsi possible de représenter un modèle discrétisé où la partie continue est du premier ordre (les variables d'état correspondent aux places différentielles). Nous avons donc des variables continues linéaires par morceaux entre chaque pas de discrétisation.

e. *Les RdP couplés à des équations différentielles algébriques*

Les RdP hybrides sont adaptés à la représentation de systèmes où la partie continue est simple, linéaire par morceaux dans les cas les plus complexes. Pour essayer de représenter des systèmes où la représentation du continu nécessite une plus grande précision, le couplage d'un RdP avec un système d'équations différentielles algébriques a été envisagé en s'inspirant des automates hybrides [And96], [Cha98] et [Val98]. Le résultat obtenu est appelé réseau prédicats-transitions-différentiels (PTD) par Champagnat [Cha98]. Les RdP prédicats-transitions font partie des réseaux de haut niveau (avec les RdP colorés) car les jetons sont individualisés c-à-d que chaque jeton contient une information propre à lui. Il est alors possible d'exprimer des conditions au niveau des transitions qui prennent en compte l'information contenue par les jetons. Dans un RdP prédicats-transitions, en plus d'un RdP classique, sont donc définis :

- un ensemble de variables,
- une application associant à chaque transition une condition sous la forme d'un prédicat utilisant les variables,
- une application associant à chaque transition une action sous la forme d'une suite d'affectations de valeurs aux variables,

Champagnat souligne que, dans un réseau prédicats-transitions, nous avons la possibilité d'introduire des conditions de franchissement (qui peuvent être assimilées à des phénomènes de seuils sur les variables) et des affectations instantanées de valeurs (qui représentent des discontinuités dans les évolutions des variables) ; ces caractéristiques sont une partie des éléments nécessaires à la manipulation de variables continues.

D'autres approches, utilisant les RdP de haut niveau existent pour la représentation des systèmes dynamiques hybrides. L'utilisation des RdP colorés permet d'introduire le caractère continu de certaines variables (via l'individualisation des jetons) [GS98]. Cependant Champagnat *et al.*, soulignent, dans [CPA+98] les limites de cette approche. En effet, des valeurs continues sont associées aux jetons mais ces valeurs ne sont modifiées que lors des tirs des transitions et donc, pour approximer au mieux le continu, il est nécessaire de multiplier les tirs (*i.e.*, obtention d'un modèle discrétisé).

f. *Comparaisons entre les extensions des réseaux de Petri à la modélisation des systèmes hybrides*

Les différentes extensions des réseaux de Petri présentées ci-dessus peuvent être synthétisées comme suit :

- Les réseaux de Petri hybrides et leurs extensions offrent une représentation graphique intéressante, puisqu'elle reste intuitive. Ils permettent la modélisation de

variables continues linéaires par morceaux. Et surtout, le modèle décrit dans le même formalisme les aspects discret et continu.

- Les approches par RdP de haut niveau (c'est à dire les RdP prédicats-transitions et colorés) offrent une alternative pour des modèles discrétisés ou échantillonnés quand le pas de discrétisation est constant. Mais il faut noter que ceci s'obtient par l'ajout de transitions qui ne servent qu'à la simulation du modèle. Ainsi, nous pouvons considérer que le réseau perd de sa clarté et de sa simplicité, caractéristiques qui sont l'un des points fort des RdP.
- Les RdP prédicats-transitions-différentiels diffèrent des autres approches. Ils sont en effet beaucoup plus proches des automates hybrides avec la possibilité de représenter des variables continues non-linéaires et d'introduire des discontinuités dans les signaux continus. Cependant, le modèle obtenu est plus complexe que les autres et sa lourdeur complique la modélisation des systèmes de grande taille.

La facilité d'utilisation et la clarté de représentation graphique sont les principaux avantages des réseaux de Petri discrets. Seuls les réseaux de Petri hybrides et leurs extensions préservent ces avantages. Cependant ils ne permettent pas de modéliser la classe générale des systèmes hybrides, mais seulement ceux qui sont à flux continu (variables d'état positifs et linéaires par morceaux).

1.3.3. Le modèle unifié de Branicky

Branicky, Borkar et Mitters [BBM94], [Bra95] ont proposé une structure de modélisation qui inclut plusieurs modèles précédemment développés, comme :

- L'approche de Antsaklis, Stiver et Lemmon [ASL93] ;
- L'approche de Back, Guckenheimer et Myers [BGM93] ;
- L'approche de Brockett [Bro93] ;
- L'approche de Nerode et Kohn [NK93] ;
- L'approche de Tavernini [Tav87] ;

Ce formalisme est basé sur une représentation du continu sous la forme d'équations différentielles

$$\dot{x} = f(x)$$

Où le champ vectoriel f dépend d'élément continus et d'autres discrets. Ce formalisme de modélisation présente l'avantage d'être générique et d'unifier de nombreuses approches de modélisation des systèmes dynamiques hybrides. Cependant, le résultat obtenu est relativement complexe et sa compréhension est peu aisée au premier abord. Ce type de modèle est par ailleurs axé sur le contrôle des systèmes.

1.3.4. Autres modèles pour les systèmes hybrides

De nombreuses autres approches ont été définies pour modéliser les systèmes dynamiques hybrides, nous citons ici quelques unes, sans entrer dans les détails :

- Des langages synchrones comme ESTEREL [BCG87], LUSTRE [CPH+87] et SIGNAL [BGG90] ont été développés avec comme objectif une représentation précise du temps. Ils permettent la représentation des équations différentielles et des équations booléennes, ce qui leur permet d'être utilisé pour la modélisation des SDH.
- Plusieurs travaux comme [ECO93] [Thé00] ont été consacrés à la modélisation des SDH par des approches orientées objet.
- Buisson et Cormerais [BC98] et Mosterman [Mos97] se servent des bond graphs qu'ils étendent au domaine de l'hybride.
- Zaytoon et ses coauteurs [ZCN+97] se servent des grafcets comme point de départ. Cette approche présente de nombreux points communs avec celles mettant en jeu des RdP.

Conclusion

Ce chapitre a été dédié à la présentation des systèmes dynamiques hybrides, des principaux phénomènes hybrides, des problèmes liés à leur modélisation et finalement des principaux formalismes de modélisation hybrides. De tous les formalismes présentés dans ce chapitre les automates hybrides permettent la modélisation de la plus grande variété de systèmes hybrides. L'existence d'outil automatisés, comme HyTech, pour leur analyse et vérification, fait que l'analyse de plusieurs autres formalismes se fait après leur traduction en automates hybrides. Ainsi, Allam et Alla [AA98] ont développé un algorithme de transformation d'un RdP hybride en automate hybride, Champagnat [Cha98] transforme le RdP prédicats-transitions-différentiel en automates hybrides pour sa simulation, Demongodin et Rouibia traduisent les RdP lots en automates hybrides pour leurs analyse. La traduction du modèle RdP en modèle automate en vue de son analyse n'est pas qu'une technique utilisée que pour les modèles hybrides, cette technique à été aussi utilisée pour les RdP temporels comme nous le montrerons dans le chapitre suivant.

Modélisation et Analyse des Systèmes Dynamiques Hybrides

Ce chapitre est consacré à deux outils de modélisation et d'analyse des systèmes dynamiques hybrides ; à savoir les automates hybrides est les réseaux de Petri hybrides. Les premiers sont le modèle le plus général pour la représentation du comportement hybride. Un automate hybride apparaît comme une association d'un automate à état fini pilotant un ensemble d'équations différentielles. Les Réseaux de Petri hybrides étendent les réseaux de Petri classiques dans le but de modéliser la classe des systèmes dynamiques hybrides positifs. Ce modèle hérite de la puissance de description des réseaux de Petri classiques.

2.1 Les automates hybrides

2.1.1. *Rappels sur les automates à états finis déterministes*

Les automates à états finis (AEF) constituent le modèle de base pour la représentation des Systèmes à événements discrets [BL99]. De façon générale, un automate est une machine abstraite qui a des entrées et des sorties discrètes ; et qui réagit à une modification de ses entrées en changeant ses sorties.

Formellement un AEF déterministe peut être défini de la façon suivante :

Définition 2.1 (automate à états finis) : Un automate déterministe A est un quintuple $A = (Q, \Sigma, T, q_0, F)$ où :

- Q est un ensemble fini de sommets ;
- Σ est un ensemble fini de symboles d'entrée, appelé alphabet d'entrée de l'automate ;
- T est un ensemble de transitions entre états. Une transition est un triplet $t_i = (q, \sigma, q')$, ou q est un sommet source, σ est un événement et q' est un sommet but. La transition ti = (q, σ, q') traduit le passage du système du sommet q vers le sommet q' suite à l'occurrence de l'événement σ ;
- q_0 est l'état initial de l'automate ;
- F est l'ensemble des états finals, où $F \subseteq Q$;

❑

Exemple 2.1 : La figure 2.1 modélise une machine simple qui peut prendre trois états : A (arrêt), M (marche) et P (panne). Ainsi $Q = \{A, M, P\}$. La machine est initialement en arrêt ce qui est repéré par une flèche entrante. On considère l'état M (marche) comme l'état sortie, il est donc représenté avec un cercle double. La transition d'un état à un autre se fait par occurrence de l'un des quatre événements début du travail (α), fin du travail (β), panne de la machine (γ) et réparation de la machine (λ).

- $Q = \{A, M, P\}$;
- $\Sigma = \{\alpha, \beta, \gamma, \lambda\}$;
- $\delta(A, \alpha) = M$; $\delta(M, \beta) = A$; $\delta(M, \gamma) = P$; $\delta(P, \lambda) = A$;
- $q0 = A$;
- $F = \{M\}$;

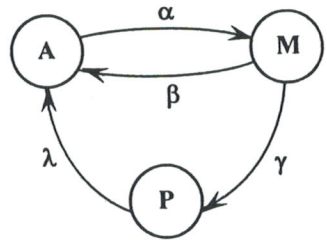


Figure 2.1 Automate à états finis modélisant une machine simple

❑

Le comportement global d'un système à événements discret est décrit par l'ensemble des séquences d'événements qui peuvent être exécutés en parcourant l'automate à partir de son état initial et jusqu'aux états finaux. Cet ensemble correspond à un langage issu de l'alphabet Σ, exprimé par une expression régulière permettant d'agréger les séquences d'événements répétitives. Ainsi l'expression régulière $\alpha(\varepsilon + (\beta\alpha)^* + (\gamma\lambda\alpha)^*)$ correspond à l'automate de la figure 2.1. Le symbole ε correspond à la séquence de longueur nulle (événement vide) et $(\beta\alpha)^* + (\gamma\lambda\alpha)^*$ indique que chacune des séquences βα et γλα peuvent être exécutées autant de fois que l'on veut. Une exécution possible de cet l'automate est donnée dans la figure 2.2.

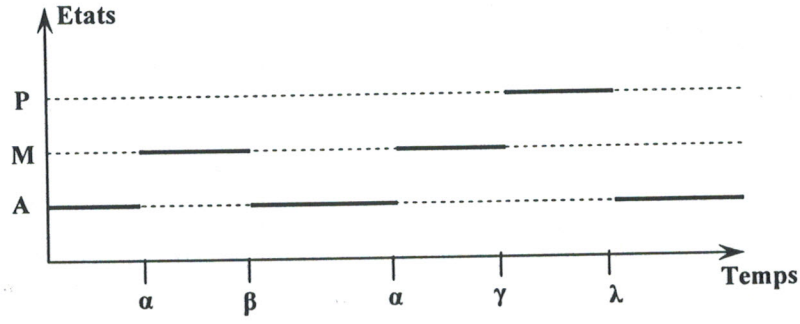


Figure 2.2 Trajectoire possible pour l'évolution de l'automate de la figure 2.1.

La modélisation d'un système comportant deux machines identiques et fonctionnant indépendamment l'une de l'autre, nécessite un automate avec 9 états et 24 transitions. Cet automate est représenté dans la figure 2.3, chaque état de cet automate est un couple dont la première composante indique l'état de la première machine et la deuxième composante indique l'état de la deuxième machine. A titre d'exemple l'état (A, A') indique que les deux machines sont en arrêt. Les transitions entre états se font par occurrence des événements α, β, λ, γ de la première machine et

α', β', λ', γ' de la deuxième machine. Cet automate est la composition asynchrone de deux automates identiques à celui de la figure 2.1.

Un automate modélisant n machines identiques à celle modélisées par l'automate de la figure 2.1 comporte 3^n états et $(4n.3^{n-1})$ transitions, ainsi pour trois machines on a 27 états et 108 transitions, et 81 états et 432 transitions pour 4 machines. De tels automates sont difficiles à élaborer et à analyser manuellement, c'est pourquoi la modélisation avec un seul automate n'est pas pratique pour des systèmes ayant un grand nombre de composantes ; leur nombre d'états croît exponentiellement avec le nombre de leurs composantes.

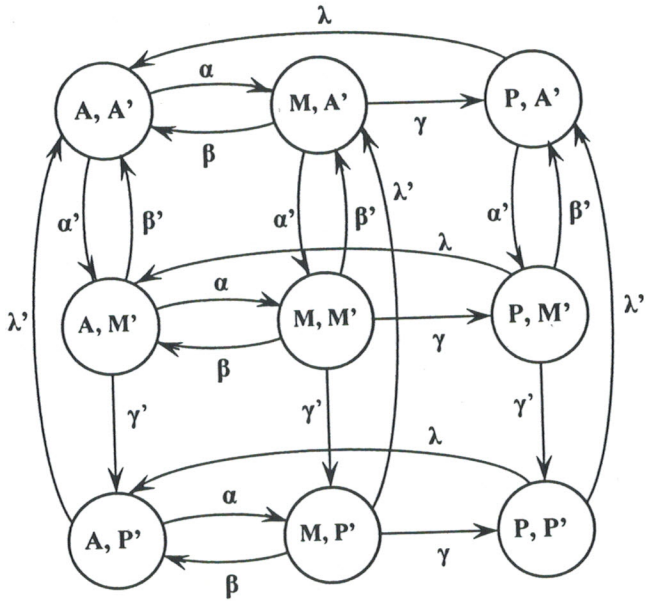


Figure 2.3. Automate à états finis modélisant deux machines indépendantes

2.1.2. *Les automates hybrides*

Les automates hybrides ont été introduits par Allur *et al.*, dans [ACH+95] comme une extension des automates à états finis. Il s'agit d'étendre par un ensemble de variables réelles $X = \{x_1, x_2, \ldots, x_n\}$ les AEF. A chaque sommet q d'un automate hybride on associe une fonction d'évolution f_q, encore dite activité, pour chaque variable continue de la forme $f(\dot{x}(t), x(t)) = 0$, et un prédicat sur la valeur des variables appelée invariant du sommet. Le système peut séjourner dans un sommet tant que les valeurs des horloges vérifient l'invariant associé. Le franchissement d'une transition de l'automate hybride est instantané, à chaque transition on associe une condition de franchissement, appelée garde, et une affectation. Une transition ne peut être franchie que si sa garde est vérifiée par les valeurs des variables. Les gardes modélisent les contraintes imposées au comportement du système, et l'affectation associée à une transition désigne les variables qui effectuent un saut suite au franchissement de la transition.

Un automate hybride est formellement défini comme suit :

Définition 2.2 (Automate hybride) : Un automate hybride de dimension n est un 7-uplet $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$, où :

- Q est l'ensemble fini de sommets ;
- Σ est l'ensemble fini des symboles ;
- $q_0 \in Q$ est le sommet initial ;
- X est un ensemble de n variables réelles ;
- Inv est une application qui associe à chaque sommet $q_i \in Q$ un invariant de sommet $Inv(q_i)$ qui est une contrainte temporelle portant sur les valeurs des horloges. Tant que l'invariant du sommet est vérifié par les valeurs des horloges, il est possible de rester dans ce sommet ;
- T est un ensemble de transitions (arcs). Une transition est un quintuple de la forme $T = (q, \sigma, g, A, q')$, où :
  - q est le sommet source de la transition ;
  - $\sigma \in \Sigma$ est un symbole associé à un événement qui peut être reçu ou généré lors du franchissement de T ;
  - g est la condition que doivent satisfaire les valeurs des horloges de l'automate hybride pour pouvoir franchir la transition T c'est la garde de cette transition ;
  - A est l'ensemble des variables à réinitialiser lorsque la transition est franchie. Cet ensemble est dit l'affectation associée à la transition. Les initialisations spécifiées par A correspondent à des fonctions calculant la nouvelle valeur de l'état a partir de sa valeur avant le franchissement, une affectation est de la forme $x := A(x)$
  - q' est le sommet but de la transition ;
- F est une application qui associe un comportement continu f_q, appelé aussi activité, a chaque sommet q. Ce comportement est le plus souvent exprimé par une équation différentielle de la forme $f(\dot{x}(t), x(t))=0$;

❑

Exemple 2.2 : L'automate hybride de la figure 2.4 modélise, d'une manière plus riche, la machine simple présentée précédemment dans l'exemple 2.1. Deux variables d'état continues sont utilisées dans le modèle hybride pour spécifier que la durée opératoire de la machine est de 3 u.t. (variable x). Le vieillissement de la machine est mesuré par la variable y, une panne ne peut surgir avant que la variable y atteint la valeur 7 depuis la dernière action de réparation. Le vieillissement de la machine est exponentiel si la machine est en marche et est linéaire lorsque la machine est à l'arrêt.

❑

Notons qu'une transition dont la garde est vraie est toujours franchissable, mais pas nécessairement franchie, lorsque son sommet source est actif. De même, un sommet peut rester inconditionnellement actif lorsque son invariant est vrai.

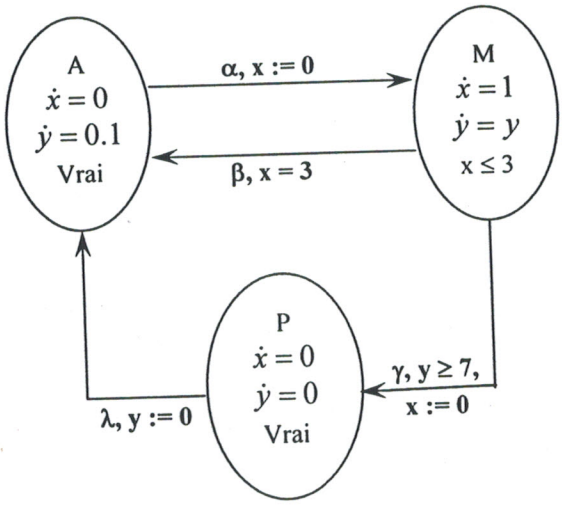


Figure 2.4. Automate hybride modélisant une machine simple.

Définition 2.3 (Valuation) : Une valuation des variables est une fonction $v : X \rightarrow \Re$ qui affecte à chaque variable sa valeur à tout instant.

❑

L'ensemble des valuations des variables $x \in X$ est noté V_X. La notation $v \parallel C$ exprime le fait que la valuation $v \in V_X$ satisfait la contrainte C.

Définition 2.4 (état d'un automate hybride) : L'état d'un automate hybride est défini par le couple $E_i = (q_i, v_i)$ où q_i est le sommet actif et v_i est une valuation qui vérifie l'invariant du sommet $Inv(q_i)$.

❑

A partir d'un état, le système peut évoluer, soit en franchissant une transition discrète qui change le sommet actif et réinitialise certaines variables, soit par la progression du temps dans le sommet courant, ce qui entraîne un changement permanent de l'état continu conformément à l'activité de ce sommet.

Une trajectoire d'un automate hybride est définie comme suit.

Définition 2.5 (trajectoire d'un automate hybride) : Une trajectoire d'un automate hybride A_H est une séquence finie ou infinie de la forme :

$$E_0 \rightarrow \underset{f0}{\overset{t0}{}} E_1 \rightarrow \underset{f1}{\overset{t1}{}} E_2 \rightarrow \underset{f2}{\overset{t2}{}} \cdots$$

Où les variables $E_i = (q_i, v_i)$ sont des états de A_H, les fonctions f_i sont les activités dans les sommets q_i et les t_i sont les temps de séjour dans q_i de tel sorte que :

- $f_i(0) = v_i$;
- $f_i(t) \parallel Inv(q_i) \qquad \forall\, t \in \Re \,/\, 0 \le t \le t_i$;
- $\forall\, i, \exists\, T = (q_i, \sigma, g, A, q_{i+1}) \in \mathbf{T}$ tel que :
  - $f_i(t_i) \parallel g$;
  - $f_{i+1}(0) = A(f_i(t_i))$

❑

La modélisation des systèmes dynamiques complexes se fait par la composition synchrone de plusieurs automates hybrides simples. Si l'on considère deux automates et un événement qui appartient à l'ensemble des événements de chacun d'eux, une transition étiquetée par cet événement ne peut être franchie dans l'un des automates que s'il existe une transition étiquetée par ce même événement franchissable dans l'autre automate. Les deux transitions sont alors franchies simultanément.

Toutes les analyses sur les automates hybrides dépendent du problème d'atteignabilité qui consiste à trouver l'espace des états atteignable par l'automate hybride A_H à partir d'un état initial. Ce problème dépend à la fois des gardes et des affectations et est prouvé de n'être décidable que pour une classe restreinte des automates hybrides, c à d qu'il n'est pas possible de trouver un algorithme qui réalise le calcul de l'espace atteignable et dont la convergence est assurée.

2.1.3. *Les automates hybrides linéaires*

Le formalisme automate hybride est particulièrement intéressant quand les activités, les invariants, les gardes et les affectations sont des expressions linéaires sur l'ensemble X des variables. Le terme linéaire dans ce contexte a le sens suivant :

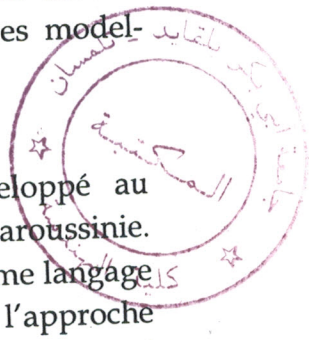
- Les activités des sommets sont des équations différentielles de la forme $\dot{x} = k$, avec k une constante rationnelle.
- Les invariants de tous les sommets sont définis par des conjonctions de prédicats (égalités ou inégalités) linéaires par rapport aux variables d'état réelles, pour lesquelles les coefficients sont des nombre rationnels.
- Les conditions des gardes des transitions sont définies par la conjonction de prédicats linéaires à coefficient rationnels sur les variables d'état continues.
- Les affectations associées aux transitions sont des fonctions linéaires des variables d'état continues.

L'intérêt de ces automates est que toutes les régions de l'espace d'état continu qu'ils définissent (invariants, gardes,...) sont des régions linéaires convexes, et que les images de régions linéaires par leurs fonctions d'évolutions continues et discrètes sont également des régions linéaires.

Une des classes les plus intéressantes des automates hybrides linéaires est la classe des automates temporisés [AD94]. Dans un automate temporisé les variables d'état continues croissent uniformément avec le passage du temps ($\dot{x} = 1$), et sont dites horloges. De plus les affectations associées aux transitions peuvent soit remettre à zéro une horloge, soit lui laisser sa valeur avant le franchissement. Le calcul de l'espace des états atteignables est prouvé être décidable pour la classe des automates temporisés.

2.1.4. *Outils automatiques d'analyse des automates hybrides*

Plusieurs outils d'analyse appelés model-checkers ont été développés pour la vérification automatique des propriétés des automates hybrides. Un système est modélisé par un ensemble d'automates hybrides se synchronisant. Ces model-checkers contiennent des procédures d'analyse d'atteignabilité. Parmi ces model-checkers nous citons :



- ***CMC***

L'outil CMC (*Compositional Model-Checking*) [CMS] a été développé au Laboratoire Spécification et Vérification à l'ENS de Cachan par François Laroussinie. Cet outil vise à vérifier des réseaux d'automates temporisés. Il utilise comme langage de spécification la logique Lv. L'algorithme qui est implémenté est l'approche compositionnelle. Des stratégies de simplifications de formules ont été aussi étudiées. Ces travaux se trouvent dans les articles [LL95], [KLL+97] et [LL98].

- ***UPPAAL***

UPPAAL [UPP] a été développé par Wang Yi, Kim G. Larsen et Paul Pettersson dans les Universités d'Uppsala en Suède et d'Aalborg au Danemark. Ce model-checker est assez facile d'utilisation puisqu'il intègre une interface graphique très conviviale et a montré son efficacité au travers plusieurs études de cas [HLS99], [LPY01], [BGK+02]. Il permet de modéliser des automates temporisés communiquant par synchronisation et de les étudier. La logique temporelle utilisée est un fragment de TCTL, ce qui ne permet que la vérification de propriétés d'accessibilité. Le point fort d'UPPAAL est le temps de calcul rapide car les algorithmes sont optimisés. De nombreux articles décrivent l'outil, par exemple [BLL+95], [LLPY97].

- ***KRONOS***

KRONOS [KRO] est également assez abouti, il a été développé au laboratoire VERIMAG de l'université de Grenoble par Conrado Daws, Alfredo Olivero, Stavros Tripakis et Sergio Yovine. Plusieurs études de systèmes et protocoles ont été réalisées à l'aide de Kronos [TY98] et [NY00]. Ce model-checker modélise des réseaux d'automates synchronisés et utilise la logique TCTL. Un automate temporisé est exprimé sous une forme textuelle. Le point fort de Kronos est l'expressivité de sa logique : Kronos implante un algorithme de model-checking pour la logique temporelle TCTL. On peut ainsi vérifier des propriétés de sûreté comme de vivacité.

- ***HyTech***

L'outil HYTECH (HYbrid TECHnology Tool) [HyT] a été développé à l'université de Berkeley par Tom Henzinger, Pei-Hsin Ho et Howard Wong-Toi. Cet outil vise à vérifier des réseaux d'automates hybrides très généraux. Il permet même d'analyser des systèmes paramétrés. Il vérifie principalement des propriétés d'accessibilité et de sûreté. La représentation symbolique qui est utilisée dans HyTech est celle des polyèdres convexes, c'est-à-dire des intersections d'hyperplans. Les algorithmes implémentés sont des algorithmes de calculs de Post et Pré. Le guide

d'utilisation de HyTech [HHWT95b] est disponible. Ainsi qu'une description de l'outil peut aussi être trouver dans [HHWT95a, HHWT97].

2.2 Les réseaux de Petri hybrides

Si les automates sont le modèle le plus classique de modélisation des systèmes à événements discrets, les réseaux de Petri ont connu depuis leur invention en 1962 par Carl Adams Petri [Pet62] un réel succès en raison de leur simplicité mathématique, des avantages de leur représentation graphique et de leur compacité. C'est l'un des outils les plus populaires pour la modélisation de systèmes à événements discrets et ses domaines d'applications sont très vastes. Une littérature très abondante existe sur les réseaux de Petri, leurs fondements théoriques et leurs applications pratiques. Nous citons cette liste non exhaustives des articles et ouvrages les plus importants traitant les réseaux de Petri et leurs propriétés ; G.W. Brams [Bra82], [Bra82], T. Murata [Mur89], R. David et H. Alla [DA92], J.L. Peterson [Pet81].

2.2.1. *Rappels sur les réseaux de Petri*

Quoique les réseaux de Petri discret et autonomes soient un outil de modélisation très connu, nous allons présenter ici les définitions les plus importantes pour unifier les notations.

D'une manière informelle, un RdP est un graphe biparti, c'est à dire avec deux types de sommets, les places (représentées par des cercles) et les transitions (représentées par des barres), des arcs permettent de relier une sorte de sommet à une autre. Un poids (nombre entier strictement positif) est affecté à chaque arc, ce poids vaut 1 quand ce n'est pas précisé. Le RdP est dit ordinaire si les valeurs de tous ses poids valent 1, et il est dit généralisé dans le cas contraire. L'ensemble des places ainsi que l'ensemble des transitions sont finis et non vides. Chaque place contient un nombre entier (qui peut être nul) de jetons ou marques, c'est le mouvement de ces jetons entre les places qui décrit la dynamique du système. Le marquage d'un RdP est un vecteur de dimension n égale au nombre de places et dont les composantes sont des entiers positifs ou nuls. La $i^{ième}$ composante indique le nombre de jetons dans la $i^{ième}$ place.

Nous illustrons cela par l'exemple suivant :

Exemple 2.3 : La figure 2.5.a représente une station de travail comportant une machine M, précédée d'un stock S de capacité 3 dans lequel les pièces attendent la disponibilité de la machine si nécessaire.

Le RdP de la figure 2.5.b décrit ce système. Initialement la machine est à l'arrêt et le stock est vide. Le RdP comporte cinq places P_1, P_2, P_3, P_4 et P_5, et trois transitions T_1, T_2 et T_3. Le nombre de jetons dans la place P_1 modélise le nombre de pièces dans le stock S. Et par complémentarité, le nombre de jetons dans la place P_2 modélise le

nombre de places disponibles dans le stock S. Les places P_3 et P_4 sont aussi complémentaires, la première indique que la machine M est occupée tandis que la deuxième indique qu'elle est disponible. La place P_5 modélise le fait qu'une seule pièce peut accéder à la machine à la fois.

Le nombre de jetons contenus dans une place P_i est dit marquage de la place P_i et est noté m_i, dans notre exemple, on a initialement $m_1 = 0$, $m_2 = 3$, $m_3 = 0$, $m_4 = 1$, $m_5 = 1$. Ainsi le marquage initial de ce RdP est donné par le vecteur : $M_0 = [0, 3, 0, 1, 1]^T$.

❑

D'une manière plus formelle nous avons les définitions suivantes :

Définition 2.6 (Réseau de Petri) : Un RdP marqué est un quadruplet $PN = (P, T, Pré, Post, M_0)$ telle que :

- P est un ensemble fini et non vide de places ;
- T est un ensemble fini et non vide de transitions. Les ensembles P et T sont disjoints $P \cap T = \varnothing$;
- *Pré* est l'application d'incidence avant, telle que :

  Pré : $(P \times T) \rightarrow \aleph$

  $(P_i, T_j) \mid \rightarrow$ Pré (P_i, T_j) = Poids de l'arc reliant la place P_i à la transition T_j ;
- *Post* est l'application d'incidence arrière, telle que :

  Post : $(P \times T) \rightarrow \aleph$

  $(P_i, T_j) \mid \rightarrow$ Post (P_i, T_j) = Poids de l'arc reliant la transition T_j à la place P_i ;
- M_0 : $P \rightarrow \aleph$

  $P_i \mid \rightarrow M_0(P_i)$ est le marquage initial de la place P_i.

❑

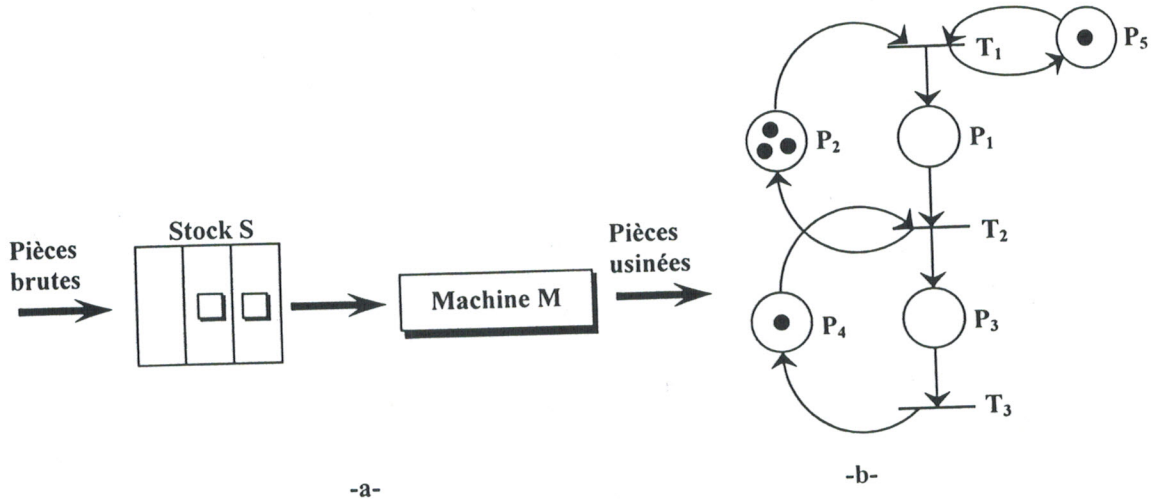


Figure 2.5.-a- Station comportant une machine et son stock d'entrée.
-b- RdP modelisant le comportement de la station.

Dans la suite du document, les notations suivantes seront utilisées :

i. ${}^{o}T_j$ (resp. ${}^{o}P_i$) représente l'ensemble des Places (resp. transitions) d'entrée de la transition T_j (resp. place P_i).

ii. T_j^{o} (resp. P_i^{o}) représente l'ensemble des Places (resp. transitions) de sortie de la Transition T_j (resp. Place P_i).

$${}^{o}T_j = \{ P_i \in P \,/\, Pré(Pi, Tj) > 0\} ;$$

$$T_j^{o} = \{ P_i \in P \,/\, Post(Pi, Tj) > 0\} ;$$

$${}^{o}Pi = \{ T_j \in T \,/\, Post(Pi, Tj) > 0\} ;$$

$$Pi^{o} = \{ T_j \in T \,/\, Pré(Pi, Tj) > 0\} ;$$

Les transitions dans un RdP modélisent les événements dont l'occurrence change l'état du système, et chaque état est modélisé par un marquage particulier du RdP. Ce marquage change chaque fois qu'une transition est franchie. Le franchissement d'une transition est conditionné par sa validation. Les définitions de la validation, du franchissement d'une transition, d'une séquence de franchissement et de conflit entre deux transition sont comme suit :

Définition 2.7 (Validation et q-validation d'une transition) : Une transition T_j est dite validée par le marquage M, ce qui est noté M $[T_j\rangle$ si le marquage M satisfait :

$$\forall P_i \in {}^{o}T_j , \frac{m_i}{Pré(P_i,T_j)} > 0$$

T_j est dite q-validée par le marquage M, ce qui signifie qu'elle a la possibilité d'être franchie β fois d'un seul coup, $(\beta \leq q)$, si :

$$\forall P_i \in {}^{o}T_j , \min \left(\frac{m_i}{Pré(P_i,T_j)}\right) = q$$

q est un entier positif, il est dit *degré de validation* de la transition T_j.

❑

Définition 2.8 (Franchissement d'une transition) : Soit M un marquage d'un RdP PN et T_j une transition validée par le marquage M. Franchir la transition T_j consiste à :

- Retirer Pré (P_i, T_j) jetons de toute place $P_i \in {}^{o}T_j$.
- Ajouter Post (P_i, T_j) jetons à toute place $P_i \in T_j{}^{o}$.

Le franchissement de la transition T_j mène le RdP du marquage M au marquage M', ce qui est noté :

$$M\,[T_j\rangle\, M'$$

Le marquage M' est donné par :

$$\forall P_i \in P : M'(P_i) = M(P_i) - Pre\,(P_i, T_j) + Post\,(P_i, T_j)$$

❑

Définition 2.9 (Séquence franchissable) : Soit M_I un marquage d'un RdP et $S = T_I T_{II} T_{III} \ldots T_N$ une séquence finie de N transitions. S est dite franchissable s'il existe une suite de marquage $M_{II}\ M_{III}\ M_{IV} \ldots M_{N+1}$.telle que :

$$\forall\ i \in \{I, II, III, \ldots, N\} : M_i\ [T_i\ \rangle\ M_{i+1}$$

Ce qui est noté :

$$M_I\ [S\ \rangle\ M_{N+1}$$

❑

Définition 2.10 (conflit entre transition) : Soit PN un RdP et T_j et T_k deux transitions de PN. T_j et T_k sont en conflit structurelle s'il existe au moins une place Pi qui est place d'entrée des deux transitions.

T_j et T_k sont en conflit structurelle si $\exists\ P_i \in {}^oT_j \cap {}^oT_k$.

Ce conflit devient effectif par un marquage M si les deux transition T_j et T_k sont validée par M et qu'il existe P_i partagé entre les deux transitions n'ayant pas assez de marquages pour les franchir toutes les deux.

T_j et T_k sont en conflit structurelle si :

$$M\ [T_j\ \rangle \text{ et } M\ [T_k\ \rangle \text{ et } \exists\ P_i \in {}^oT_j \cap {}^oT_k \text{ tel que } M(P_i) < \text{Pré}\ (P_i, T_j) + \text{Pré}\ (P_i, T_k)$$

❑

Un marquage effectif un nom déterminisme consternant les transitions franchies dans un RdP. Ce non déterminisme peut être éliminé en choisissant une politique de résolution de conflit appropriée.

Dans le RdP de la figure 2.5.b, seule la transition T_1 est validée par le marquage initial. Le franchissement de cette transition mène Le RdP au marquage $M = [\ 1\ 2\ 0\ 1\ 1\ ]^T$. A partir de ce marquage T_1 et T_2 sont validées et leurs franchissements conduisent le modèle à d'autres marquages. L'ensemble des marquages atteignable par le RdP à partir du marquage initial, est noté $M_0\ [\rangle$.

Définition 2.11 (Ensemble des marquages accessibles) : Soit un RdP marqué et M_0 son marquage initial. L'ensemble des marquages accessibles par PN à partir de M_0 est l'ensemble des marquages tel qu'il existe une séquence franchissable y menant depuis M_0.

$$M_0\ [\rangle = \{M\ /\ \exists\ S : M_0\ [S\rangle M\ \}$$

❑

Cet ensemble est généralement représenté par un graphe, dit graphe des marquages accessibles du RdP dont les nœuds sont les marquages et les arcs sont étiquetés par les transitions faisant passer d'un marquage à un autre. La figure 2.6 représente le graphe des marquages accessibles du RdP de la figure 2.5.b.

L'une des méthodes les plus importantes pour l'étude et l'analyse des propriétés des RdP, est l'utilisation du graphe des marquages accessibles. Cet outil permet de générer la totalité de états atteignables à partir de l'état initial, ainsi que les séquences de franchissement nécessaires pour atteindre chaque état. Ce graphe peut très rapidement exploser en fonction du nombre de jetons mise en jeux, ce qui réduit beaucoup le nombre de propriétés pouvant être pratiquement étudiée, et cela constitue la limitation majeure dans l'utilisation des RdP.

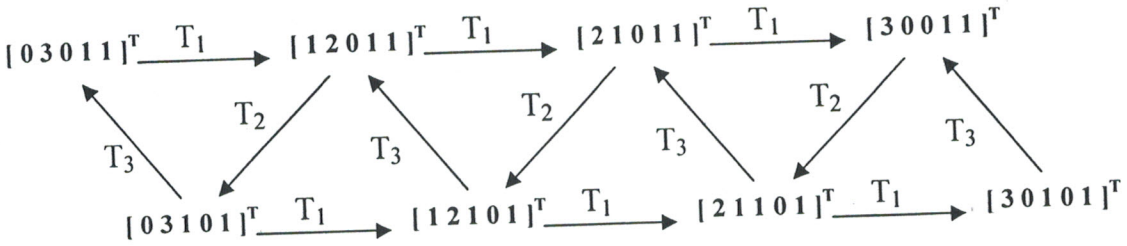


Figure 2.6. Graphe des marquages accessibles du RdP de la figure 2.1.b.

Comme nous l'avons déjà souligné, dans un RdP tout franchissement d'une transition entraîne une modification du marquage. Cette modification est exprimée par la matrice d'incidence du RdP qui est définie comme suit :

Définition 2.12 (Matrice d'incidence d'un RdP) **:** La matrice d'incidence W d'un RdP est une application de P x T dans **Z** définie par :

$$\forall P_i \in P, \forall T_j \in T, W(P_i, T_j) = Post(P_i, T_j) - Pré(P_i, T_j).$$

❑

Le RdP de la figure 2.5.b a la matrice d'incidence suivante :

$$W = \begin{array}{c} \begin{matrix} T_1 & T_2 & T_3 \end{matrix} \\ \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \begin{matrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \end{matrix}$$

Une colonne de la matrice W correspond à la modification du marquage apportée par le franchissement de la transition correspondante à cette colonne. Dans la matrice W ci-dessus, la deuxième colonne indique que le franchissement de la transition T_2 consiste à retirer un jeton des places P_1 et P_4, et à ajouter un jeton aux places P_2 et P_3.

Chaque séquence de franchissement S est caractérisée par un vecteur noté $\overline{S}$. C'est un vecteur de dimension m (nombre de transitions du RdP) dont la $i^{ème}$ composante représente le nombre d'occurrences de la $i^{ème}$ transition dans la séquence S. A titre d'exemple, le vecteur caractéristique de la séquence $S = T_1T_1T_2$ dans le RdP de la figure 2.1.b est $\overline{S} = [2\ 1\ 0]^T$.

Le marquage M obtenu après le franchissement d'une séquence S est donné par l'*équation d'état* suivante :

$$\mathbf{M} = \mathbf{M_0} + \mathbf{W}\ \overline{S}$$

Où : M_0 est le marquage initial.

W est la matrice d'incidence du RdP.

$\overline{S}$ est le vecteur caractéristique de la séquence S.

M est le marquage atteint en franchissant S à partir de M_0.

Pour notre exemple, le marquage M atteint à partir de M_0 après le franchissement de la séquence $S = T_1\ T_1\ T_2$ est :

$$M = \begin{bmatrix}0\\3\\0\\1\\1\end{bmatrix} + \begin{bmatrix}1 & -1 & 0\\-1 & 1 & 0\\0 & 1 & -1\\0 & -1 & 1\\0 & 0 & 0\end{bmatrix}\begin{bmatrix}2\\1\\0\end{bmatrix} = \begin{bmatrix}1\\2\\1\\0\\1\end{bmatrix}$$

2.2.2. *Modélisation du temps dans les réseaux de Petri*

La prise en compte explicite du temps est un des sujets les plus importants qu'a traité l'informatique des trois dernières décennies. Plusieurs extensions temporisées des RdP ont été proposées. En effet le temps peut être associé à chacun des éléments d'un RdP : les places [CR83], [Kha97], [Sif79], les transitions [HV87], [Mer74], [RH80], [Ram74], [Sta78], les arcs [Wal83], ou sur plusieurs de ces éléments en même temps, [Cer99]. Dans [Boy01] on trouve des présentations complètes de différentes extensions temporisées des RdP ainsi que des comparaisons entre ces modèles. La totalité de ces modèles peuvent être scindés en deux familles, les RdP temporisés d'un coté, et qui trouvent leur origine dans le travail de Ramchandani [Ram74], et les RdP temporels de l'autre et dont l'origine et la thèse de Merlin [Mer74]. De façon informelle, les RdP temporisés utilisent la notion de *durée* fixe à opposer à la notion de *délai* de franchissement pour les RdP temporels.

Dans cette section nous allons présenter deux des modèles RdP intégrant explicitement le temps, les plus anciens et les plus expressifs ; il s'agit des RdP T-temporisés et T-temporels.

2.2.2.1. *Les réseaux de Petri T-temporisés*

On distingue deux types de RdP temporisés à savoir : les RdP T-temporisés, ou une temporisation est associée à chaque transition ; et les RdP P-temporisés ou les temporisations sont associées aux places. Sifakis a montré dans [Sif77] l'équivalence de ces deux modèles, de plus la traduction d'un modèle en l'autre est très simple,

certains auteurs les englobent sous le nom commun "*timed place-transition nets*". C'est pourquoi nous nous contentons de présenter ici le modèle RdP T-temporisé.

Les RdP T-temporisés développés par Ramchandani dans [Ram74] font associer à chaque transition T_j une durée d_j, qui modélise la durée minimale d'un état. Ce modèle est formellement défini comme suit :

Définition 2.13 (Les réseaux de Petri T-temporisés) **:** Un RdP T-temporisé PN_T est un couple (PN, Tempo) tel que :

- PN est un RdP autonome marqué ;
- Tempo : $T \rightarrow Q^+$

  $T_j \rightarrow d_j$ = durée de temporisation associée à la transition T_j.

❑

Exemple 2.4 : Reprenons le système décrit en exemple 2.3 et considérons les contraintes temporelles suivantes :

i. entre deux arrivées successives de pièces s'écoulent 3 unités de temps.
ii. La durée opératoire d'une pièce sur la machine M consomme 2 unités de temps.

Ce nouveau système est modélisé par le RdP en figure 2.6.b où la transition T_1 représente l'arrivé des pièces ; et la transition T_2 représente l'accès des pièces à la machine M. Cet accès ne peur surgir qu'au toutes les deux u.t., ce qui permet le changement de structure entre le RdP autonome de la figure 2.5.b et le RdP T-temporisé de la figure 2.7.b.

❑

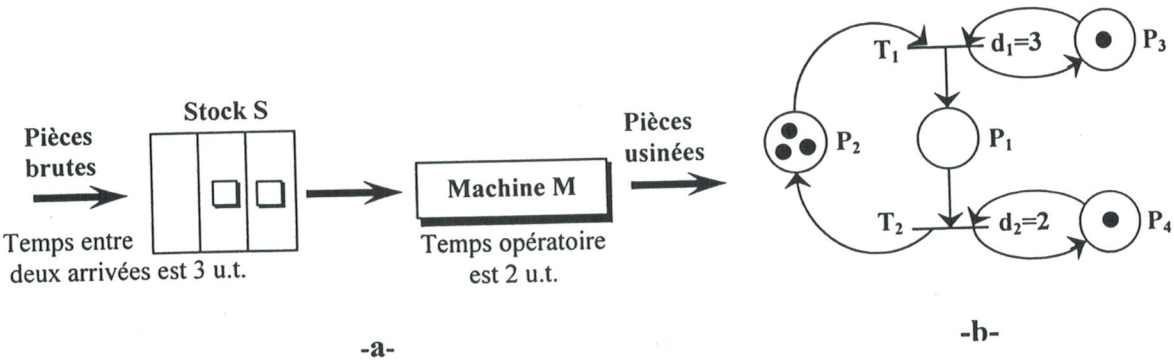


Figure 2.7. –a- Poste de travail avec contraintes temporelles. –b- RdP T-temporisé modèle du poste de travail.

Dans un RdP T-temporisé à chaque transition T_j, on affecte une constante d_j qui représente la durée minimale entre l'instant de validation de T_j et sa date de franchissement. En d'autres termes une transition T_j peut être franchise si elle est restée validée d'une manière continue, (c à d sans être dé-validée part le franchissement d'une autre transition T_k en conflit avec T_j) pendant d_j unités de temps.

Un fonctionnement "*au plus tôt*", ou "*à vitesse maximale*" est défini pour les RdP T-temporisés. Une transition T_j est franchie après avoir été validée d'une manière continue pendant exactement sa temporisation.

L'état d'un RdP T-temporisé fonctionnant à sa vitesse maximale est donné par son marquage et le vecteur des durées résiduelles de franchissement. Ce dernier et défini comme suit :

Définition 2.14 (Vecteur des durées résiduelles de franchissement) : soit PN_T un RdP et T_j une transition qui s'est validée à l'instant t_1 et d_j sa temporisation ;

Pour tout instant t_2 tel que $t_1 \leq t_2 \leq t_1 + d_j$, et si T_j n'a pas été dé-validée par un autre franchissement, la durée $(t_1+d_j)-d_2$ est dite la durée résiduelle de franchissement de la transition T_j.

Pour une transition q-validée est associé un vecteur de durées de franchissement résiduelles.

❑

Dans la figure 2.7 nous représentons le fonctionnement au plus tôt du RdP T-temporisé de la figure 2.6.b, à l'instant initial seule la transition T_1 est validée son franchissement s'effectuera à t = 3 u.t. Ce qui donne lieu à une validation simultanée de T_1 et T_2, qui seront, respectivement, franchies à t = 5 et t = 6. Ce fonctionnement est schématisé en figure 2.8, sur laquelle on peut voir que les 3 premières unités de temps constituent un régime transitoire pour le fonctionnement du RdP, après lesquelles le fonctionnement va atteindre un régime périodique. En fait, ceci est une propreté générale des RdP temporisés bornés.

Propriété 2.1 : Le fonctionnement à vitesse maximale d'un RdP T-temporisé borné conduit à un *fonctionnement périodique*, au bout d'un temps fini.

❑

Les contraintes temporelles associées à un RdP temporisé permettent de modéliser la durée minimale associée à une activité, et/ou la date la plus tôt d'occurrence d'un événement. Ces modèles sont principalement utilisés pour l'analyse des performances et pour l'optimisation des chaînes de production [GM99]. Les modèles qui associent un certain indéterminisme aux durées des activités et aux dates d'occurrences des événements, en utilisant des intervalles de temps au lieu des constantes, généralisent l'approche des RdP temporisés. Ces modèles, dits RdP temporels, trouvent leur origine dans le modèle RdP T-temporel, que nous allons présenter au paragraphe suivant.

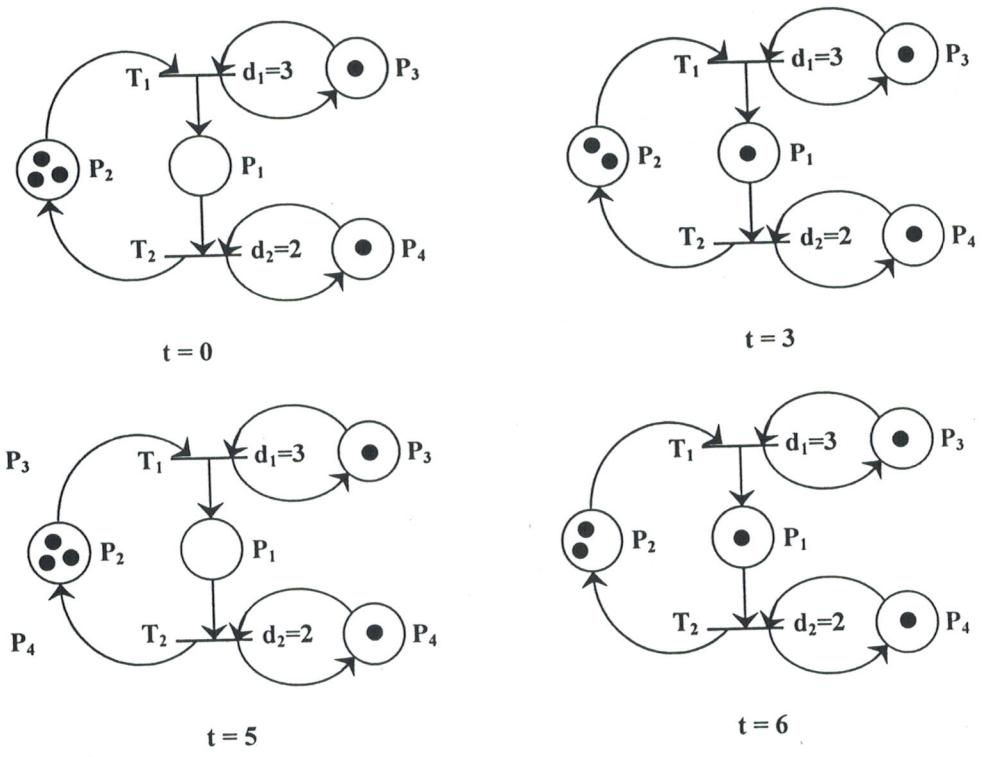


Figure 2.8. Le fonctionnement à vitesse maximale d'un RdP T-temporisé.

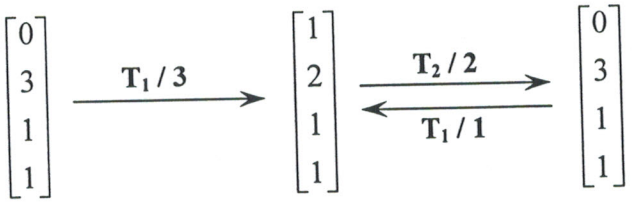


Figure 2.9. Graphe des marquages du fonctionnement à vitesse maximale du RdP T-temporisé en figure 2.4.

2.2.2.2. *Les réseaux de Petri T-temporels*

Les RdP T-temporels (Time Petri nets) ont été introduits en même temps que les RdP temporisés par Merlin dans sa thèse [Mer74]. L'idée fondatrice des RdP T-temporels est d'associer un intervalle de temps $[\alpha_j, \beta_j]$ à chaque transition T_j. Si cette dernière est validée de façon continue pendant au moins α_j unités de temps, elle peut être franchie. De plus si elle est validée pendant β_j unités de temps de manière continue, elle *doit être* franchie. On trouve ici les notions de délais minimum et maximum dans un état au lieu de durée d'un état dans les modèles temporisés.

Intuitivement, les RdP temporels généralisent les RdP temporisés, dans la mesure où, chaque modèle temporisé est équivalent à un modèle temporel ou les bornes inférieures des intervalles correspondent aux constantes du modèle temporisé, et toutes les bornes supérieures valent $+\infty$. De plus on peut aussi considérer que tout modèle RdP autonome est équivalent à un RdP temporel, où tous les intervalles valent $[0\ +\infty]$.

Définition 2.15 (Les réseaux de Petri T-temporels) : Un RdP T-temporel PNτ est un couple PNτ = (PN, SIM) tel que :

- PN est un RdP autonome marqué ;
- SIM : $T \rightarrow Q^+ \times (Q^+ \cup \{\infty\})$

  $T_j \rightarrow [\alpha_j^s, \beta_j^s]$ = intervalle statique de franchissement de la transition T_j.

SIM, la fonction d'intervalle statique (Static Interval Mapping), associe à chaque transition un intervalle statique de franchissement. Cet intervalle est dit *statique* car pendant l'évolution d'un tel RdP, des occurrences d'intervalles de franchissement *dynamiques* (qui évoluent avec le temps) apparaissent, ces derniers sont notés $[\alpha_j, \beta_j]$.

❑

Comme présenté auparavant, une transition T_j d'intervalle temporel $[\alpha_j, \beta_j]$ peut être franchie à un instant t si et seulement si $t \in [\theta+\alpha_j, \theta+\beta_j]$, θ représente l'instant ou le marquage a validé la transition T_j. C'est donc l'instant du dernier franchissement d'une transition.

Un des problèmes posés par l'analyse des RdP T-temporel est la gestion de la multi-validation des transitions, une transition T_j est dite multi-validée par le marquage M si $\forall P_i \in {}^\circ T_j : M(P_i) \geq 2 \,.\, Pré(P_i, T_j)$. Berthomieu et Diaz souligne dans [BD91] la difficulté d'analyse des RdP T-temporel intégrant la notion de la multi-validation. Dans [Boy01], l'auteur a présenté les difficultés liées à la gestion de la multi-validation. Généralement, pour analyser un RdP T-temporel, on suppose que le degré de validation maximum d'une transition est de 1.

L'état d'un RdP T-temporel a été défini de deux manières. Dans la définition originelle, présentée dans [BD91] on considère que l'état d'un RdP T-temporel est constant entre deux franchissements successifs de transitions, seules les transitions discrètes entres états sont considérées. Par contre, d'autres auteurs considèrent deux types de transitions d'état [Lim04], à savoir les transitions discrètes, causées par le franchissement d'une transition, et les transitions continues causées par l'écoulement du temps. La définition originelle est présentée ci-après.

Définition 2.16 (État d'un RdP T-temporel) : Soit le RdP T-temporel PNτ, un état *E* du RdP T-temporel PNτ est un couple (M, I) avec :

- $M : P \rightarrow \aleph$ est un marquage du PNτ.
- $I : T \rightarrow (Q^+ \times (Q^+ \cup \{\infty\})) \cup \{\varnothing\}$

$$T_j \rightarrow I(T_j) = \begin{cases} [\alpha_j, \beta_j] \text{ Si } T_j \in T^M \\ \varnothing \qquad \text{Si } T_j \notin T^M \end{cases}$$

I est le vecteur des intervalles de franchissement possibles pour les transitions validées, avec T^M l'ensemble des transitions validées par le marquage M

❑

L'état initial d'un RdP T-temporel $S_0 = (M_0, I_0)$ est constitué de son marquage initial M_0 et de vecteur I_0 qui comporte les intervalles statiques des transitions validées par M_0. Il est clair que le nombre d'états atteignables par un RdP T-temporel est infini. Construire un graphe des marquages accessibles est en général impossible, en effet les transition peuvent être tirées à tout instant dans leur intervalles de franchissement. Ce qui permet à chaque état d'admettre une infinité de successeurs, d'où l'idée de regrouper certains états dans une *classe d'états*, cette idée à pour but de permettre une représentation finie de ce graphe.

***Exemple** 2.5 :* Considérons les contraintes temporelles suivantes sur le fonctionnement du système décrit en exemple 2.3 :

i. Le temps opératoire de la machine M prend ses valeurs dans l'intervalle [2, 3]
ii. La période entre deux arrivées successives de deux pièces varie entre 2 et 5 unités de temps.

Dans cet exemple un certain indéterminisme est associé aux dates d'occurrence des événements. Le comportement de ce système est modélisé par le RdP T-temporel de la figure 2.6.

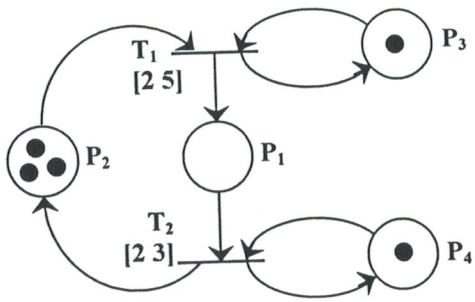


Figure 2.10. RdP T-temporel modèle du système décrit en exemple 2.5.

❑

Dans un RdP T-temporel, une transition T_j est franchissable à l'instant t ($t \in \Re^+$) depuis l'état E = (M, I), ce qui est noté $E[T_j\rangle_t$, si et seulement si les trois conditions suivantes sont satisfaites :

1. T_j est validée par le marquage M.
2. t appartient à l'intervalle de franchissement de T_j.

$$t \in I(T_j) = [\alpha_j, \beta_j].$$

3. Aucune autre transition ne doit être franchie avant t.

$$\forall T_k \in T^M, E[T_k\rangle, T_k \neq T_j, t \leq \beta_k.$$

On note par $E[T_j\rangle_t E'$, le fait que la transition T_j est franchissable depuis l'état E à l'instant t, et son franchissement donne lieu à l'état E'. L'état E' est tel que :

- M' est calculé comme pour un RdP autonome.

$$\forall P_i \in P, M'(P_i) = M(P_i) - Pré(T_j, P_i) + Post(T_j, P_i).$$

- I' est calculé en trois étapes :

1. Remplacer par ∅ tous les intervalles associés à des transitions validées par M, mais non validée par M - Pré (T_j). Cet ensemble noté D (E, T_J) comporte les transitions dé-validées par le franchissement de T_j depuis E.

$$D(E, T_J) = \{T_k \text{ tel que } E[T_k\rangle, \exists Pi \in P, M(P_i) - Pré(T_j, P_i) < Pré(T_k, P_i)\}$$

Cette première étape permet de définir un vecteur I_1' comme suit :

$$I_1'(T_k) = \begin{cases} \varnothing \text{ si } T_k \in D(E, T_J) \\ I(T_k) \quad \text{sinon} \end{cases}$$

2. On décale dans le temps tous les intervalles de I_1' pour obtenir I_2'

$$I_2'(T_k) = \begin{cases} [Max(0, \alpha_k - t), \beta_k] \text{ si } I_2'(T_k) = [\alpha_k, \beta_k] \\ \varnothing \quad \text{sinon} \end{cases}$$

3. On initialise les transition nouvellement validées par le nouveau marquage M'. Ainsi l'intervalle I' résultat est obtenu par :

$$I'(T_k) = \begin{cases} [\alpha_k, \beta_k] \text{ si } M'[T_k\rangle \text{ et } I_1'(T_k) = \varnothing \\ I_2'(T_k) \quad \text{sinon} \end{cases}$$

La règle de franchissement que nous venons de décrire, définit une relation d'accessibilité sur l'ensemble des états d'un RdP T-temporel. Les *séquences de franchissements* sont définies de la même manière que pour les RdP classiques.

Un *échéancier de franchissement* associe une séquence de franchissement S à une séquence de dates ω. C'est une séquence de couples :

$$(T_I, t_1) \rightarrow (T_{II}, t_{II}) \rightarrow \ldots \rightarrow (T_N, t_N)$$

Cet échéancier (S, ω) est dit *réalisable* depuis un état E si les transitions de la séquence S sont successivement franchissables depuis l'état E, aux dates relatives de franchissement qui leur correspondent dans la séquence ω.

Le fonctionnement d'un RdP T-temporel peut être caractérisé par l'ensemble des états accessibles depuis son état initial ou, de façon duale, par l'ensemble des échéanciers réalisables depuis son état initial.

Analyse d'accessibilité : La méthode des classes d'états

Comme précédemment mentionné, l'ensemble des états atteignables par un RdP T-temporel est en général infini. La méthode des classes d'états permet de grouper un certain nombre d'état en une classe d'état. Cette méthode est une méthode d'analyse par énumération pour les réseaux T-temporels introduit dans [Ber91] et améliorée dans [Lil99]. Elle permet, pour une large classe de réseaux temporels, une analyse d'accessibilité semblable à la méthode du graphe des marquages utilisée pour l'analyse d'accessibilité des RdP autonome.

Une classe d'états regroupe tous les états obtenus depuis l'état initial en franchissant une même séquence S. Tous ces états ont le même marquage, et leurs intervalles de franchissement ne diffèrent que par un décalage de certaines composantes.

Définition 2.17 (Classe d'état d'un RdP T-temporel) : Soit le RdP T-temporel PNτ, une classe d'état du PNτ est un couple C = (M, D) telle que :

- M est un marquage du RdP
- D est un polyèdre appelé domaine de franchissement ; Les inéquations de D sont de deux types [BD91]

$$\begin{cases} \alpha_i \le t_i \le \beta_i & \forall i \ \ tel \ \ que \ \ T_i \ \ est \ \ validée \\ \gamma_{kj} \le t_i - t_k \le \gamma_{jk} & \forall j,k \ \ tel \ \ que \ \ j \ne k \ \ et \ \ T_j, \ T_k \ \ validées \end{cases}$$

t_i représente l'instant de franchissement de la transition validée T_i relativement à l'instant où l'on est entré dans la classe.

❑

La classe initiale d'un RdP T-temporel est donnée par $C_0 = (M_0, D_0)$, où M_0 est le marquage initial et D_0 est le domaine donné par les intervalles de franchissement statiques des transitions validées par M_0. La transition T_j est franchissable depuis la classe C = (M, D) si :

i. T_j est validée par M.

ii. Le domaine D n'est pas vide, autrement dit le système d'inéquations définissant D admet des solutions.

Soit une classe C= (M, D) et une transition T_j franchissable depuis C ; la classe C' = (M', D') successeur de C par franchissement de T_j est donné par :

- Le nouveau marquage M' est calculé de façon classique par

$$\forall \ P_i \in P, \ M'(P_i) = M(P_i) - Pré\ (T_j, P_i) + Post\ (T_j, P_i) \ ;$$

- Le nouveau domaine de franchissement D' est calculé selon les étapes suivantes

  1. Changement des variables, $\forall\ i,\ t_i = t_j + t_i'$;
  2. $\forall\ T_i \ne T_j$, Ajout des contraintes $t_i' \ge 0$;

3. Elimination des variables correspondant à des transitions dé-validées par le franchissement de T_j (ce qui inclut T_j) ;
4. Ajout des inéquations relatives aux transitions nouvellement validées par le franchissement de T_j ;

Le changement de variables modélise l'écoulement du temps pour les transitions validées par un changement d'origine du temps. La nouvelle origine devient l'instant de franchir T_J. Ceci exprime que T_j est choisie franchie en premier et que toutes les autres transitions franchissables seront franchies après.

Dans [BER 91] les auteurs ont démontré que le nombre de classes atteints par un RdP T-temporel est fini si :

- Le RdP autonome sous-jacent est borné
- Les bornes des intervalles temporelles associés aux transitions prennent leur valeur dans Q (ensemble des rationnels), dans la définition originelle donnée par Merlin [Mer74] [MF76] ces bornes temporelles sont des réels positifs.

Le graphe des classes d'états accessibles (GCEA) par un RdP T-temporel est un graphe dont chaque nœud représente une classe d'états, et les arcs sont étiquetés par les noms de transitions. Ce graphe possède la propriété suivante : Il existe une séquence S menant de la classe $C_0 = (M_0, D_0)$ à la $C = (M, D)$ dans le GCEA si et seulement si il existe un échéancier (S, ω) menant de M_0 vers M. En conséquence le GCEA ne peut être utilisé que pour analyser des propriétés d'accessibilité de marquages, qui sont des propriétés non temporelles, ce qui est insuffisant pour l'analyse des propriétés *quantitatives*. D'où l'idée de traduire les RdP T-temporels en automate temporisé, et l'utilisation de ces derniers pour une analyse quantitative (temporelles).

2.2.3. *Les réseaux de Petri continus*

Les RdP continus (RdPC) ont été définis par David et Alla comme une limite des RdP discrets, obtenus à partir de ces derniers en fluidifiant les marques. Une place d'un RdPC est dite place continue ou C-place. Son marquage est un nombre réel positif ou nul. De même, une transition dans un RdPC est dite transition continue, ou C-transition, elle est validée si toutes ses places d'entrée sont marquées. A l'inverse des RdP discrets où le franchissement d'une transition est un événement instantané, une C-transition est franchie continûment dans le temps.

2.2.3.1. *Les réseaux de Petri continu autonomes*

Les RdPC est un outil de modélisation des systèmes dynamiques dont toutes les variables d'état sont continues, positives et linéaires par morceaux. Ils constituent aussi une bonne approximation pour la modélisation des systèmes dynamiques à événements discrets où les marquages sont grands. Pour illustrer l'idée de cette approximation, considérons l'exemple suivant.

Exemple 2.6 : La figure 2.11.a schématise une station de travail comportant une machine M de capacité unitaire, La machine puise les pièces dans un stock S_1 de capacité C_1 et alimente un deuxième stock S_2 de capacité C_2. Ce système possède $2(1 + C_1)(1 + C_2)$ états, ce nombre peut très rapidement exploser en fonction des valeurs numériques de C_1 et C_2.

❑

Le RdP de la figure 2.11.b, qui modélise le système pour $C_1 = C_2 = 6$, possède 98 marquages accessibles. Il peut atteindre 242 marquages pour $C_1 = C_2 = 10$. Comme mentionné précédemment, il est pratiquement difficile d'analyser un RdP dont le nombre d'états atteignables est important. C'est pour pallier à ce problème que les RdP continus ont été définis.

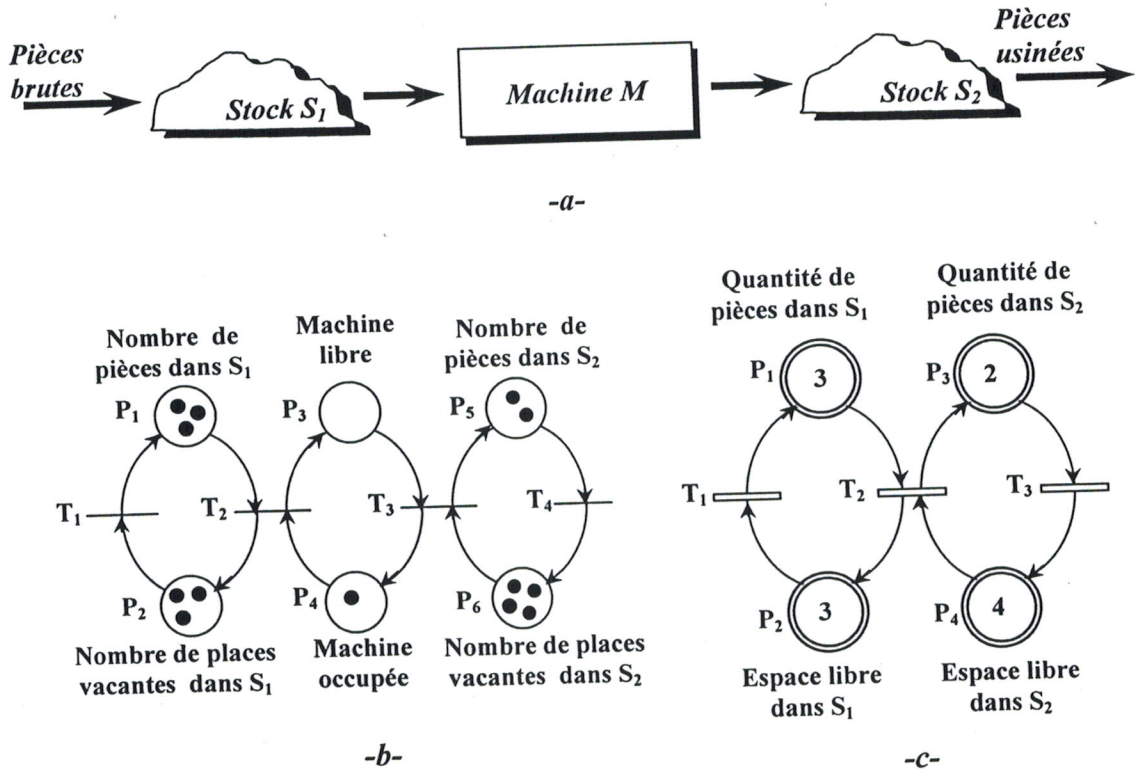


Figure 2.11. -a- Station de travail comportant une machine et ses stocks d'entrée et de sortie. -b- RdP discret modélisant la station -c- RdPC modélisant la station

Définition 2.18 (Réseau de Petri continu autonome marqué) : Un RdPC autonome et marqué est un quintuple PNc = (P, T, Pré, Post, M_0) ou on retrouve les même composants d'un RdP discret avec les différences suivantes :

i. Les applications d'incidences avant et arrière Pré et Post prennent leurs valeurs dans Q^+ (ensemble des rationnels positifs) au lieu de $\aleph$.

ii. De même les composants du vecteur M_0 prennent leurs valeurs dans $\Re^+$ (ensemble des réels) et non dans $\aleph$.

❑

Définition 2.19 (Validation et q-validation d'une C-transition) : Comme pour un RdP discret, La C-transition T_j est dite validée par le marquage M si ce dernier satisfait :

$$\forall P_i \in {}^{o}T_j, \frac{m_i}{Pr\acute{e}(P_i, T_j)} > 0$$

T_j est dite q-validée (avec q un réel positif) si et seulement si :

$$\forall P_i \in {}^{o}T_j, \min\left(\frac{m_i}{Pr\acute{e}(P_i, T_j)}\right) = q$$

Une C-transition T_j q-validée peut être franchie β fois simultanément, avec β un réel inférieur ou égal à q. β est dit *quantité de franchissement* de la C-transition Tj La notation $[T_J]^{\beta}$ dénote le franchissement simultané de la transition T_j β fois.

❑

Le RdPC en figure 2.11.c modélise le système décrit en exemple 2.6. Pour différencier un RdP continu d'un RdP discret, un double cercle est utilisé pour représenter une C-place et une barre vide pour représenter une C-transition. L'état d'une C-transition est booléen, elle est soit en franchissement soit non. C'est ce que nous permet de remplacer l'ensemble $\{T_2, T_3, P_3, P_4\}$ du RdP discret (figure 2.11.b), qui possède en fait deux états (Soit P_3 marquée, soit P_4 marquée) par la C-transition T_2 dans le RdPC (figure 2.3.c).

Le marquage atteignable d'un RdPC est non dénombrable, d'où l'impossibilité de construction d'un graphe des marquages accessibles. Ce dernier est remplacé par un graphe d'atteignabilité, ou chaque nœud modélise un ou plusieurs marquages. La construction de ce graphe est basée sur la notion du macro-marquage [DA05].

Définition 2.20 (macro-marquage) : Soit M_k un marquage d'un RdPC, l'ensemble des places P peut être divisé en deux sous-ensembles :

i. $P^+(M_k)$ l'ensemble des places P_i tel que $m_i > 0$

ii. $P^0(M_k)$ l'ensemble des places P_i tel que $m_i = 0$

Un macro-marquage, noté M^* est l'union de tous les marquages M_k ayant le même ensemble $P^+(M_k)$ de places marquées. Un macro-marquage est caractérisé par son ensemble de places marquées $P^+(M_k)$.

❑

Propriété : Un RdPC ayant n places peut atteindre au maximum 2^n macro-marquages.

❑

Dans un macro-marquage, l'état de toute place est booléen, elle est soit marquée soit non marquée. Un RdPC de n places à 2^n macro-marquages atteignables, et cela est vrai même s'il n'est pas borné, étant donné que le nombre de places d'un RdP est toujours fini (par définition d'un RdP).

Le RdPC en figure 2.11.c peut au maximum atteindre $2^4 = 16$ états, en réalité il n'a que 7 états atteignables (figure 2.12), puisque les places P_1 et P_2 sont complémentaires, $m_1 + m_2 = 6$, quelque soit le marque atteint. De même P_3 et P_4 sont complémentaires, $m_3 + m_4 = 6$. On ne pourra donc jamais atteindre un macro marquage où m_1 et m_2, ou m_3 et m_4 sont nuls en même temps, et en particulier on n'a pas le macro-marquage $[0\ 0\ 0\ 0]^T$.

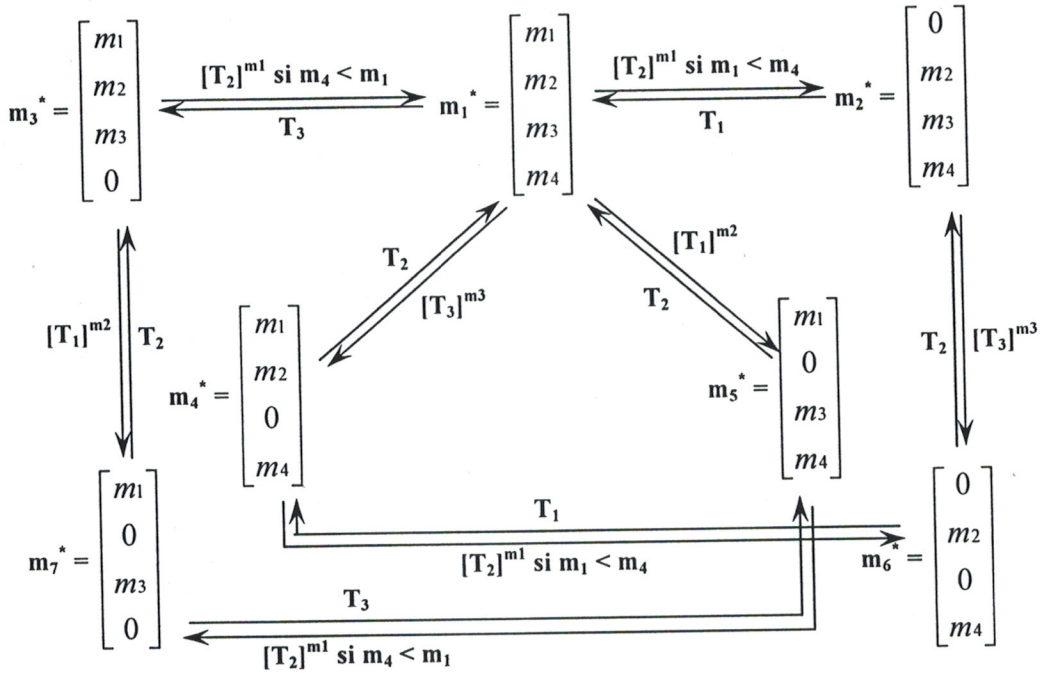


Figure 2.12. Graphe d'atteignabilité du RdPC de la figure 2.11.c.

2.2.3.2. *Les réseaux de Petri continu non autonome*

L'intégration explicite du temps dans les RdPC a été inspiré des RdP discrets T-temporisés. Dans un RdPC temporisé, on associe à chaque transition T_j sa vitesse de franchissement V_j. Les RdPC sont utilisés pour la modélisation des systèmes à flux continus, sous ce nom nous regroupons les systèmes continus positifs et les systèmes à événements discrets dont le flux de marques est important. Quatre types de RdPC ont été définis, la différence entre ces modèles réside dans la manière de définir les vitesses de franchissement des transitions. Dans ce qui suit les RdP continus à vitesse constante seront présentés en détail, les autres modèles seront brièvement présentés ensuite.

- **Le RdP Continu à Vitesses Constantes [DA92]**

Des quatre modèles RdPC les RdPC à vitesse constante a été le premier à être défini c'est aussi le plus simple. Une vitesse de franchissement maximale est associée à chaque transition T_j. Tant que la place d'entrée de la transition n'est pas vide, la transition est franchie selon sa vitesse maximale. Si la place d'entrée est vide mais qu'elle est alimentée en amont par une autre transition T_k, alors la vitesse de franchissement est le minimum des vitesses maximales des deux transitions T_j et T_k.

Définition 2.21 (RdP continu à vitesse constante) : Un RdPCC est un couple $PN_{CC} = (PNc, V)$ tel que :

- PN_C est RdPC autonome ;
- $V : T \rightarrow \Re^+$

  $T_j \rightarrow V_j$ = la vitesse de franchissement maximale de la transition T_j ;

❑

Le fonctionnement de ce modèle est clairement exprimé à travers l'équation fondamentale du RdPC. L'équation fondamentale du RdP Discret qui est

$$M' = M + W \cdot \overline{S}$$

devient pour un RdPCC

$$\dot{M} = W.v(t)$$

Où W est la matrice d'incidence du RdPC et $v\ (t)$ est le vecteur des vitesses de franchissements instantanées des transitions.

A titre d'illustration considérons les deux exemples suivants :

Exemple 2.7 : Considérons un système de réservoirs constitué de deux réservoirs et de trois vannes, comme illustré en figure 2.13.a. Les vannes sont caractérisées par leur débit V_1, V_2 et V_3 (litres/seconde), et sont supposées être ouvertes en permanence.

❑

Exemple 2.8 : Un système manufacturier comporte trois machines M_1, M_2 et M_3, dont les vitesses de traitement des pièces sont respectivement V_1, V_2 et V_3 (pièces/minute). Le système comporte aussi deux stocks tampons S_1 et S_2 pour l'attente des disponibilités des machines (figure 2.13.b).

❑

Les deux systèmes décrits en exemples 2.7 et 2.8 sont modélisés par le RdPCC de la figure 2.13.c. pour les valeurs numériques suivants : $V_1 = 3$, $V_2 = 4$ et $V_3 = 6$.

Les marquages des places P_1 et P_2 représentent, les quantités de liquide dans les réservoirs ou le nombre des pièces dans les stocks. Les vitesses associées aux transitions du RdPC modélisent les débits des vannes ou les vitesses des machines.

A l'instant initial, les trois transitions sont fortement validées, une transition est dite fortement validée si toutes ses places d'entrée sont marquées, et elles sont franchies à leurs vitesses maximales. Le marquage des places P_1 et P_2 (Figure 2.14) évolue suivant les équations suivantes :

$$m_1 (t + dt) = m_1 (t) + (V_1 - V_2).dt$$

$$m_2 (t + dt) = m_2 (t) + (V_2 - V_3).dt$$

Puisque $M_0 = [20\ 15]^T$

$m_1(t) = 20 - t$

$m_2(t) = 15 - 2.t$

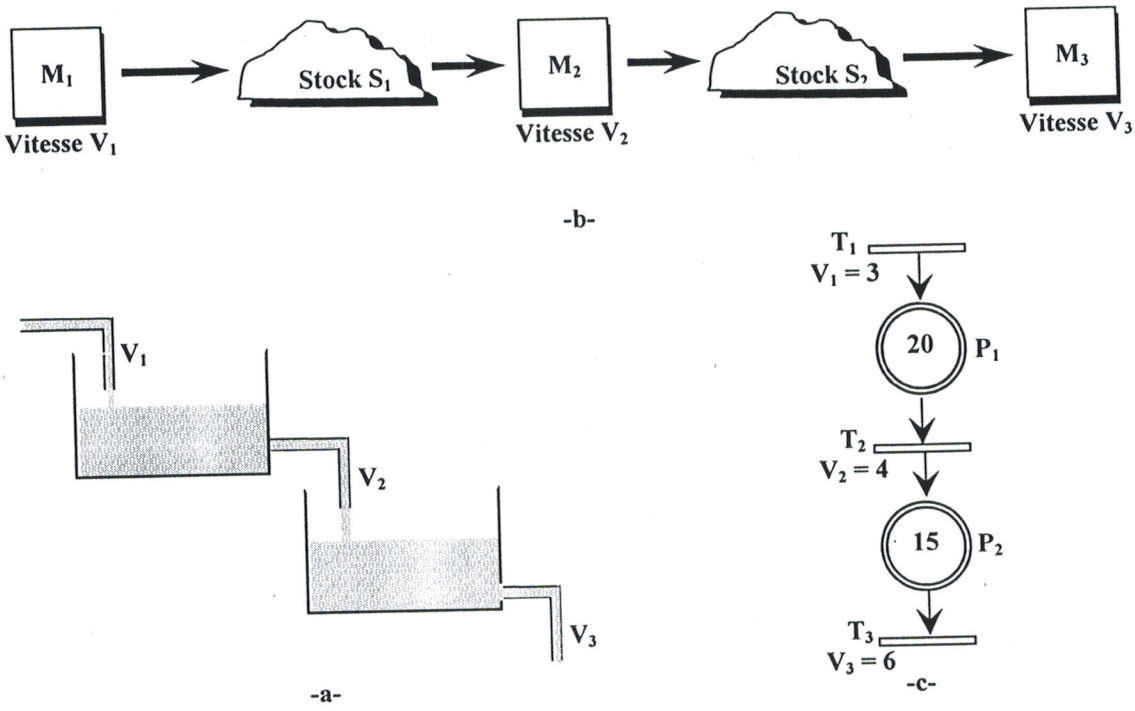


Figure 2.13.-a-System hydraulique. –b- Système manufacturier. -c- RdPC modèle des systèmes en figures a et b.

Ces équations restent vraies aussi longtemps que $m_1 > 0$ et $m_2 > 0$. A t = 7.5s m_2 s'annule, ce qui empêche T_3 d'être franchie à sa vitesse maximale. Mais puisque la place P_2 est toujours alimentée à une vitesse de 4 litres/s par la transition T_2, T_3 est aussi franchie avec cette même vitesse, qui n'est plus sa vitesse maximale, T_3 est dite faiblement validée (Sa seule place d'entrée n'est pas marquée mais elle est alimentée). A t = 20s m_1 s'annule, ce qui empêche T_2 d'être franchie à sa vitesse maximale, elle sera franchie à la vitesse de T_1.

Par convention les vitesses maximales de franchissement sont notées par des majuscules alors que les vitesses instantanées de franchissement par des minuscules. Le vecteur des vitesse instantané du RdPCC en figure 2.13.c est donné par :

$$\begin{pmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \end{pmatrix} = \begin{cases} [V_1 \;\; V_2 \;\; V_3]^T = [3 \;\; 4 \;\; 6]^T & Pour \;\; 0 \le t \le 7.5 \\ [V_1 \;\; V_2 \;\; V_2]^T = [3 \;\; 4 \;\; 4]^T & Pour \;\; 7.5 \le t \le 20 \\ [V_1 \;\; V_2 \;\; V_1]^T = [3 \;\; 3 \;\; 3]^T & Pour \;\; t \ge 20 \end{cases}$$

L'évolution du marquage en fonction du temps est illustré en figure 2.14.

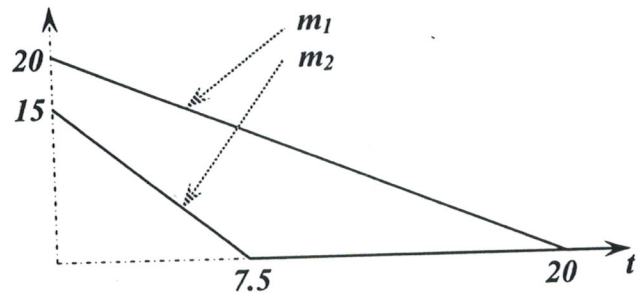


Figure 2.14 Illustration du comportement du RdPCC en figure 2.13.c.

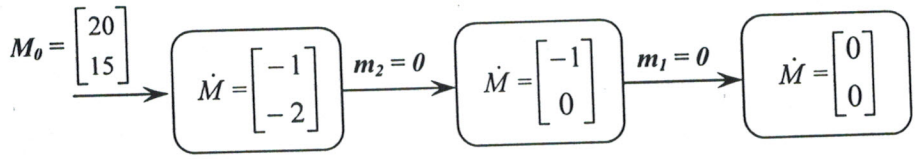


Figure 2.15 Graphe d'évolution du RdPCC en figure 2.13.c.

L'état d'un RdPCC est caractérisé soit par le vecteur des franchissements instantanés ou, par dualité, par les dérivés des marquages. Le seul événement susceptible de changer l'état d'un RdPCC est que le marquage d'une place devient nul. Le comportement de ce modèle est généralement représenté par un graphe d'évolution dont les nœuds correspondent aux vecteurs des dérivés des marquages et les arcs sont étiquetés par les places dont l'annulation de marquage a causé le changement d'état. Il est évident que chaque état correspond à un macro-marquage. Une des propriétés les plus importantes du modèle RdPCC est que son graphe d'évolution a toujours un nombre fini de nœuds même si le RdPCC est non borné. À titre d'illustration le graphe d'évolution du RdPCC en figure 2.13.c est représenté en figure 2.15. Le graphe d'évolution d'un RdPCC est facilement assimilable à un automate hybride, où les variables d'état sont données les marquages des C-places et les gardes des transitions sont de la forme $m_i = 0$. Nous avons ici un modèle de système continu qui a un comportement événementiel.

- **Le RdP Continu à Vitesses Variables [DA92]**

Dans ce modèle la vitesse maximale de franchissement d'une transition dépend du marquage des places en amont de la transition et d'une valeur constante associée à la transition. Les marquages et les vitesses sont donc des fonctions continues du temps, alors que dans le RdP continu à vitesse constante cette propriété est vraie pour les marquages mais pas pour les vitesses qui sont simplement constantes par morceaux. Ce modèle fournit une meilleure approximation du RdP discret, en particulier lorsque le nombre de marques est petit, mais les simulations sont plus longues. Ici, le comportement événementiel est perdu, il faut discrétiser le temps pour faire une simulation.

- **Le RdP Continu Asymptotique [Bail et al. 93]**

Il est possible de distinguer plusieurs phases d'évolution du marquage. Chaque phase est caractérisée par une période durant laquelle le vecteur des vitesses de franchissement est constant. La vitesse de franchissement d'une transition est alors constante par morceaux, et l'évolution du marquage est donc composée de plusieurs phases de fonctionnement. Il s'agit de l'approximation du RdP Continu à Vitesses Variables, mais la simulation est facilitée.

- **Le RdP Continu à Vitesses Fonction du Temps [Dubois et al. 94]**

Cette extension des RdP continus temporisés permet de prendre en compte l'environnement du système modélisé comme par exemple la fluctuation de l'approvisionnement d'un atelier ou l'influence d'une commande extérieure. La vitesse maximale de franchissement d'une transition est soit une fonction continue du temps, soit une fonction constante par morceaux. La vitesse de franchissement à un instant donné est déterminée par une relation identique à celle du RdP continu temporisé, avec comme seule différence une vitesse maximale fonction du temps. Dans le cas de vitesses constantes par paliers, le RdP obtenu peut être vu comme une suite de RdP continus à vitesses constantes.

Le modèle RdPC est donc utilisé soit modéliser un système continu, soit pour constituer une approximation d'un système à événements discret, plus ou moins finement selon la modèle utilisée. Dans ce dernier cas, il permet de réduire le nombre d'événements qui devient importantes lorsqu'un modèle RdP discret contient un grand nombre de jetons. Ces modèles ne traitent pas l'évolution du système événement par événement mais modélisent des variations moyennes.

2.2.4. *Les réseaux de Petri hybrides*

Les réseaux de Petri continus sont adaptés pour la modélisation d'un système à flux continu mais ne permettent pas de représenter l'influence de conditions logique sur ce flux. Considérons le cas de l'exemple 2.7, le système décrit dans cet exemple comporte des vannes, le transfert de matière à travers ces vannes a deux états, si la vanne est ouverte, le débit est traduit par une vitesse maximale de franchissement d'une transition V_{max}, et si la vanne est fermée, la vitesse correspondante devient alors nulle au lieu de V_{max}. Ce comportement est équivalent à avoir brusquement un autre RdP continu, ou bien à faire varier la vitesse de franchissement d'une transition dans un réseau de Petri Continu à Vitesse Fonction du Temps. Les RdP Hybrides [LeB92] contenant des nœuds discrets (des D-places et des D-transitions) et des nœuds continues (des C-places et des C-transitions) permettent de représenter au sein d'un même formalisme ce couplage entre un fonctionnement continu et un fonctionnement logique (représentation graphique unifiée des variables continues et des variables discrètes).

Un RdP hybride (RdPH) est un modèle combinant un RdP discret et un autre continu, aucune condition n'est imposée sur la nature des RdP discret et continu, en effet ces derniers peuvent être de n'importe quelle nature. Toutefois le modèle RdPH

de base combine un RdPCC et un RdP discret T-temporisé, et c'est le modèle défini dans ce qui suit.

Définition 2.22 (RdP hybride) : Un RdP hybride est un octuple $PN_H = (P, T, Pré, Post, h, Tempo, V, M_0)$, tel que :

- $P = \{P_1, P_2, \ldots, P_m\}$ est un ensemble de m places, $P = P^D \cup P^C$ avec :

  $P^D = \{P_1, P_2, \ldots, P_{m'}\}$ est l'ensemble de m' D-places,

  $P^C = \{P_{m'+1}, \ldots, P_m\}$ est l'ensemble des places continues ;
- $T = \{T_1, T_2, \ldots, T_n\}$ est un ensemble de n transitions, $T = T^D \cup T^C$ avec :

  $T^D = \{T_1, T_2, \ldots, T_{n'}\}$ est l'ensemble de n' D-transitions,

  $T^C = \{T_{n'+1}, \ldots, T_n\}$ est l'ensemble des C-transitions ;
- *Pré* et *Post* désignent respectivement les applications d'incidence avant et arrière ; ces applications doivent satisfaire la condition suivante :

  $$\forall (P_i, T_j) \in P^D \times T^C : Pré (P_i, T_j) = Post (P_i, T_j) ;$$
- $h : P \cup T \rightarrow \{D, C\}$ est une application qui désigne les nœuds discrets, $h(x) = D$, des nœuds continus, $h(x) = C$;
- $Tempo : T^D \rightarrow \mathbf{Q}^+$ est une application qui associe à chaque D-transition sa duré de temporisation.
- $V : T^C \rightarrow \Re^+$ est une application qui associe à chaque C-transition sa vitesse maximale de franchissement :
- M_0 est le marquage initial, les D-places contiennent un marquage entier positif et les C-places contiennent un marquage réel positif ;

❑

La condition sur les applications d'incidence avant et arrière est repérée sur le RdP par des boucles liants les D-Places aux C-transitions, elle signifie qu'une marque discrète ne peut pas être fluidifiée par une transition continue. Le modèle RdP hybride ainsi défini permet donc la modélisation des conditions logiques influant sur le comportement du système, mais il permet aussi la modélisation de transformation de lot en pièces et vis-versa (de pièces en lot).

Numéroter les nœuds du RdP de telle sorte que les nœuds discrets aient les indices les plus petits, comme présenté en définition 2.22, fait que la matrice d'incidence a la forme suivante :

$$W = \begin{bmatrix} W_D & 0 \\ W_{CD} & W_C \end{bmatrix}$$

Les RdP élémentaires constituent une classe particulière de RdP hybrides où il n'y a pas de transformation de marquage, du discret vers le continu ou du continu vers le discret. Dans ce modèle le RdP T-temporisé contrôle le comportement du RdPCC via des boucles connectant certaines D-places à certaines C-transitions, ce qui signifie que ces dernières ne sont validées et par conséquent ne peuvent être franchies que si les D-places sont marquées. Le RdPCC à son tour peut influencer le

comportement du RdP T-temporisé, une D-transition T_j peut avoir comme condition de franchissement le marquage d'une C-place P_i qui atteint un seuil *S*. Graphiquement, ceci est représenté de deux manières soit par une boucle (deux arcs de P_i vers T_j et de T_j vers P_i) dont le poids est *S* si ce seuil est un seuil supérieur, c-à-d, si le marquage de P_i ne peut être supérieur à *S*. dans le cas contraire, si le marquage de Pi ne doit pas être inférieur à *S*, un arc inhibiteur est utilisé pour relier T_j à P_i, et dans les deux cas le franchissement de T_j ne modifie pas le marquage de P_i.

Définition 2.23 (RdP hybrides élémentaire) : Un RdPH élémentaire est un couple (PN_H, I) tel que :

- PN_H est un RdP dont les applications Pré et Post satisfais la condition suivante :

$$\forall (P_i, T_j) \in (P^D \times T^C) \cup (P^C \times T^D), \text{ alors Pré } (P_i, T_j) = \text{Post } (P_i, T_j)$$

- $I : (P_i, T_j) \rightarrow \Re$, est une application d'inhibition, si un arc inhibiteur de poids S relie la place P_i à la transition T_j, le franchissement de T_j n'est possible que si le marquage de P_i est inférieur à S.

❑

Exemple 2.9 : Reprenons l'exemple 2.7, et supposons qu'on veut limiter le niveau de liquide dans le bac 2 entre S_{min} et S_{max}. en fermant et ouvrant la vanne 2 (Figure 2.16.a). Ce système est modélisé par le RdPH élémentaire en figure 2.16.b. Dans ce modèle le franchissement de la C-transition T_5 n'est possible que si la D-place P_2 est marquée. De même franchir les D-transitions T_1 et T_2 n'est possible que si le marquage de la C-place est respectivement supérieur à S_{max} et inférieur à S_{min}.

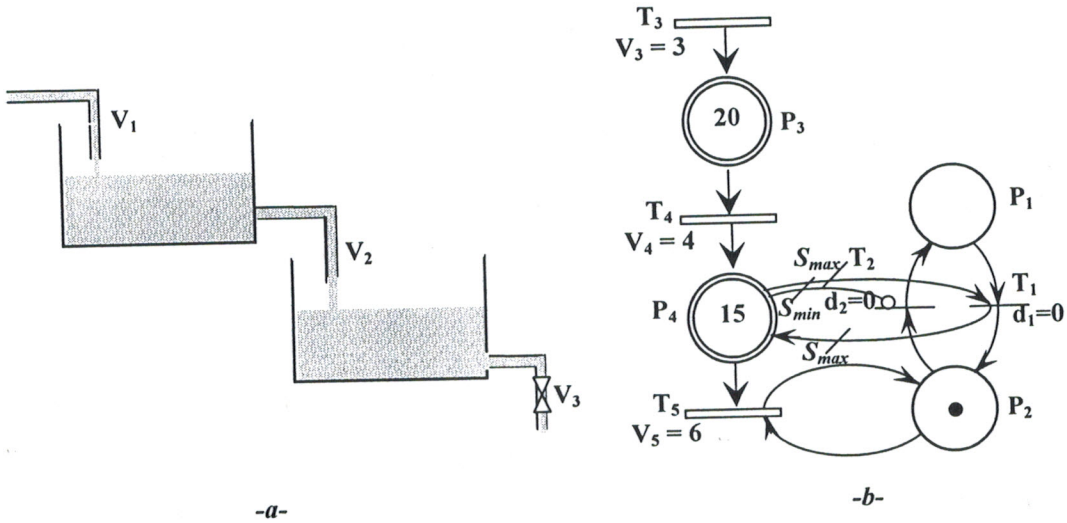


Figure 2.16 –a- Système de réservoirs, -b- RdPH élémentaire le modélisant.

Le modèle RdPH présenté en définition 2.22 est un modèle déterministe dans le sens ou, connaissant son état à l'instant initial, on peut connaître exactement son état à n'importe quel instant t et cet état est unique (à condition de choisir une

politique de résolution des conflits). L'état de ce modèle est donné par l'état des parties discrète est continue. Les événements qui peuvent changer l'état d'un RdPH sont de deux types, ce sont les événements qui peuvent changer l'état d'un RdP T-temporisé et ceux qui change l'état d'un RdPCC, à savoir :

i. Une D-transition est franchie ;
ii. Le marquage d'une C-place s'annule ;
iii. Le degré de validation d'une D-transition change à cause du marquage d'une C-place. (Le degré de validation d'une D-transition change soir par le franchissement d'une autre D-transition soit parce que le marquage d'une C-place a atteint un certain seuil).

Entre deux événements successifs l'état du modèle, donné par les dérivés des marquages des C-places et les marquages des D-places, est le même. Cet état est appelé IB-état (Invariant Behavior state) ou état comportemental invariant.

Définition 2.24 (IB-état d'un RdPH) : Un IB-état d'un RdPH est un intervalle de temps pendant lequel :

i. Le marquage des D-places est constant ;
ii. Les dérivés des marquages des C-places sont constants ;
iii. Les degrés de validation des D-transition sont constants ;

❑

Le comportement d'un RdPH est généralement modélisé par un graphe d'évolution dont les nœuds représentent les IB-états et les arcs sont étiquetés par les événements dont l'occurrence change l''IB-état avec leurs date d'occurrence. Le graphe d'évolution du RdPH élémentaires en figure 2.16.b possède 7 IB-états pour $S_{min} = 10$ et $S_{max} = 30$, comme présenté en figure 1.17.

2.3 Relations entre les automates et les réseaux de Petri

La relation entre les automates et les RdP est ancienne, le principal outil d'analyse des RdP est le graphe des marquages accessibles qui peut être vu comme un automate à états fini étiqueté par les noms des transitions. et c'est le cas pour plusieurs autres extensions et abréviations des RdP, leur analyse se fait après leur traduction en automate.

D'une manière générale, les automates (discrets temporisés ou hybrides) sont des modèles qui permettent une manipulation formelle facile mais sont difficilement utilisables pour la modélisation, contrairement aux RdP qui présentent l'avantage d'avoir une modélisation intuitivement claire mais ne présentent pas une énumération exhaustive de l'espace d'état. Cependant, cela rend difficile la tâche d'analyse. D'où l'idée de coupler la puissance d'analyse des automates à la puissance de modélisation des RdP. Cette idée a été utilisée principalement pour les extensions des réseaux de Petri temporels et hybrides.

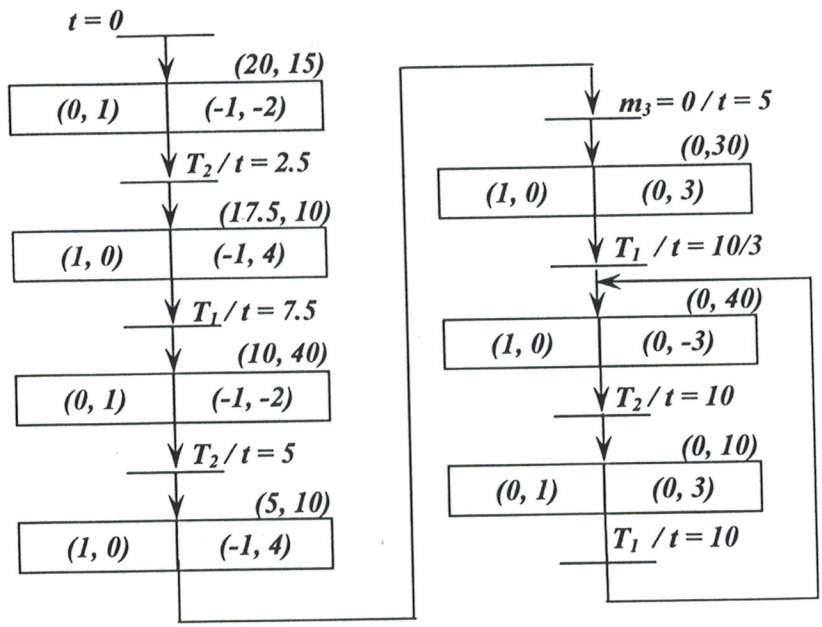


Figure 2.17 Graphe d'évolution du RdPH élémentaire de la figure 2.16.b.

Dans le domaine des RdP temporels, le premier travail consacré à la traduction des RdP en automates a été proposé par Sifakis et Yovine [SY96] qui étudient une sous-classe des réseaux de Petri temporels à flux dont le réseau sous-jacent est sauf (STPN). Dans ce modèle, étant donnée une transition T_j et une place amont P_i de T_j, un intervalle de temps [α β] est associé à l'arc (P_i T_j). Un jeton arrivant dans la place P_i doit attendre une durée comprise entre α et β avant d'être disponible pour la transition T_j. Les auteurs proposent une traduction de ces RdP en automates temporisés ; une horloge est associée à chaque place du RdP. Elle est remise à zéro à chaque fois que la place reçoit un jeton. Les sommets de l'automate correspondent au graphe des marquages du RdP. Les gardes et les invariants sont dérivés des intervalles associés aux arcs.

Dans [BST98], Bornot, Sifakis et Tripakis s'intéressent aux RdP avec échéances (PND) qui sont des RdP saufs, étendus avec des horloges. Un PND peut être défini comme un automate temporisé à échéances (TAD) dont la structure discrète est le graphe des marquages du RdP. Les transitions de ce TAD sont soumises aux mêmes contraintes temporelles que les transitions du PND. Le PND et le TAD ont le même nombre d'horloges. Les auteurs proposent une traduction des RdP temporels en TAD avec une horloge par arc entrant du RdP temporel initial. Les automates temporisés à échéances pouvant être considérés comme des automates temporisés standard, la méthode fournit donc, par transitivité, une traduction des RdP temporels dont le RdP sous-jacent est sauf vers les automates temporisés. Le résultat possède un nombre d'horloges supérieur ou égal au nombre de transitions du RdP initial.

Sava et Alla [Sav01] [SA01] considèrent les RdP T-temporels bornés dont le RdP sous-jacent n'est pas nécessairement sauf et proposent un algorithme calculant le graphe des marquages du RdP T-temporel sous la forme d'un automate temporisé.

L'automate résultat possède une horloge pour chaque transition du RdP T-temporel. Ce travail nécessite le calcul de l'espace d'état du RdP T-temporel. L'automate résultat est ensuite restreint pour la synthèse de contrôleur.

Lime et Roux [LR03] proposent une extension du graphe des classes d'état qui permet de construire le graphe des classes d'états comme un automate temporisé. Les auteurs prouvent la minimalité relative de nombre d'horloges de l'automate résultat.

Cassez et Roux ont développé une méthode de traduction structurelle d'un RdP T-temporel en un automate temporisé [CR03, CR04]. Chaque transition du RdP est encodée dans un automate temporisé avec variables. Tous les automates ainsi obtenus sont combinés avec un superviseur en un produit synchronisé d'automates temporisés. Le calcul est très rapide et applicable à tous les RdP T-temporel même non bornés car il est structurel. La méthode n'est cependant pas bien adaptée à la vérification car le produit d'automates possède une horloge par transition du RdP T-temporel et ce nombre ne peut être réduit par des algorithmes à cause de la structure de produit du résultat.

La plupart des extensions des RdP vers l'hybride ont été traduites en automates hybrides en vue de leur analyse. C'est le cas des RdPH présentés en paragraphe 2.2.4. Ce modèle est la première extension des RdP vers l'hybride a être établie, Allam et Alla ont présenté, dans [AA98] et [All98], un algorithme qui permet la construction de l'automate hybride équivalent au RdPH, l'automate résultat à autant de sommets que d'IB-états du RdPH, les variables d'état étant les marquages des places continues et les horloges qui mesurent le temps pour les transitions validées. L'automate résultat est linéaire et déterministe. Pour assurer la convergence de cet algorithme de passage, le RdPH doit être borné et les temporisations associées aux transitions doivent être des nombres rationnels positifs.

En s'inspirant du travail de Allam et Alla [AA98] [All98], Demongodin et Rouibia [DR03] ont présenté une procédure qui permet de systématiser le passage des RdP lots en automates hybrides. L'automate résultat a un sommet pour chaque IB-état du RdP lots. Si ce dernier vérifie les conditions de bornitude et de rationalité des temporisations associées aux transitions discrètes, les auteurs montrent la convergence de l'algorithme. La méthode d'analyse en avant est ensuite utilisée pour calculer la région atteignable, à partir de la région initiale définie par le marquage initial. Le calcul de la région atteignable, permet de décrire le fonctionnement périodique du RdP lots.

Conclusion

Nous avons présent" dans ce chapitre deux modèles pour la modélisation et l'analyse des systèmes dynamiques hybrides. Les automates hybrides, qui constituent le modèle le plus général et le plus utilisé en analyse, sont d'abord présentés. Ce modèle présente l'avantage de permettre une énumération exhaustive

de l'espace d'état pour certaines de ces classes, ce qui facilite son analyse. Le deuxième modèle présenté est le RdPH. La principale motivation pour utiliser les RdP pour modéliser les systèmes dynamiques hybrides, est que tous les avantages qui font des RdP un modèle puissant pour les SED, sont toujours vrais pour les RdPH, à savoir que le RdP n'exige pas une énumération exhaustive de l'espace d'état et peut donc représenter d'une manière finie des systèmes dont l'espace d'état est infini. De plus leur utilisation pour la modélisation est intuitivement claire. Certains travaux associant la puissance d'analyse des automates hybrides à la puissance de modélisation des RdP ont été aussi présentés.

Dans le chapitre suivant nous allons présenter le RdPH D-élémentaire qui est une sous-classe des RdPH et que nous allons utiliser pour la synthèse de contrôleur des systèmes dynamiques hybrides. Une procédure qui permet le passage des RdPH D-élémentaires en automates hybrides sera présentée.

Construction du Modèle Automate Structurel à partir du RdP hybride

Dans ce chapitre nous présentons un nouveau modèle, il s'agit du RdPH D-élémentaire, que nous allons utiliser pour la synthèse de contrôleur des systèmes dynamiques hybrides. Le modèle RdPH D-élémentaire se distingue du RdPH élémentaire présenté en paragraphe 2.2.4 dans le fait qu'il combine un RdP continu à vitesse constante et un RdP T-temporel, de plus le comportement de la partie discrète commande la partie continue, mais cette dernière n'a aucune influence sur la partie discrète. Le non déterminisme associé à ce modèle est représenté par les intervalles de franchissement associés aux transitions discrètes. Ceci permet d'utiliser ces dernières comme points de commande, lorsqu'elle modélise des événements contrôlables. Une procédure qui permet la traduction des RdPH D-élémentaires en automates hybrides est présentée dans la deuxième partie du chapitre.

3.1 Le modèle réseau de Petri hybride considéré

Un modèle RdP hybride est génériquement défini comme une combinaison d'un RdP discret et d'un RdP continu. Dans cette définition, aucune contrainte n'est imposée quand au type des modèles RdP discret et continu, en effet ces derniers peuvent être de n'importe quelle nature. Le modèle RdPH présenté en littérature [DA01] combine un RdPCC et un RdP T-temporisé fonctionnant en vitesse maximale, ce modèle a un comportement déterministe, tous les évènements ont des dates d'exécution précises. C'est pourquoi ce modèle a été essentiellement utilisé dans le cadre de l'évaluation de performances et n'a donc pas été utilisé pour définir une politique de contrôle.

Le modèle RDPH que nous considérons dans ce mémoire combine un RdP T-temporel et un RdP continu à vitesse constante. Ce modèle est élémentaire, c à d qu'il ne considère pas la transformation de marquage, de plus seule la partie discrète influence le comportement de la partie continue, et cette dernière n'a aucune

influence sur la partie discrète. Ce modèle est dit RdPH D-élémentaire puisqu'il est élémentaire et est contrôlé par la partie discrète.

3.1.1 *Exemples introductifs*

Dans un RdPH D-élémentaire, un événement n'a pas une date exacte d'occurrence mais un intervalle dans lequel l'événement a la possibilité de se produire. Pour introduire de manière informelle les RdPH . D-élémentaires, considérons les deux exemples suivants.

Exemple 3.1 : Soit le système des bacs présenté en exemple 1.2 et repris en Figure 3.1. Le modèle RdPH D-élémentaire en Figure 3.2 décrit ce système. Les places discrètes P_1 et P_2 modélisent la position de la vanne, si P_1 est marquée la vanne est en position A et si P_2 marquée la vanne est en position B. Les D-places P_1 et P_2 contrôlent respectivement les C-transitions T_3 et T_5, si P_1 (resp. P_2) est marquée T_3 (resp. T_5) est franchie à sa vitesse maximale, dans le cas contraire la vitesse de franchissement est nulle. Les intervalles de franchissement associés aux transitions T_1 et T_2 signifient que la vanne peut passer d'une position à l'autre à n'importe quel instant $t > 0.5$, l'origine du temps étant la date d'arrivée de la vanne à sa position actuelle, dans le modèle cette date correspond à l'instant de validation de la transition. La durée 0.5 correspond au temps nécessaire pour réaliser le déplacement.

❑

Exemple 3.2 : considérons le système manufacturier décrit en exemple 2.3 (Figure 3.3). Ce système est modélisé par le RdPH D-élémentaire en figure 3.4, dans ce modèle les C-Places P_5 et P_6 modélisent respectivement les quantités de pièces dans les stocks 1 et 2. Les C-transitions T_7 et T_8 sont respectivement contrôlées par les D-places P_1 et P_3. P_1 marquée signifie que la machine 3 est en marche, et P_3 marquée signifie que la demande est active. Les intervalles de franchissement associés aux D-transitions T_1 et T_2 signifient qu'on peut faire commuter la machine 3 d'un état à l'autre à n'importe quel instant $t > 2$, l'origine du temps étant l'instant d'entrée de la machine dans son état actuel. La demande est périodique de période 40 u.t.

❑

Dans un RdPH D-élémentaire, les parties discrètes et continue sont connectées à travers des boucles reliant des C-transitions à des D-places, c à d que les franchissements de ces C-transtions sont conditionnés par le marquage des D-places, la partie continue peut donc être contrôlée par la partie discrète. Cependant, franchir une transition discrète ne dépend que de son marquage et de la variable continue indépendante qui est le temps. Son comportement est donc indépendant est peut être étudié indépendamment de la partie continue. C'est cette idée de base que nous allons exploiter dans ce travail.

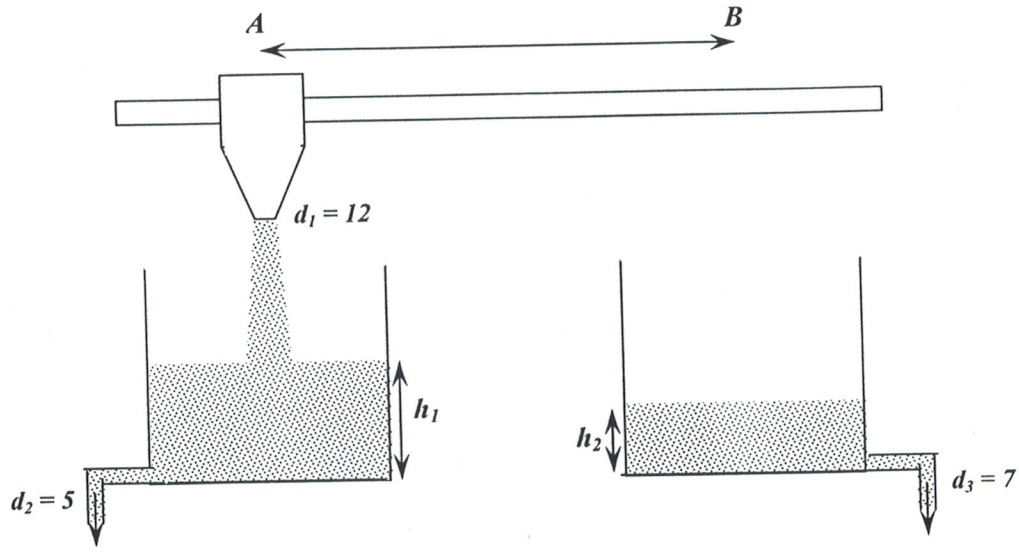


Figure 3.1 Système de bacs.

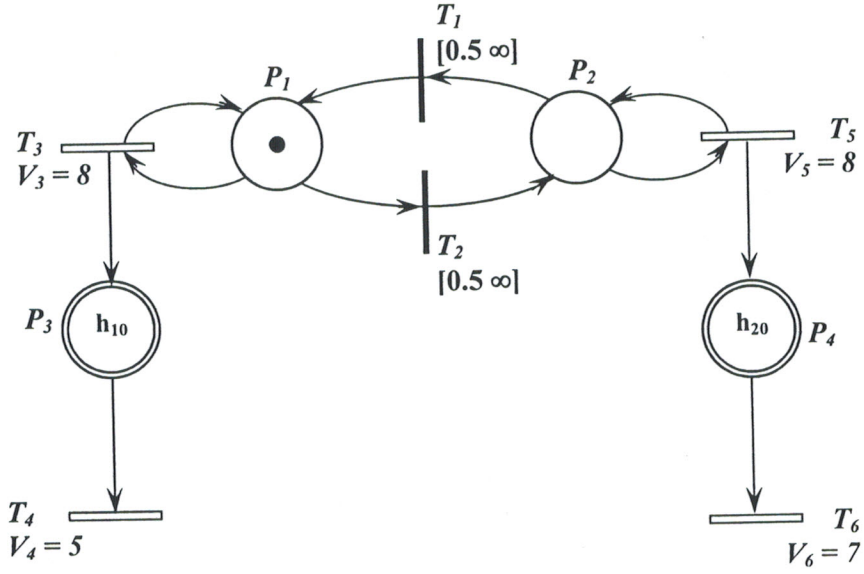


Figure 3.2 RdPH D-élémentaire modélisant le système des bacs en figure 3.1.

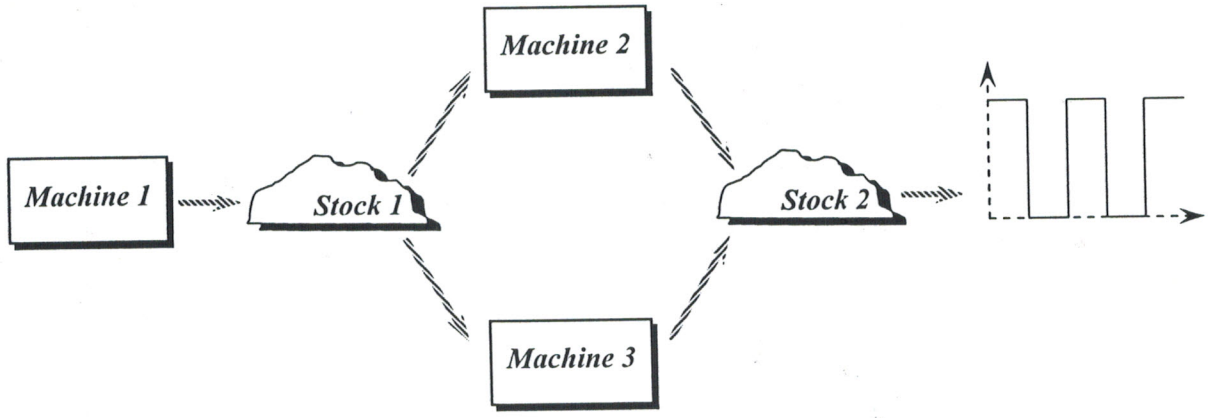


Figure 3.3. Système manufacturier.

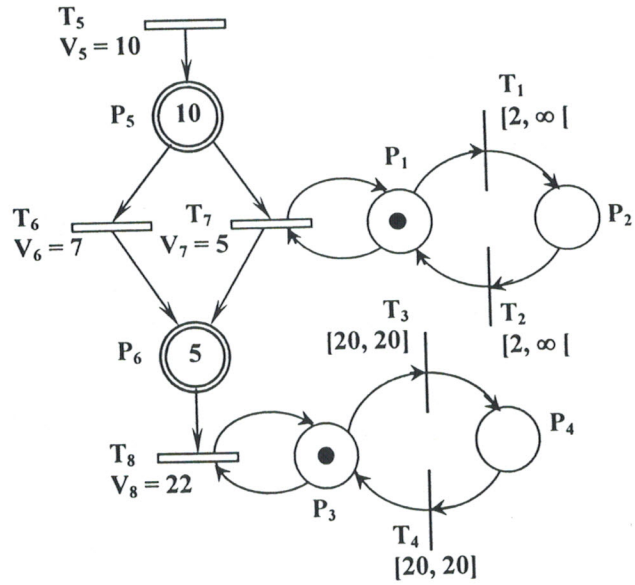


Figure 3.4 RdPH D-élémentaire modélisant le système manufacturier en figure 3.3.

3.1.2 *Définition et sémantique du modèle*

Un RdPH D-élémentaire est formellement défini comme suit.

***Définition 3.1* (RdP hybride D-élémentaire)** : Un RdP hybride D-élémentaire est un octuple $PN_D = (P, T, Pré, Post, h, Tempo, V, M_0)$ tel que :

- $P = \{P_1, P_2, \ldots, P_m\}$ est un ensemble fini de m places ;
- $T = \{T_1, T_2, \ldots, T_n\}$ est un ensemble fini de n transitions ;

On note par $P^D = \{P_1, P_2, \ldots, P_{m'}\}$ l'ensemble de m' D-places et $P^C = P - P^D$ l'ensemble des places continues. De la même façon, on note par $T^D = \{T_1, T_2, \ldots, T_{n'}\}$ l'ensemble des n' D-transitions et $T^C = T - T^D$ l'ensemble des transitions continues.

- *Pré* et *Post* désignent respectivement les applications d'incidence avant et arrière ; ces applications doivent satisfaire la condition suivante :

$$\forall (P_i, T_j) \in P^D \times T^C : Pré (P_i, T_j) = Post (P_i, T_j) ;$$

$$\forall (P_i, T_j) \in P^C \times T^D : Pré (P_i, T_j) = Post (P_i, T_j) = 0 ;$$

- $h : P \cup T \rightarrow \{D, C\}$ est une application qui désigne les nœuds discrets, $h(x) = D$, des nœuds continus, $h(x) = C$;
- $SIM : T^D \rightarrow Q^+ \times (Q^+ \cup \infty)$ est une application qui associe à chaque D-transition T_j son intervalle de franchissement statique $[\alpha_j, \beta_j]$;
- $V : T^C \rightarrow \Re^+$ est une application qui associe à chaque C-transition T_j sa vitesse maximale de franchissement V_j ;
- M_0 est le marquage initial ;

❑

Cette définition impose deux conditions sur les applications d'incidence avant et arrière ; la première condition signifie que si un arc relie une D-place P_i à une C-transition T_j, l'arc réciproque, reliant T_j à P_i doit exister. En d'autre terme, une

marque discrète ne peut être fluidifiée, et tout lien entre une D-place et un C-transition modélise le contrôle effectué par la place discrète sur la vitesse de franchissement de la transition continue. La deuxième condition signifie qu'aucun arc ne doit relier une C-place à une D-transition et réciproquement, c à d que le franchissement d'une D-transition ne dépend que du marquage discret et du temps.

Le comportement d'un système dynamique hybride comporte, en général, un nombre fini ou infini de phases, chaque phase est caractérisée par la fonction suivant laquelle évolue l'état continu du système. La transition entre deux phases change la fonction suivant laquelle évoluent les variables d'états continues, mais la date de cette transition dépend des valeurs de ces variables et son franchissement peut modifier leurs valeurs. Ces notions apparaissent nettement dans le modèle automate hybride que nous avons présenté en détail dans les chapitres précédents.

Dans un modèle RdPH l'état discret du modèle est donné par le marquage du RdP T-temporisé. Pour chaque marquage discret, l'état continu est donné par le marquage des places continues qui évoluent suivant l'équation continue linéaire

$$\dot{M}_C = W_C.v(t)$$

Où M_c est le marquage du RdPCC, W_c est la matrice d'incidence de la partie continue et v(t) est le vecteur des vitesses de franchissement instantanées.

Le franchissement d'une transition discrète change le vecteur des vitesses de franchissement instantanées, où la date de ce franchissement peut dépendre les valeurs du marquage de certaines places continues.

Le comportement du RdPH D-élémentaire que nous venons de présenter, est similaire au RdPH dans le fait qu'un état discret est modélisé par un marquage du RdP discret (RdP T-temporel), mais diffère de lui dans le fait, que le passage à un état discret ne dépend que de l'état discret courant et du temps. Ceci n'est pas une grande restriction, puisque nous aspirons à utiliser ce modèle pour la synthèse de contrôleurs. En effet, il paraît naturel de piloter un système par des décisions discrètes, c'est le cas le plus courant que l'on rencontre dans les systèmes réels. Notre démarche de modélisation consiste d'abord à décrire le comportement du système en boucle ouverte, sans contraintes sur son comportement. Par exemple, on pourra modéliser un système avec une vanne qu'on peut ouvrir et fermer, une machine qu'on peut mettre en marche et à l'arrêt,...etc. La partie contrôle sera prise en compte par la suite, par exemple, une vanne qu'on ferme quand un niveau de liquide atteint le seuil S, ou une machine qu'on met à l'arrêt quand le niveau d'un stock atteint le seuil S'.

L'état d'un RdPH D-élémentaire, à un instant t est un couplage de l'état du RdP T-temporel et l'état du RdPCC. La notion d'IB-état peut être définie pour cette classe de RdPH, sauf qu'un IB-état n'a pas une durée fixe. Deux types d'événements peuvent changer un IB-état du RdPH D-élémentaire, à savoir :

1. Franchissement d'une D-transition T_j, la date de cet événement est comprise entre deux instant α_j et β_j. la date de référence étant l'instant d'activation du RdPH D-élémentaire dans l'IB-état actuel.

2. Le marquage d'une C-place devient nul. Cet événement à une date d'exécution précise, qui dépend de la configuration du RdP T-temporel et du marquage continu à l'instant d'entrée du RdPH D-élémentaire dans l'IB-état courant. Cependant, ce dernier ne correspond pas à un marquage mais à un ensemble de marquage, dû au non déterminisme associé aux franchissements des transitions discrètes.

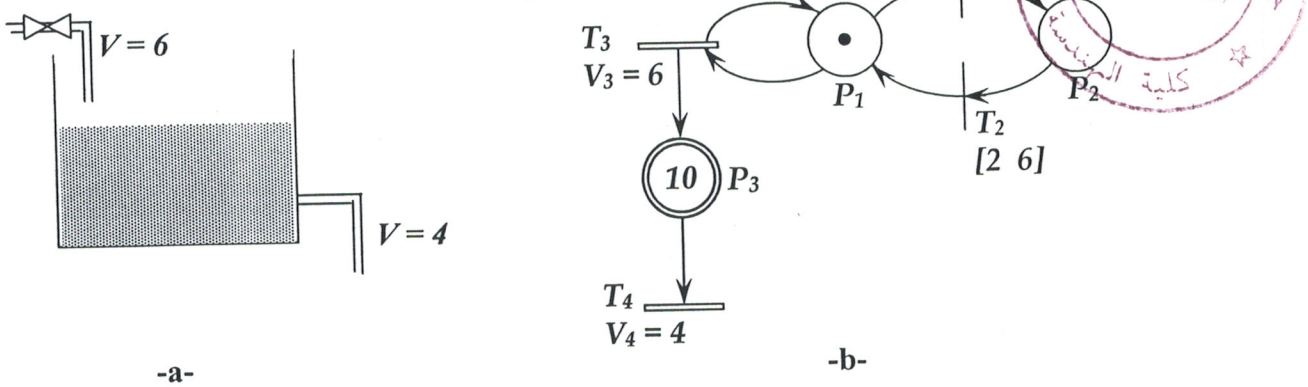


Figure 3.5. –a- Exemple de bac. –b- Modèle RdPH D-élémentaire correspondant

Le graphe d'évolution modélisant le comportement dans le temps d'un RdPH D-élémentaire comporte un nœud pour chaque IB-état du modèle, les transitions entre les nœuds sont étiquetées par les événements et leurs intervalles de dates d'occurrence. Considérons, à titre d'exemple le RdP D-élémentaire de la figure 3.5.b, ce modèle correspond à un réservoir qui peut être rempli ou vidé (Figure 3.5.a), contrôlé par une vanne qui peut s'ouvrir après 2 à 6 u.t. à partir de l'instant de sa dernière action de fermeture et qui peut se fermer après 2 à 4 u.t. de l'instant de sa dernière action d'ouverture. Son graphe d'évolution est représenté en figure 3.6. Dans ce graphe d'évolution à chaque événement correspond un intervalle de dates d'occurrence possibles et non une date précise comme dans le cas des RdPH classiques, par conséquences, on a des intervalles de marquages continus possibles, à l'entrée de chaque IB-état.

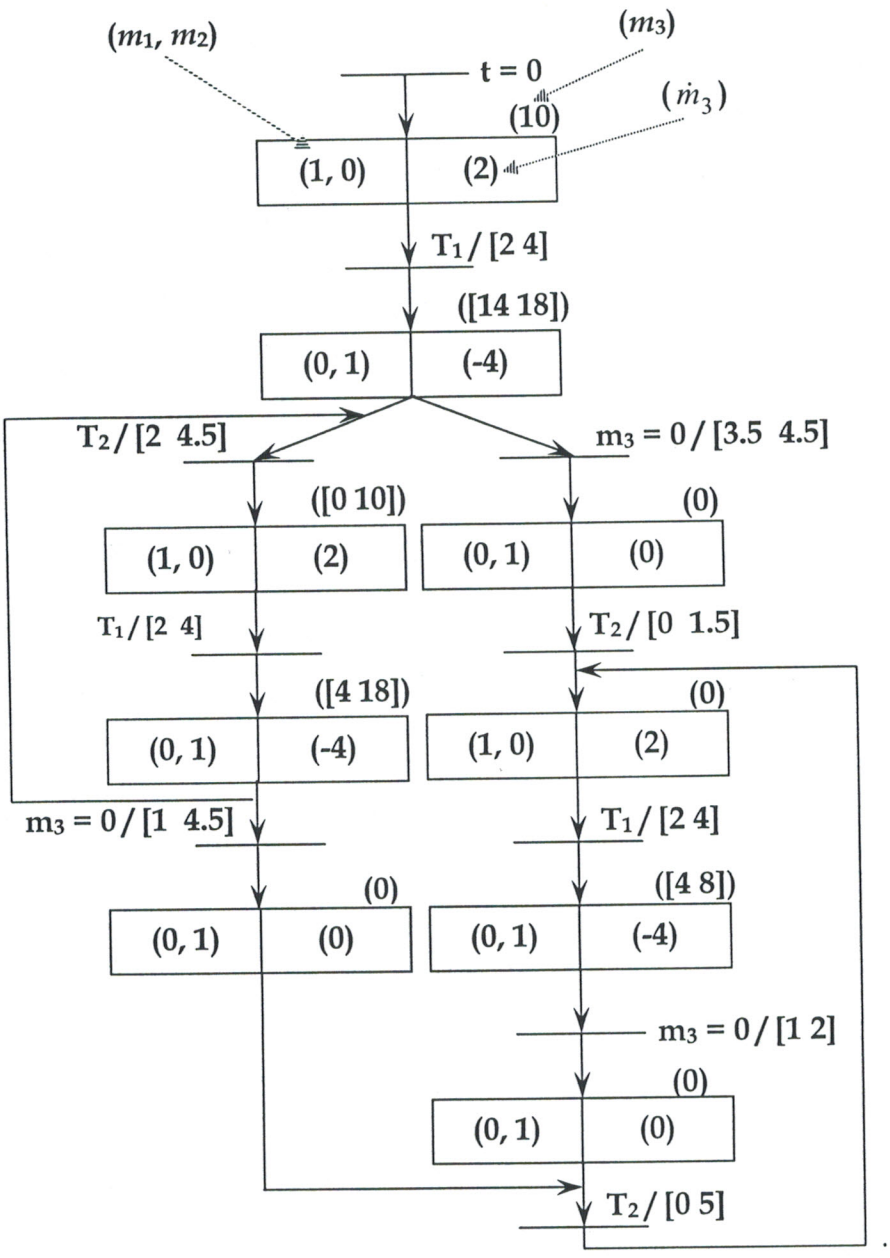


Figure 3.6. Graphe d'évolution du RdPH D-élémentaire de la figure 3.5.b.

Notons que, si dans ce cas particulier, la construction du graphe d'évolution a été possible, à cause de la bornitude du RdP, ceci n'est pas possible dans le cas général.

3.2 Passage des RdP hybride D-élémentaires en automates hybrides

Traduire un RdPH, combinant un RdP T-temporel et un RdP continu, en automate hybride est une opération complexe à cause du couplage important entre les dynamiques des parties discrète et continue. Dans ce travail nous considérons une classe particulière de ce modèle, dans laquelle seule la partie discrète contrôle la dynamique de la partie continue, c à d, la partie continue n'a aucune influence sur le

comportement de la dynamique discrète. C'est le RdP D-élémentaire défini ci-dessus. Ce modèle permet la modélisation d'une classe très fréquemment rencontrée, il s'agit des systèmes à flux continu commandé par des SED.

Cette sous-classe de RdPH offre l'avantage de découpler les dynamiques entre la partie discrète et la partie continue. Ce découplage est vrai dans un seul sens, la dynamique de la partie discrète peut être construite indépendamment de celle de la partie continue, l'inverse n'étant pas vrai. La construction de l'automate hybride modélisant le comportement d'un RdPH D-élémentaire se fait alors d'une manière hiérarchique. D'abord la partie discrète est traduite en automate temporisé, ensuite la partie continue est prise en charge.

3.2.1 *Traduction des RdP T-temporels aux automates temporisés*

Les travaux qui ont été consacrés à la traduction des RdP T-temporels sous leur forme générale (pas nécessairement saufs) aux automates temporisés peuvent être classés en trois catégories suivantes :

- Traduction structurelle.
- Calcul par le graphe des régions.
- Calcul par le graphe des classes d'état.

Dans la suite on va exposer les trois approches existant dans la littérature de traduction des RdP T-temporels en automates temporisés.

3.2.1.1 Traduction structurelle

Cassez et Roux [CR03] ont présenté une procédure structurelle pour systématiser le passage des RdP T-temporels en automate temporisé. Le principe de la procédure est le suivant ; Soit $PN\tau = (P, T, Pré, Post, SIM, M_0)$ un RdP T-temporel avec un ensemble de places $P = \{P_1, P_2, \ldots, P_n\}$ et un ensemble de transitions $T = \{T_1, T_2, \ldots, T_m\}$. Un automate temporisé A_i est construit pour chaque transition T_i du RdP T-temporel (Figure 3.7). Cet automate temporisé possède une seule horloge x_i. Les états de l'automate A_i donne l'état de la transition T_i. Dans l'état T la transition est validée, Dans l'état $\overline{T}$ la transition n'est pas validée et dans l'état F la transition est en train d'être franchie. Cet automate temporisé mis à jour un vecteur M qui représente le marquage du RdP T-temporel et qui est partagé par tout les automates temporisés A_i. L'état initial de chaque automate dépend du marquage initial M_0 du RdP T-temporel.

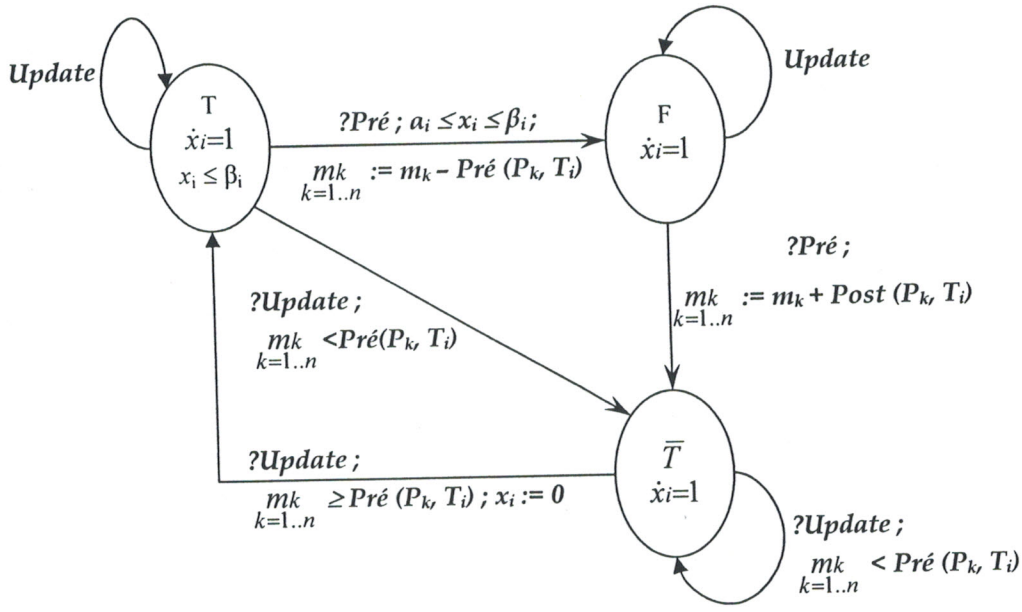


Figure 3.7. Automate A_i associé à une transition T_i du RdP T-temporel.

Les m automates ainsi obtenus sont combinés avec un superviseur (Figure 3.8) en un produit synchronisé d'automates temporisés. Le calcul est très rapide et il est applicable à tous les RdP T-temporel même les non-bornés, ce calcul étant structurel.

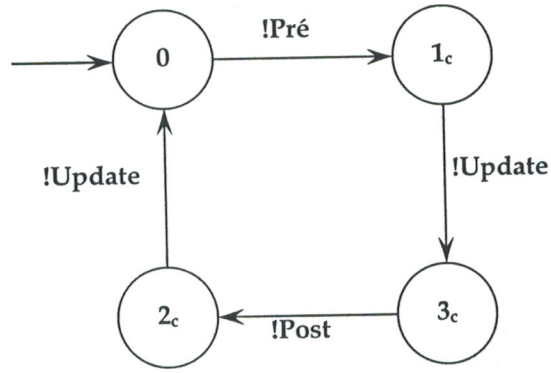


Figure 3.8. Automate du superviseur.

3.2.1.2 Calcul par graphe des régions

Sava et Alla ont développé un algorithme pour la traduction des RdP T-temporels borné en automate temporisé. L'algorithme a pour principe d'associer un sommet L_i de l'automate temporisé à chaque marquage M_i accessible par le RdP T-temporel. L'automate temporisé résultat comporte autant d'horloges que de transitions du RdP T-temporel. L'algorithme est convergent à condition que le RdP T-temporel soit borné et que les bornes des intervalles statiques de franchissement soient des nombres rationnels.

Gardey et Roux proposent une méthode pour appliquer le calcul du graphe des régions des automates temporisés aux réseaux de Petri T-temporels bornés, en

utilisant. Ce calcul peut être utilisé pour générer un automate temporisé des marquages du réseau de Petri en s'inspirant de la méthode de Sava et Alla [Sav01]. Cet automate possède un sommet par marquage et une horloge par transition.

L'analyse sur les automates, obtenus par traduction structurelle et par le calcul du graphe des régions, n'est donc pas très efficace car sa complexité dépend exponentiellement du nombre d'horloges. Dans le premier cas (traduction structurelle) le nombre d'horloge ne peut pas être réduit par des algorithmes comme celui de Daws et Yovine [DY96] à cause de la structure de produit du résultat. Tandis que dans le cas du calcul du graphe des régions, les auteurs ont montré que l'application des algorithmes de réduction du nombre d'horloges de Daws et Yovine permet de réduire sensiblement le nombre d'horloges de l'automate temporisé résultat.

3.2.1.3 Calcul par graphe des classes d'états

Lime et Roux [LR03] ont présenté une extension du graphe des classes d'états [BD91] qui permet d'obtenir un automate temporisé à la place d'un simple graphe. Les auteurs ont mis l'accent sur la minimisation du nombre d'horloge de l'automate temporisé résultat.

Le Graphe des classe d'états, introduit dans [BD91], comporte un ensemble de classes d'état dont chacune est caractérisée par un marquage et un domaine de franchissement. Ce dernier représente les contraintes sur les dates de franchissement des transitions relatives à la date d'entrée dans la classe d'état. Ce qui ne prend pas en considération les valuations des états continus d'un RdP T-temporel. Pour ce faire Lime et Roux ont introduit la notion de classe d'état étendue.

Définition 3.2. (Classe d'état étendue) : Une classe d'état étendue d'un RdP T-temporel est un quadruplet C_E = (M, D, X, Trans) avec :

- M est un marquage ;
- D est un domaine de franchissement ;
- X est un ensemble d'horloges ;
- Trans : X → T est une application qui associe à chaque horloge un ensemble de transitions.

❑

Une classe d'état a été donc étendue par deux composants, X l'ensemble des horloges nécessaires pour exprimer le temps de validation pour chaque transition validée par M. et la fonction Trans désigne pour chaque horloge les transitions dont elle représente la valuation.

Le calcul des successeurs d'une classe d'état étendue diffère de celui d'une classe d'état par les calculs requis pour les deux composantes supplémentaires. Le calcul du

nouveau marquage et celui du nouveau domaine de franchissement restent inchangés.

Soit une classe d'états étendue C = (M, D, X, Trans) et une transition franchissable T_j. Pour calculer la classe d'états étendue C' = (M', D', X', Trans') obtenue par le franchissement de T_j depuis C, nous avons les étapes supplémentaires suivantes :

1. Pour chaque horloge x de X, les transitions dé-validée par le franchissement de T_j sont retirées de Trans (x). ;
2. Les horloges dont l'image par la fonction Trans est vide sont retirées de X ;
3. S'il y a des transitions nouvellement validées par le franchissement de T_j , deux cas sont possibles :
   - il existe une horloge x dont la valeur est 0. Alors, les transitions nouvellement validées sont ajoutées à trans(x),
   - il n'existe pas d'horloge nulle. Alors nous créons une nouvelle horloge x_i associée aux transitions nouvellement validées. L'indice i choisi est le plus petit entier qui n'est pas dans les indices de horloges de X. On ajoute x_i à X et trans(x_i) est l'ensemble des transitions nouvellement validée.

Le calcul de l'automate temporisé des classes d'états se fait en deux étapes. Dans la première on calcule un graphe des classes d'états étendues de façon assez similaire au graphe des classes d'états classiques. Puis on déduit syntaxiquement l'automate temporisé de ce graphe.

La classe d'états étendue initiale est C_0 = (M_0, D_0, X_0, $Trans_0$), où M_0 est le marquage initial, $D_0 = \{\alpha i \leq ti \leq \beta i \ / \ Ti \in T^{M0}\}$, $X_0 = \{x_0\}$, $Trans_0 = (x_0, TM_0)$.

Le calcul des successeurs est itéré depuis C_0, comme pour le graphe des classes d'états classiques.

3.2.2 *Principe de l'algorithme de traduction des RdPH D-élémentaires en automates hybrides*

Comme mentionné ci-dessus, dans un RdPH D-élémentaire, la partie discrète a un comportement indépendant de la partie continue, son évolution ne dépend que de son marquage initial et de la variable continue indépendante qui est le temps. Elle peut donc être étudiée seule. Pour chaque marquage accessible par le RdP T-temporel, correspondra une configuration du RdPCC. Le modèle RdPH D-élémentaire peut donc être étudié d'une manière hiérarchique, d'abord la partie discrète est considéré, ensuite, pour chaque marquage accessible de cette dernière, la configuration continue est étudiée. L'algorithme de traduction de traduction des RdPH D-élémentaires en automate hybride que nous proposons dans ce mémoire comporte trois étapes.

1. *Isoler la partie RdP T-temporel du molèle hybride est construire son automate temporisé équivalent*

Dans le modèle hybride, les parties discrète et continue sont reliées via des boucles (deux arcs de même poids dans les deux sens) connectant certaines D-places à certaines C-transition. Ces boucles constituent les limites entre le RdP T-temporel et le RdPCC. En cassant ce lien, on aura les deux parties discrète et continue séparées.

Cette première étape de l'algorithme de traduction consiste à traduire le RdP T-temporel en automate temporisé. Comme présenté au chapitre précédent, plusieurs travaux ont été consacré à la traduction des RdP T-temporels en automate temporisés [SA01], [LR03] et [CR03]. Ces travaux ont comme but la vérification des propriétés temporelles des RdP T-temporels. La plupart d'entre eux ne sont applicables qu'aux RdP T-temporels bornés [SA01] et [LR03]. Puisque, ce n'est que pour cette classe des RdP T-temporel que l'algorithme de traduction converge, c à d l'automate résultat a un nombre fini de sommets.

Cette étape étant déjà était résolue, nous réutilisons les résultats existants.

Considérons à titre d'illustration le RdPH D-élémentaire de la figure 3.9. La limite entre le RdP T-temporel et le RdPCC est schématisée sur la figure par la ligne en pointillés.

La figure 3.10 représente la partie discrète du RdPH D-élémentaire de la figure 3.9 ainsi que son automate temporisé équivalent, et qui est construit d'une manière intuitive.

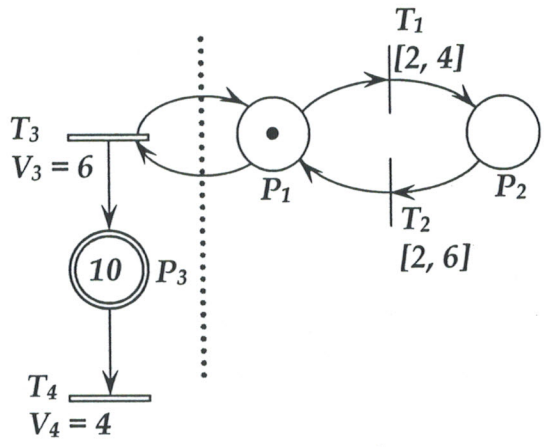


Figure 3.9. Limites entre les parties discrète et continue dans un RdPH D-élémentaire.

2. ***Construire l'automate hybride correspondant à chaque sommet de l'automate temporisé résultat de l'étape précédente***

Dans la suite du document nous appellerons macro-sommets, les sommets de l'automate temporisé qui résulte de l'étape précédente, puisque chacun d'eux va correspondre à plusieurs sommets dans l'automate hybride final.

Chaque macro-sommet correspond à un marquage du RdP T-temporel, et donc à une configuration du RdPCC, puisque dépendant du marquage des D-places. On peut imaginer la construction de l'évolution du RdPCC indépendamment de celle du modèle discret. Ensuite, on ajoute l'influence de dernier. Celle-ci se traduira par le fait que certaines C-transitions peuvent être franchissables ou infranchissables selon l'état de la partie discrète. Une transition infranchissable sera éliminée de l'évolution du RdPCC.

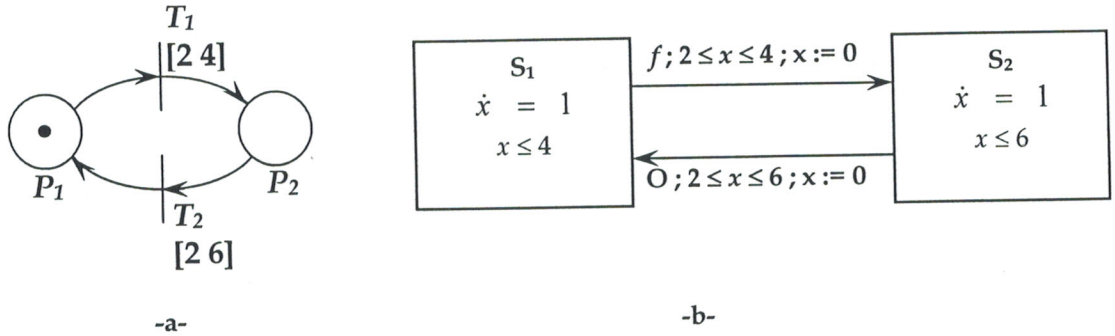


Figure 3.10. –a- RdP T-temporel correspondant au modèle hybride de la figure 3.9. –b- Automate temporisé correspondant.

Revenons à l'exemple du bac (figure 3.9), la figure 3.11 représente la configuration du RdPCC pour les deux marquages accessibles par le RdP T-temporel. Pour $[m_1\ m_2]^T = [1\ 0]^T$, Le modèle hybride se réduit au RdPCC de la figure 3.11.a, et il se réduit au RdPCC de la figure 3.11.b pour $[m_1\ m_2]^T = [0\ 1]^T$.

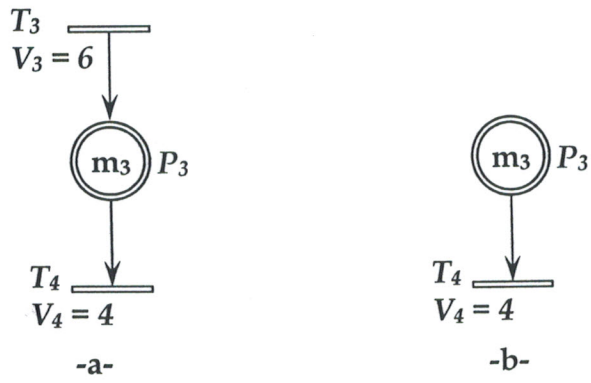


Figure 3.11. -a- RdPCC correspondant au marquage discret $[1\ 0]^T$. *-b- RdPCC correspondant au marquage discret* $[0\ 1]^T$.

Le but de cette étape est de construire les automates hybrides modélisant les comportements des RdPCC correspondant à chaque macro-sommet. L'évolution d'un RdPCC est généralement modélisée par son graphe d'évolution, comme expliqué au paragraphe 2.2.3.2, le graphe d'évolution d'un RdPCC est assimilable à un automate hybride linéaire, dans lequel les variables d'état correspondent au marquage des C-places et les transitions modélisent l'événement, marquage d'une C-place s'annule. La difficulté dans la construction des automates hybrides correspondants au RdPCC, dans notre cas, réside dans le fait que le marquage initial ne correspond pas à une valeur fixe mais à un ensemble souvent infini de valeurs, à cause de l'incertitude dans les dates de commutation entre deux macro-sommets. Pour cette raison, nous allons procéder à une traduction structurelle des RdPCC en automates hybrides, et ceci on supposant que le marquage initial du RdPCC correspond à un macro-marquage dont le support est tout P^C. La notion de macro-marquage, présentée au paragraphe 2.2.3.1, à été introduite par David et Alla [DA05] dans le but de représenter d'une manière finie le nombre de marquages infini accessibles par un RdPC autonome. Pour un RdPC, autonome ou non, à n places, le nombre maximum de macro-marquages accessibles est 2^n, ceci est dû à la nature booléenne de l'état d'une place dans un macro-marquage.

Propriété : Pour un RdPCC, le nombre de macro-marquages atteignables est maximal pour le macro-marquage initial qui a pour support P (l'ensemble de toutes les C-places).

❑

Sachant que le seul événement pouvant changer l'état d'un RdPCC est que le marquage d'une C-place s'annule, l'occurrence de cet événement n'est possible que pour les C-place marquées et dont le marquage est décroissant. Un macro-marquage m^* dans lequel la C-place Pi n'est pas marquée, est atteignable depuis le macro-marquage m_0^*, dans lequel P_i est marquée. Le contraire n'est pas possible.

Pour cette raison et pour prendre en compte tous les marquages accessibles par les RdPCC, nous allons considérer que le macro-marquage initial a pour support P^C et construire l'automate hybride modélisant le comportement de chaque RdPCC sans considération pour son marquage initial. Pour ce faire, nous procédons comme suit :

i. Initialement $m_0^* = P^C$, ce qui signifie que toutes les places sont initialement marquées et par conséquent toutes les transitions sont franchies à leurs vitesses maximales.

Créer le sommet initial de l'automate hybride et lui associer l'activité

$$\dot{M} = W \cdot V$$

Avec W la matrice d'incidence du RdPCC et V le vecteur des vitesses de franchissement maximales.

ii. Créer depuis le sommet initial une transition pour chaque place P_i dont le marquage est décroissant, et lui affecter la garde $m_i = 0$. Après cette étape on aura une transition pour chaque place dont le marquage est décroissant.

iii. Pour chaque transition T_j construite dans l'étape ii. créer un sommet destination et lui affecter l'activité :

$$\dot{M} = W \cdot v(t)$$

Avec $v(t)$ le vecteur des vitesses de franchissement instantanées [DA92]. Fusionner les sommets ayant la même activité.

iv. Vérifier si pour chaque place P_i dont le marquage est initialement décroissant, le marquage a atteint un bilan nul ($\dot{m}i = 0$). Si oui arrêter l'algorithme et si non refaire l'étape ii. pour chaque sommet criés en étape iii.

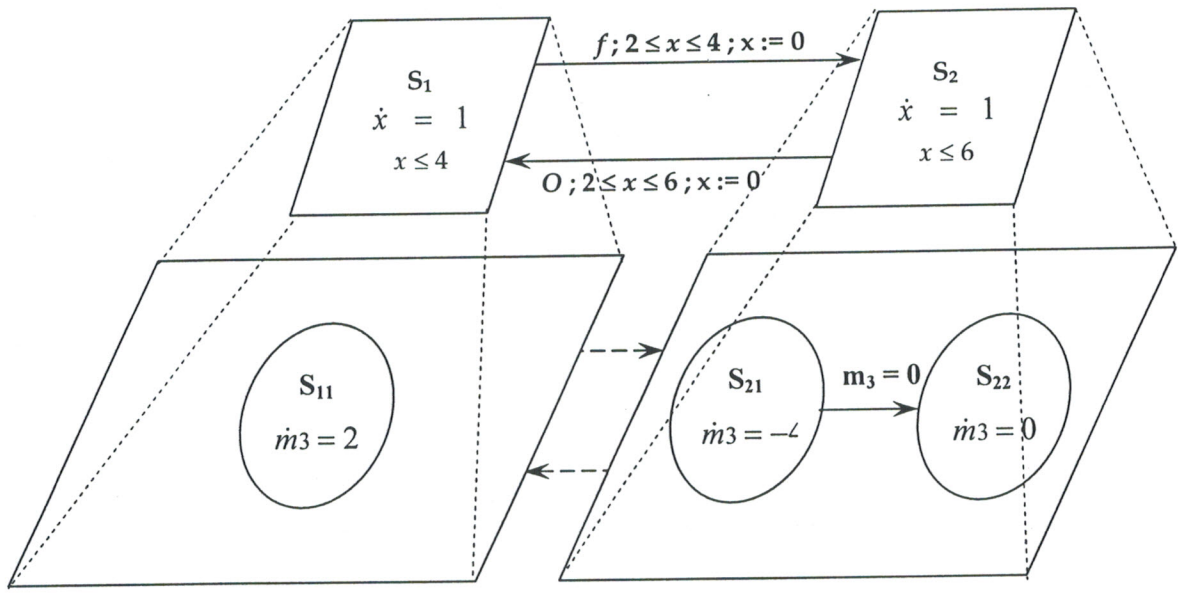


Figure 3.12. Forme hiérarchique de l'automate hybride après la deuxième étape de l'algorithme.

Si on considère à nouveau le RdPH D-élémentaire de la figure 3.9, modèle du système de bac (Figure 3.5), après cette deuxième étape, on aura une forme hiérarchique de l'automate hybride, comme schématisée sur la figure 3.12. La figure 3.11.a représente la configuration de la partie continue correspondante au marquage $[1\ 0]^T$, Ce RdPCC à une seule C-place dont le bilan de marquage est croissant, elle est donc modélisée par un automate hybride possédant un seul sommet. La deuxième configuration de la partie continue, est donnée par la figure 3.11.b, et correspond au marquage discret $[0\ 1]^T$. Pour ce RdPCC le marquage de la place P_3 est décroissant et peut donc s'annuler. Ce RdPCC est équivalent à un automate hybride avec deux sommets (Figure 3.12).

3. *Remplacer les transitions entre les macro- sommets par des transitions entre les sommets internes*

Après l'application de la deuxième étape de l'algorithme de traduction des RdPH D-élémentaires en automate hybrides, on obtient une forme hiérarchique d'un automate hybride comportant des macro–sommets et chacun de ces derniers comporte un automate hybride qui décrit la dynamique continue correspondant au macro-sommet.

Les deux événements susceptibles de changer l'état d'un RdPH D-élémentaires et qui sont : la franchissement d'une D-transition ou le marquage d'une D-place qui s'annule, correspondent respectivement, dans la forme hiérarchique de l'automate hybride que nous avons obtenu, aux transitions entre les macro-sommets et aux transition entre les sommets internes aux macro-sommets. Dans une dernière étape, nous remplaçons les transitions entre les macro-sommets par des transitions entre leurs sommets internes.

Pour cela, considérons la notion de macro-marquages, un macro-marquage d'un RdPCC de n places, peut être caractérisé un vecteur booléen $\overline{M}$ de dimension n, avec :

$$\overline{M_i} = \begin{cases} 1 \ si \ P_i \ est \ marqué \\ 0 \ si \ non \end{cases}$$

Cette étape consiste à remplacer chaque transition entre deux macro-sommets par des transitions entre tous les sommets internes, à condition que le vecteur caractéristique du sommet source soit inférieur ou égal (composante à composante) au vecteur caractéristique du sommet but. Ceci signifie que franchir une transition discrète dans un RdPH D-élémentaire, peut marquer une place vide, mais ne peut pas annuler le marquage d'une place marquée.

La figure 3.13 modélise l'automate hybride résultat de l'application de l'algorithme de traduction au RdPH D-élémentaire de la figure 3.9. Dans la forme hiérarchique de l'automate hybride, résultat de l'étape 2 de l'algorithme, nous avons deux transitions entre les macro-sommets étiquetées respectivement *f* et *O*. La transition f est remplacée dans l'automate hybride final par une seule transition reliant les sommets S_{11} et S_{21}, tandis que la transition *O*, est remplacée par deux transition O_1 et O_2.

Dans la seconde étape de l'algorithme, nous avons construit un automate hybride équivalent au comportement du RdPCC correspondant à chaque macro-sommet, et ceci sans considération pour le marquage initial du RdPCC à l'entrée du macro sommet. Ceci peut engendrer des sommets dans l'automate final qui ne seront jamais visités. Ce modèle constitue une sur-approximation du modèle exact dans la

mesure où il contient toutes les trajectoires réelles ainsi que d'autres qui ne sont réalisables. Pour éliminer les sommets non atteignables ainsi que les transitions associées, nous allons analyser l'automate résultat par l'application HyTech que nous présenterons en détail dans le chapitre suivant.

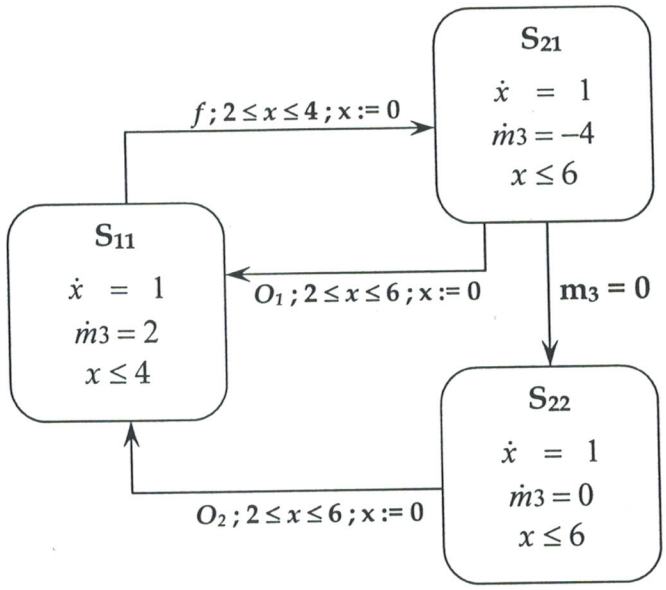


Figure 3.13. Automate hybride résultat de l'application de l'algorithme de traduction au RdPH D-élémentaire de la figure 3.9.

3.2.3 *Analyse de l'algorithme de traduction*

L'algorithme de traduction des RdPH D-élémentaires que nous avons proposé, intègres deux étapes importantes, à savoir la traduction des RdP T-temporels en automates temporisés et la traduction des RdPCC en automates hybrides. Pour résoudre la première étape, plusieurs algorithmes ont été proposés dans la littérature, pour la plupart d'entre eux [Lim04], [SA01] la convergence est conditionnée par la bornitude du RdP T-temporel. Quoique cette dernière a été prouvée être indécidable pour cette classe des RdP, des conditions suffisantes restrictives ont été proposées par Berthomieu et Diaz dans [BD91] pour la vérification. Tandis que la traduction du RdPCC en automate hybride converge toujours, même si le RdPCC n'est pas borné. Ceci est dû au fait qu'on caractérise une place non pas par son marquage mais par son macro-marquage qui est une grandeur booléenne. Un RdPCC à n places peut être décrit par un automate hybride dont le nombre de sommets est inférieur ou égale à 2^n.

Le nombre de sommets de l'automate hybride résultat de l'algorithme proposé, dépend donc de deux paramètres, il s'agit du nombre de macro-sommets de l'automate temporisé décrivant le comportement de la partie discrète, ce paramètre est noté N, et du nombre de C-places du RdPH D-élémentaire, qu'on a noté n'. On a donc un automate hybride dont le nombre de sommets est inférieur ou égale à $N.2^{n'}$.

Les variables d'état continues sont de deux types, les horloges qui permettent de compter le temps depuis l'instant de la dernière validation d'une D-transition, et les variables continues linéaires qui modélisent le marquage des C-places. Ces deux types de variables sont linéaires, L'automate résultat est donc linéaire.

Conclusion

Dans ce chapitre nous avons présenté un modèle RdP hybride couplant un RdP discret T-temporel et un RdPCC. Ce modèle dit RdPH D-élémentaire a pour but de modéliser les systèmes à flux continu supervisés par des systèmes à évènements discrets. Sous le nom système à flux continu, nous regroupons les systèmes continus linéaires ainsi que systèmes à évènements discrets impliquant un grand nombre de pièces.

Un algorithme qui permet la traduction des RdPH D-élémentaires a été proposé. L'idée de traduire les RdP en automates est déjà apparue dans plusieurs travaux, son but est d'associer la puissance de modélisation des RdP à la puissance d'analyse des automates. Dans ce même contexte, la traduction des RdPH D-élémentaires en automates hybrides a pour but d'utiliser l'automate hybride résultat pour la synthèse de contrôleurs pour la classe des systèmes hybrides considérée.

Dans le chapitre suivant nous allons utiliser l'application HyTech pour analyser l'automate hybride résultat, cette étape nous permet d'avoir l'automate exact équivalent au RdPH D-élémentaire.

Le Modèle Automate Hybride Atteignable

Dans le chapitre précédent un algorithme structurel et hiérarchique pour la traduction des RdPH D-élémentaires en automates hybrides a été présenté. L'automate hybride résultant peut contenir des sommets qui ne seront jamais visités. Pour cette raison nous allons utiliser l'application HyTech pour calculer l'automate hybride atteignable par l'automate résultat. Dans ce chapitre nous allons donner quelques méthodes utilisées pour le calcul de l'espace atteignable par un automate hybride. Nous expliquons ensuite comment utiliser l'application HyTech pour effectuer ce calcul.

4.1 Introduction

Dans un RdPH D-élémentaire, la date de franchissement d'une transition discrète prend ses valeurs dans un intervalle de temps dense. Ceci a pour conséquence, que le marquage de chaque C-place, après ce franchissement, correspond à un intervalle de valeurs possibles. C'est pour cette raison que nous avons procédé à une traduction structurelle de la partie continue dans l'algorithme de traduction des RdPH D-élémentaires en automates hybrides, présenté au chapitre précédent. Nous avons supposé que l'état initial de la partie continue correspond à un macro-marquage dont le support est P (l'ensemble des places), c à d, que toutes les C-places sont supposées marquées à l'instant de commutation entre deux macro-sommets. Dans un RdPCC, un IB-état peut être caractérisé soit par le vecteur des dérivées des marquages des C-places, soit par le vecteur des vitesses de franchissement instantanées des C-transitions et soit par le macro-marquage, et c'est le macro marquage dont le support est P qui permet d'engendrer tous les autres, c'est pour cette raison que nous avons choisi de considérer un macro-marquage de support P à l'entrée de chaque macro-sommet. Cependant ceci va donner une sur-approximation du modèle exact.

Considérons à titre d'exemple les RdPH D-élémentaires en figures 4.1.a et 4.2.a, ces deux modèles ont la même structure et ne diffère que dans les intervalles de franchissement associés aux D-transitions. L'application de traduction des RdPH D-élémentaire en automates hybrides, sur ces deux modèles, donne respectivement les automates hybrides en figures 4.1.b et 4.2.b. On peut facilement vérifier que dans

l'automate de la figure 4.1.b, le sommet S_{22} ne sera jamais visité par l'automate, et les transitions V et O_2 ne seront jamais franchises.

Pour éliminer les sommets non visités par l'automate hybride nous allons déterminer l'automate hybride atteignable par l'application HyTech.

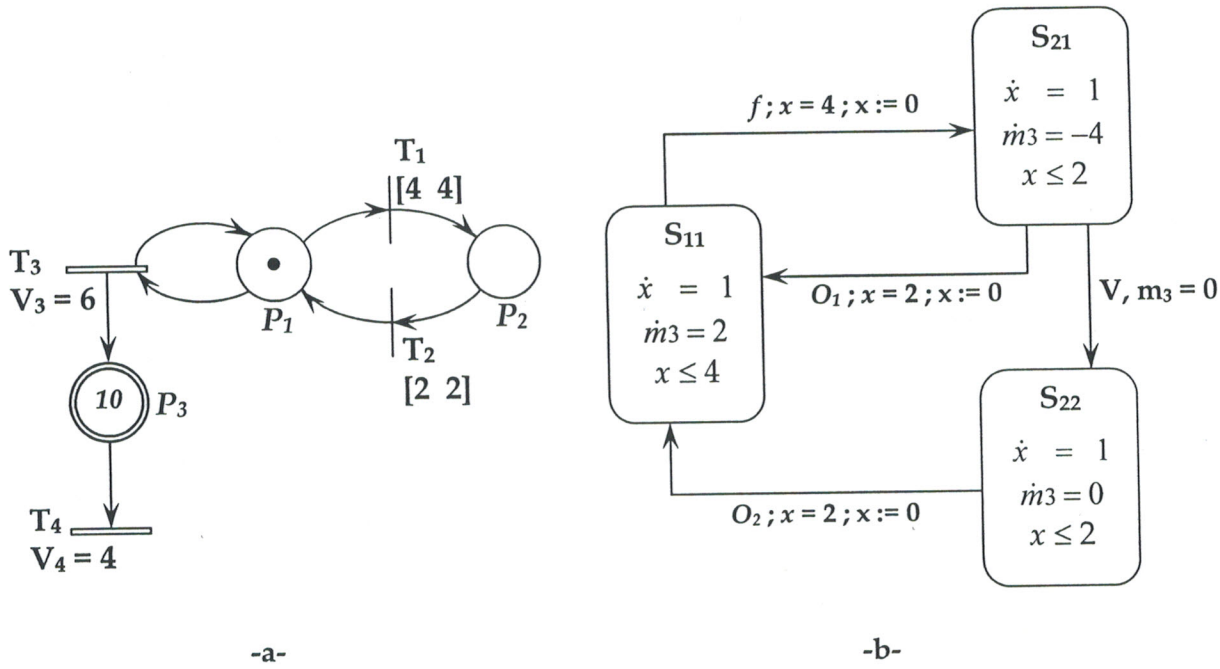


Figure 4.1 –a- RdPH D-élémentaire. –b- automate hybride résultat de l'application de l'algorithme de traduction.

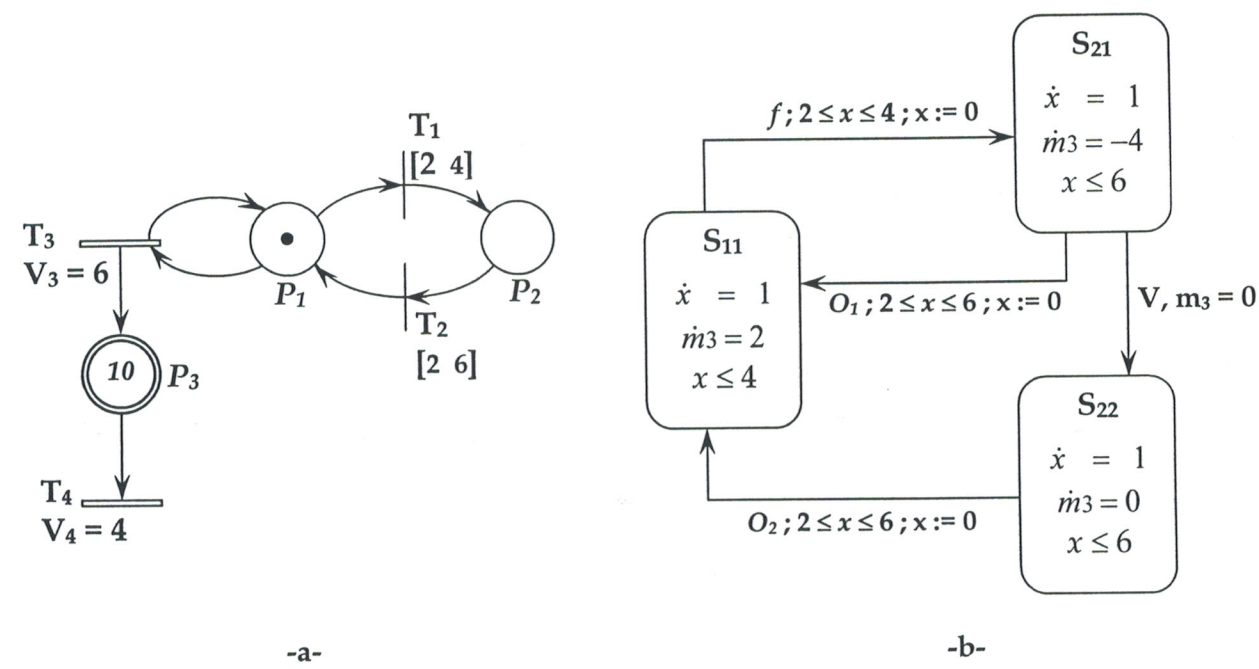


Figure 4.2 –a- RdPH D-élémentaire. –b- automate hybride résultat de l'application de l'algorithme de traduction.

4.2 Calcul de l'automate atteignable

L'analyse des automates hybrides est généralement basée sur le calcul de l'espace d'état atteignable par cet automate. Considérons un automate hybride $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$ pour lequel on cherche à déterminer l'ensemble d'états atteignables depuis une région (q, I) caractérisé par un sommet q et une région I dans laquelle se trouve l'état continu. On a vu que les trajectoires possibles pour un automate hybride sont constituées chacune par une succession de transitions continues pendant lesquelles le système évolue selon la dynamique continue du sommet courant, définie par F, et de transitions discrètes correspondant aux franchissements de transitions appartenant à T. un état d'un automate hybride peut avoir donc, deux types de successeurs, des successeurs discrets et des successeurs continus.

Un successeur discret d'un état est obtenu en franchissant une transition de l'automate hybride.

Définition 4.1 (successeur discret d'un état) : soit $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$ un automate hybride et T = (q, σ, g, A, q') une transition de T. l'état (q', v') est un successeur discret de l'état (q, v) par le franchissement de T si :

$$v \parallel g \text{ et } A(v) = v' \text{ et } v' \parallel Inv(q')$$

❑

Ce qui signifie qu'une transition ne peut être franchie que si sa garde est vérifiée par les valeurs des variables continues ($v \parallel g$). Lors du franchissement de la transition T, les valeurs des variables continues sont modifiées selon l'affectation associée à T ($A(v) = v'$). De plus les valeurs des variables continues après le franchissement de T doivent satisfaire l'invariant but ($v' \parallel Inv(q')$).

Un successeur continu d'un état est obtenu en restant dans le même sommet et en laissant le temps s'écouler.

Définition 4.2 (successeur continu d'un état) : soit $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$ un automate hybride. L'état (q, v+t) est un successeur continu de l'état (q, v) si :

$$\exists\, t \in \Re^+ \text{ tel que } \forall\, t' \leq t, f_q(v+t') \parallel Inv(q)$$

❑

Ce qui veut dire que les variables continues évoluent suivant l'activité du sommet courant tout en restant dans ce sommet. L'invariant de ce dernier doit être vérifié par les valeurs des variables continues, tout au long de la période de séjour dans ce sommet.

De la même manière que pour les états on peut définir les successeurs continus et discrets d'une région qui est un ensemble d'état dans un même sommet.

Définition 4.3 (successeur discret d'une région) : soit $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$ un automate hybride, T = (q, σ, g, A, q') une transition de **T** et (q, I) une région. L'ensemble des états atteignables depuis tout état (q, v) dans la région (q, I) en franchissant T est dit successeur discret de la région (q, I) et est noté $Post_T$ (I).

$$v' \in Post_T (I) \text{ si } \exists\, v \in I \text{ tel que } v \parallel g \text{ et } v' = A(v) \text{ et } v' \parallel Inv (q')$$

❑

Exemple 4.1. Considérons l'automate hybride de la figure 4.3, et soit (q_1, I_1) une région dans le sommet q_1. L'espace I_1 (figure 4.4.a) est décrite par les contraintes suivantes sur les valeurs des horloges.

$$0 \le x\text{-}y \le 3 \text{ et } 1 \le y \le 2$$

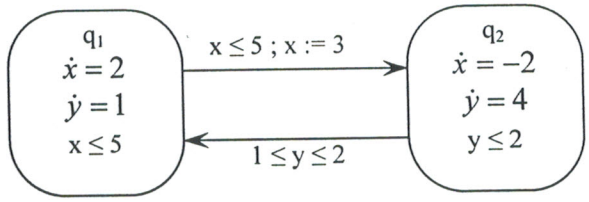


Figure 4.3. Automate hybride.

La région (q_2, $post_T(I_1)$) qui est le successeur discret de la région (q_1, I_1) par le franchissement de T = (q1, x=5, x :=3, q2) est schématisé en figure 4.4.b. Pour que le franchissement de T soit possible, il faut que sa garde (x ≤ 5) soit vérifiée. Lors du franchissement de cette transition la valeur de la variable y reste inchangée, tandis que la valeur de x est réinitialisé à 3.

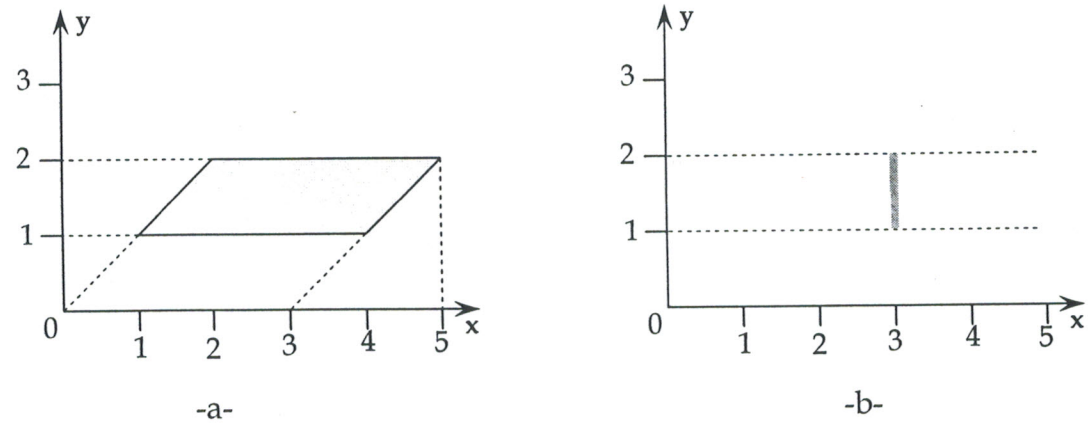


Figure 4.4. –a- région de l'espace d'état continu. –b- son successeur discret.

Définition 4.3 (successeur continu d'une région) : soit $A_H = (Q, \Sigma, q_0, X, Inv, T, F)$ un automate hybride. L'ensemble des états atteignables à partir d'un état à partir de tout état (q, v) dans la région (q, I) en laissant le temps s'écouler tout en restant dans le même sommet est dit successeur continu de la région (q, I). Cet ensemble donne la région $(q, Post_t(I))$.

$$v' \in Post_t(I) \text{ si } \exists\, v \in I,\ t \in \Re^+ \text{ tel que } \forall\, t' \leq t,\ f_q(v+t') \parallel Inv(q)$$

❑

Exemple 4.2. Considérons à nouveau l'automate hybride de la figure de la figure 4.3. Soit (q_1, I_2) une région dans le sommet q_1 dont l'espace d'état continu I_2 (figure 4.5.a) correspond à :

$$I_2 = 1 \leq x \leq 3 \text{ et } 0 \leq y \leq 1$$

Le successeur continu de la région (q_1, I_2) est schématisé sur la figure 4.5.b. La variable x évolue suivant l'équation $\dot{x} = 2$, et la variable y suivant l'équation $\dot{y} = 1$. Le système peut séjourner dans le sommet q_1 tant que son invariant ($x \leq 5$) est satisfait par les variables d'état.

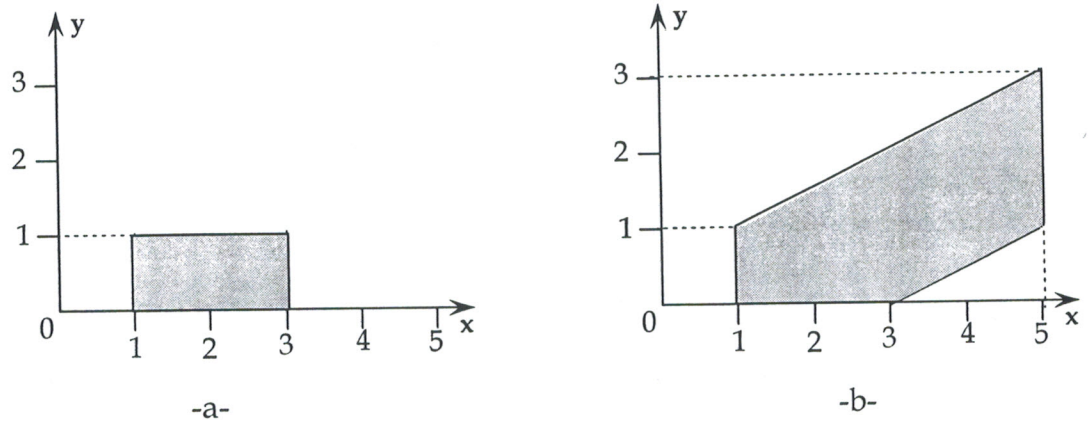


Figure 4.5. -a- Région de l'espace d'état continu. -b- Son successeur continu.

Le calcul de successeurs discrets et continus de région est itéré pour obtenir l'espace d'états atteignable à partir d'une région donnée.

Pour savoir si la région (q', I') est atteignable depuis la région (q, I) deux méthodes peuvent être utilisées. La première est basée sur le calcul de l'espace de tous les états qui peuvent être atteints depuis des états appartenant à la région (q, I). Cet espace est dit successeur de la région (q, I), s'il contient des états appartenant à la région (q', I'), on peut conclure que cette région est atteignable depuis (q, I). Cette méthode est dite méthode d'analyse en avant.

La deuxième méthode est basée sur le calcul de l'ensemble de tous les états à partir desquels on peut atteindre des états de la région (q', I'). L'ensemble de ces états est appelé prédécesseur de la région (q', I'). Si l'espace calculé contient des états qui appartiennent également à la région (q, I), alors on peut conclure que la région (q', I') est atteignable depuis (q, I). Cette méthode, duale à la méthode d'analyse en avant, est appelée méthode d'analyse en arrière.

Ces deux procédures de calculs sont complémentaires et il n'est pas possible de prévoir laquelle sera la plus efficace dans un cas particulier.

La procédure d'analyse en avant de l'espace d'état atteignable par A_H est basé sur la succession de calcul suivante.

A partir de la région (q, I)

1. Calculer la région (q, Post_t (I)) = Inv (q) $\cap f_q$ (I) ;

L'intersection de l'invariant du sommet q, Inv (q), et de l'extension de I par F (q) = f_q donne l'ensemble des points accessibles à partir d'un point de I par la dynamique continue f_q. L'espace d'état représenté par cet intersection est noté I'.

2. Pour chaque transition T = (q, σ, g, A, q') de T dont q est le sommet source, calculer l'ensemble de points Post_T (I') comme suit.

   i. Calculer I_1, intersection de I' avec l'ensemble défini par la garde de la transition T considérée.

   ii. Calculer I_2, image de I_1 par l'affectation A de la transition T.

   iii. Calculer I_3, intersection de I_2 avec l'invariant de q', Inv (q').

   iv. Calculer I_4, intersection de l'extension de I_3 par F(q') = $f_{q'}$ la dynamique du sommet q' avec son invariant Inv (q').

3. On obtient ainsi un ensemble de régions de la forme (q', I_4) que l'on ajoute à l'espace atteignable, et à partir desquels on relance l'étape 2. pour calculer Post (q', I_4).

Cette suite de calcul d'ensembles obtenus en suivant les dynamiques continues et discrètes est réitérée jusqu'à la convergence, c à d, jusqu'à ce que l'ensemble des régions n'évolue plus. On obtient ainsi l'espace d'état atteignable.

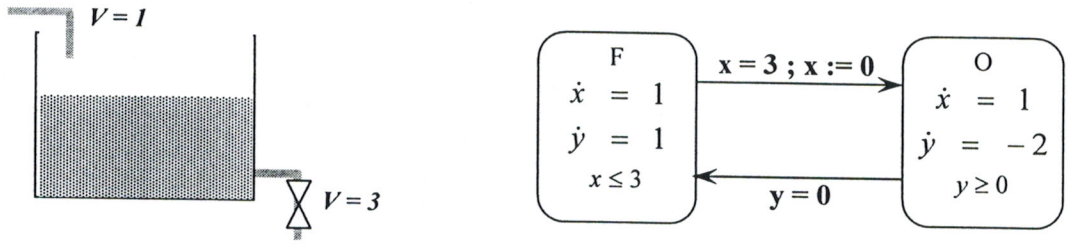


Figure 4.6. –a- Réservoir d'eau. –b- automate hybride associé au réservoir d'eau.

Pour illustrer cette procédure de calcul Guéguen et Zaytoon [GZ01] ont utilisé un automate hybride simple (figure 4.6.b) correspondant au modèle du réservoir en figure 4.6.a. Pour savoir si la région S_F = (F, $1 \leq x \leq 3$ et $x = y+1$) est atteignable depuis la région S_0 = (F, $x = y$ et $x \leq 3$) (figure 4.7.), la première itération de calcul de la procédure d'analyse en avant est la suivante :

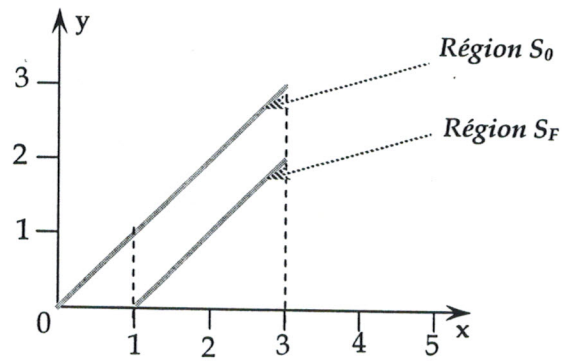


Figure 4.7. La région S_F est il atteignable depuis la région S_0 ?

Le région initiale est (F, I) avec I = ($x = y$ et $x \leq 3$)

1. I' = *Inv* (F) $\cap$ f_F (I) = ($x = y$ et $x \leq 3$)
2. Calcul de Post (F, I') :

   i. I_1 = ($x = y = 3$), puisque il y a une seule transition franchissable dont la garde est ($x = 3$)

      i. $I_2 = I_3$ = ($x = 0$ et $y = 3$). Etant donné que l'affectation de la transition est $x := 0$ et que l'invariant du sommet O est ($y \geq 0$).

      ii. I_4 = ($y + 2x = 3$ et $y \geq 3$), défini par la dynamique du sommet O.

L'espace atteignable après cette première itération est $S_1 = S_0 \cup$ (O, $y + 2x = 3$ et $y \geq 3$).

Dans ce cas, le calcul de l'espace atteignable par la procédure d'analyse en avant ne converge pas, par contre, le calcul inverse par la procédure d'analyse en arrière à partir de S_F = (F, $1 \leq x \leq 3$ et $x = y+1$) permet de conclure en deux itérations que les régions qui permettent d'accéder à S_F sont lui même et (O, $y+2x = 2$ et $y \geq 0$). Comme la région S_0 à une intersection vide avec ces deux régions, il est possible de conclure qu'il n'est pas possible d'atteindre S_F à partir de S_0. Ce petit exemple montre que même pour des modèles simple le calcul de l'espace atteignable peut ne pas converger.

Les problèmes du calcul de l'espace atteignable et de l'existence d'un algorithme effectuant ce calcul sont très importants pour l'analyse, la vérification et la commande des systèmes hybrides. Malheureusement d'un point de vue théorique il est prouvé que, en dehors de quelques cas particulier [HKP+95], ce problème n'est pas décidable. Les cas particulier où la convergence de l'algorithme peut être garanti appartiennent principalement à la classe des automates hybrides linéaires.

L'intérêt de cette classe des automates hybrides est que toutes les régions de l'espace d'état continu qu'ils définissent (invariants, gardes) sont des régions linéaires convexes et que les images des régions linéaires par leurs fonctions d'évolution continues et discrètes sont également des fonction linéaires [HR98].

Le calcul de l'espace atteignable dans un automate hybride nécessite de pouvoir :

- Représenter des régions de l'espace d'état continu ;
- Calculer des intersections et des compléments de telles régions ;
- Déterminer que des régions sont vides ;
- Calculer l'image ou l'image inverse de telles régions par les dynamiques continues ;

Dans le cas des automates hybrides linéaires ces différentes régions sont des polyèdres, et sont donc relativement facile à manipuler. Par conséquence il est possible de développer des outils utilisant des bibliothèques de calcul sur les polyèdres pour effectuer l'analyse. L'outil HyTech (Hybrid technology tool) [HHW95b] permet la vérification de modèles obtenus par la composition d'automates hybrides, en s'appuyant sur la bibliothèque de Halbwachs [Hal93]. HyTech permet la manipulation de modèles incomplètement spécifiés dans lesquels certaines valeurs sont définie comme des paramètres, ce qui luis permet d'apporter une assistance à la conception en précisant les conditions sur ces paramètres qui assurent que telle propriété est vérifiée.

4.3 Mise en œuvre par l'application HyTech

HyTech est une application destinée à l'analyse des systèmes hybrides linéaires. Cette application a connu trois générations ; la première était réalisée par une programmation en Mathematica, elle bénéficie ainsi de la puissance de manipulation symbolique de cette dernière. Par contre la manipulation des prédicats et en particulier le calcul des successeurs continus était difficilement réalisée par cette version. La deuxième version utilisait un programme principal écrit en langage Mathematica et qui appelait des subroutines, écrites en C++, pour le calcul des successeurs continus des régions. La troisième version est entièrement écrite en C++, pour éviter les conversions entre les expressions en Mathematica et les structures de données en C++.

L'application HyTech traite des fichiers textuels, comportant deux parties séparées, La première partie est destinée à la description du système et la seconde à son analyse, c'est une séquence de commande d'analyse itérative.

4.3.1 Description de l'automate

La description du système est une représentation textuelle d'automates hybrides linéaires. L'utilisateur décrit le système comme une composition de plusieurs composants, chacun est décrit par un automate hybride linéaire. HyTech effectue ensuite la composition synchrone de tous les automates fournis.

Un nom est donné à l'automate grâce à l'instruction **automaton**. L'instruction **synclabs** est utilisée pour la déclaration des labels de synchronisation. L'état initial de l'automate est spécifié grâce à l'instruction **initialy** et est caractérisé par le sommet initial et les valeurs initiales des variables d'état continues.

```
automaton Reservoir          -- nom du réservoir.
synclabs : fermer, ouvrir ;  -- labels de synchronisation
initially S1 & x = 0 & y = 10; -- région initiale
```

Un automate est décrit par ses composants, comme suit.

Les variables : Les variables sont données en haut de la description par l'instruction **var**, elles peuvent être de cinq types :

- `discrete` : Une variable discrète à une dérivée qui vaut toujours 0 ;
- `clock` : Une horloge à une dérivée qui vaut toujours 1 ;
- `Stopwatch` : Une variable dont la dérivée vaut soit 1 soit 0.
- `Parameter` : Un paramètre à une dérivée nulle dans tous les sommets, et ne doit jamais être assignée à une valeur numérique.
- `Analog` : Une variable analogique n'a pas de restriction syntactique.

```
var
        x, y, z : clock ;          -- des horloges
        m1, m2, m3 : analog; -- des variables continues
```

Les termes, les expressions et les contraintes linéaires : Un terme linéaire est soit un nombre rationnel ou une variable multiplié par un nombre rationnel. Une expression linéaire est une somme de termes linéaires. Une contrainte linéaire est une égalité (=) ou une inégalité (<, ≤, >, ≥) entre des expressions linéaires.

Les sommets : Chaque composant de l'automate hybride comprend une liste de sommets, dont chacun est caractérisé par un nom, son invariant et les dérivées de variables continues.

La déclaration des sommets s'effectue par l'instruction **loc**. Le terme **dx** est utilisé pour designer $\dot{x}$. L'invariant du sommet exprimant la durée de séjour maximale dans ce sommet est introduit en utilisant l'instruction **while**. L'instruction **wait** est utilisée pour décrire la fonction d'évolution des variables dans le sommet. Par exemple l'instruction

```
loc S1            -- nom du sommet
while x <= 20  -- l'invariant du sommet
wait {dx=2, dm2 = 4} --  dynamique des variable x et m2
```

signifie qu'on peut rester dans le sommet S1 tant que la variable continue x dont la dérivé est 2, est inférieur à 20. L'invariant associé à un sommet peut être une conjonction d'expressions linéaires, comme par exemple ($x \leq 3$ et $10 \leq m1 \leq 20$). mais ne doit pas être une disjonction d'expressions linéaires.

A chaque sommet est affecté un ensemble de transitions sortantes de ce sommet.

Les transitions : Une transition est caractérisée par sa garde, son affectation et son sommet destination. Dans HyTech la garde est exprimée par l'instruction **when**, l'affectation par l'instruction **do**, elle permet la mise à jour des valeurs de certaines variables continues après l'exécution de cette transition. L'instruction **goto** permet d'identifier le sommet but de la transition. Ainsi l'instruction

```
when true goto S2;
when x ≤ 5 & m3 = 10 do {} goto S3;
when asap do {x' = 1} goto S4;
```

Les gardes des transitions sont des conjonctions d'expressions linéaires mais non des disjonctions. Une transition spécifiée par le mot clé **asap** signifie que cette transition est urgente est qu'elle doit être franchise des que possible.

La déclaration d'un automate se termine par un **end**. Dans HyTech un commentaire est précédé par "—".

Exemple 4.1 : Considérons l'automate hybride en figure 4.1.b. La description de cet automate en langage HyTech est représentée comme suit :

```
-- Automate du réservoir
var
  x, m3 : analog;
automaton Reservoir
initially S11 & x=0 & m3=10;
 --Initialement le sommet S11 est actif et les variables
continues valent respectivement 0 et 10.
synclabs:;
-- sommet S11
loc S11: while x <= 4 wait {dx = 1,dm3 = 2}
  when x = 4 do {x'= 0} goto S21;
-- sommet S21
loc S21: while x<=2 wait {dx=1, dm3=-4}
  when m3=0 goto S22;
  when x = 2 do {x'=0} goto S11;
-- sommet S22
loc S22: while x = 2 wait {dx=1, dm3=0}
  when x = 2 do {x'=0} goto S11;
end
```

4.3.2 Analyse de l'automate

La section d'analyse comporte deux parties, une partie pour la déclaration des variables de type régions, et une partie de commandes itératives pour l'analyse. Ces commandes manipulent deux types d'objets, les expressions sur les régions, et les expressions booléennes. Une région peut être sauvegardée dans une variable. A titre d'exemple,

```
Var     reg_init, reg_final : region;
```

Cette expression déclare deux variable, reg_init et reg_final de type région.

HyTech manipule des régions définies à partir des inégalités linéaires et des contraintes sur les sommets. il fournit un certain nombre d'opérations pour manipuler ces régions, incluant le calcul de l'ensemble des états accessibles depuis une région, le calcul des successeurs, la quantification existentielle, l'enveloppe convexe et les opérations élémentaires booléennes (comparaison, test du vide,. . .).

Les inégalités linéaires : Les expressions de région les plus simples sont des inégalités linéaires. Par exemple, x <= 100 est une expression de région qui désigne l'ensemble de tous les états où la variable x à une valeur inférieur ou égale à 100.

Les contraintes sur les sommets :

```
loc [auto] = som
```

Le sommet dont le nom est som doit être un des sommet de l'automate auto. Cette expression désigne l'ensemble des états dont le composant sommet est som.

Combinaisons booléennes :

```
~ exp_reg ;              -- négation d'expression.
exp_reg & exp_reg ;      -- conjonction d'expressions.
exp_reg | exp_reg ;      -- disjonction d'expression.
```

L'opérateur ~ désigne la négation d'une expression de région, il représente le complément de son opérande. Les opérateurs & et | désignent respectivement l'intersection et réunion de leurs opérandes.

La différence :

```
diff (exp_reg, exp_reg) ;      -- differnce des régions.
```

L'expression diff (R_1, R_2) est équivalente à l'expression R_1 & ~R_2, elle désigne l'ensemble des état satisfaisant le premier argument mais pas le second.

Post / Pré :

```
Post (exp_reg) ;      -- successeur discret et continu d'une région.
Pre (exp_reg) ;       -- prédécesseur discret et continu d'une
région.
```

L'expression Post (W) donne tout l'espace d'état atteignable depuis un état de W en franchissant une transition discrète ou continue (passage d'une durée de temps finie). Et de la même manière Pre (W) donne tout l'espace d'état à partir duquel on peut atteindre W en franchissant une transition discrète ou continue.

Quantification existentielle :

```
hide (liste_variables) in (exp_reg) endhide
```

L'expression **hide** évalue une région en quantifiant une liste de variables. à titre d'exemple l'expression

```
prints hide x in x <= 1 & y = x endhide
```

donne en sortie la région $y \leq 1$. Généralement, les variables à quantifier sont citées séparées par des virgules. elles peuvent être remplacées par le mot clé **all** (pour toutes les variables) ou non-parameters (pour toutes les variables qui ne sont pas déclarées comme étant des paramètres).

L'enveloppe convexe :

```
hull (exp_reg) ;
```

L'expression hull (exp_reg) donne une région qui est l'enveloppe convexe de toutes les valuation des variables définissant la région exp_reg. A titre d'exemple :

```
S1 := loc[auto_1]=S_a & loc[auto_2]=S_b1;
S2 := loc[auto_1]=S_a & loc[auto_2]=S_b2;
S3 := hull(S1 & x=0 | S1 & x=1 | S2 & x=1);
```

La région S3 est donnée par (S1 et $0 \leq x \leq 1$) ou (S2 et $x = 1$).

L'atteignablilité :

```
reach forward from (exp_reg) endreach ;
reach backward from (exp_reg) end reach ;
```

Ces deux expressions sont utilisées pour le calcul de l'espace atteignable. La première utilise la méthode d'analyse en avant pour le calcul de l'espace atteignable depuis une région initiale en réitérant la fonction Post jusqu'à la convergence. La seconde utilise la fonction d'analyse en arrière pour calculer l'espace d'état à partir duquel en peut atteindre une région donnée, et ceci en réitérant la fonction Pre jusqu'à la convergence.

La suite d'instruction suivante permet de déterminer si l'automate du réservoir (figure 4.1.b) visite les sommets S21 et S22.

```
var
   reg: region;
reg := reach forward from (loc[Reservoir]=S11 & x=0 & m3=10)
endreach;
if empty (reg & loc[Reservoir] = S21)
then prints "Le sommet S21 n'est pas atteignable depuis la région
initiale";
else prints "Le sommet S21 est atteignable depuis la région initiale ";
endif;
if empty (reg & loc[Reservoir] = S22)
then prints "Le sommet S22 n'est pas atteignable depuis la région
initiale ";
else prints "Le sommet S22 est atteignable depuis la région initiale ";
endif;
```

Comme prévu le résultat donné par HyTech confirme que l'automate de la figure 4.1.b ne visite jamais le sommet S22.

Si dans ce cas particulier HyTech a pu calculer l'espace atteignable en temps fini, ceci n'est pas toujours le cas. En effet, dans plusieurs le calcul de l'espace atteignable peu être très coûteux en temps ou tout simplement ne converge pas.

A titre d'exemple le calcul de l'espace atteignable par l'automate de la figure 4.2.b ne converge pas. Dans ces situations HyTech offre la possibilité d'itérativement raffiner le calcul jusqu'à l'obtention d'un résultat avec une précision suffisante.

La suite d'instructions suivante décrit l'automate de la figure 4.2.b, et vérifie si tous les sommets sont visités par l'automate.

```
-- Automate du réservoir
var
    x, m3 : analog;
automaton Reservoir
initially S11 & x=0 & m3=10;
 --Initialement le sommet S11 est actif et les variables continues
valent respectivement 0 et 10.
synclabs:;
--sommet S11
loc S11: while x <= 4 wait {dx = 1,dm3 = 2}
    when x>=2 & x <= 4 do {x'= 0} goto S21;
--Sommet S21
loc S21: while x<=6 wait {dx=1, dm3=-4}
    when m3=0 goto S22;
    when x>=2 & x<=6 do {x'=0} goto S11;
--Sommet S22
loc S22: while x<=6 wait {dx=1, dm3=0}
    when x>=2 & x<=6 do {x'=0} goto S11;
end

var reg_init, reached :region;
reg_init := (loc[Reservoir]=S11) & x=0 & m3=10;
reached := post(reg_init);
while empty(reached & loc[Reservoir]=S21)
do reached := post(reached);
endwhile;
prints "Le sommet S21 est atteignable depuis la région initiale";

reached := post (reg_init);
while empty (reached & loc[Reservoir]=S22)
do reached := post(reached);
endwhile;
prints "Le sommet S22 est atteignable depuis la region initiale";
```

Conclusion

Ce chapitre a pour but d'utiliser l'application HyTech pour éliminer les sommets non visités par l'automate hybride résultant de l'algorithme de traduction des RdPH D-élémentaires en automates hybrides présenté au chapitre précèdent. Pour cela nous vérifions pour chaque sommet q de l'automate hybride s'il existe au moins une valuation v des variables continues tel que l'état (q, v) est atteignable par l'automate hybride. L'application HyTech permet non seulement l'élimination des sommets non visités par l'automate hybrides, mais en plus elle permet le calcul de l'espace d'état atteignable ce qui est nécessaire pour la synthèse de contrôleur.

Le Problème Général de la Synthèse de Contrôleur

Ce chapitre a pour but de poser la problématique de la synthèse de contrôleur pour la classe des systèmes dynamiques hybrides considérés dans ce mémoire. La notion de synthèse de contrôleur utilisée ici dérive directement de la théorie présentée par Ramadge et Wonham pour les systèmes à événements discrets. Cette théorie a été étendue dans un premier temps aux systèmes à événement discrets temporisés. Actuellement plusieurs formulations du problème de la synthèse de contrôleur pour les systèmes dynamiques hybrides linéaires sont présentées.

5.1 Notion de synthèse de contrôleur pour les systèmes à évènements discrets

La théorie de synthèse de contrôleur a été initiée par les travaux de Ramadge et Wonham [RW87] [RW89] pour la commande des systèmes à événements discrets, elle sépare les concepts du fonctionnement du procédé en boucle ouverte et celui du système commandé en boucle fermée. Cette théorie a permis de définir pour les systèmes à évènements discrets les notions de commandabilité d'observabilité et de commande optimale, en employant les automates à états finis et les langages formels. L'objectif principal de cette théorie est la synthèse de la stratégie de commande la plus permissive possible et qui respecte les spécifications de fonctionnement désirées. Ces spécifications permettent de garantir

- la sûreté des trajectoires c'est à dire éviter au système d'entrer dans des états indésirables.
- l'atteignabilité d'un ensemble d'états but, à partir d'un sous ensemble d'états initiaux.

Pour synthétiser cette stratégie de commande, le procédé à commandé ou le système en boucle ouverte est modélisé par un automate $A = (Q, \Sigma, T, q_0, F)$ qui est vu comme un générateur spontané d'évènements. Dans cet automate l'ensemble des évènements Σ est partitionné en deux sous-ensembles, $\Sigma = \Sigma_C \cup \Sigma_U$. Σ_C est l'ensemble des évènements contrôlables, ce sont les évènements qui peuvent être autorisés ou interdits par la loi de commande. Tandis que Σ_U est l'ensemble des évènements incontrôlables, sur lesquels la loi de commande n'a aucune influence. F représente

l'ensemble des états finaux encore dits états marqués, ces états représentent l'accomplissement d'une tâche.

L'automate A modèle du procédé à contrôler est caractérisé par deux langages (un langage est un ensemble de séquences d'événements pouvant être générées par le système).

- L(A) est le langage accepté par A, c'est l'ensemble de toutes les séquences d'évènements physiquement possibles pouvant être générées par celui-ci ;

- $L_m(A) \subset L(A)$ est le langage marqué accepté par A, cet ensemble comporte les séquences d'événements qui mènent le système vers un état final (un état de F). Ce sont les séquences d'événements qui terminent une tâche avec succès.

La problématique du contrôle est la suivante :

Etant donné un procédé à contrôler et un ensemble de contraintes imposées sur le fonctionnement de ce procédé. Cet ensemble de contraintes est dit objectif du contrôle ou encore spécification. On veut réaliser un superviseur qui restreint le fonctionnement du système à un fonctionnement qui ne viole pas les contraintes imposées par la spécification, tout en restant le plus permissif possible.

Le superviseur (Figure 5.1) a pour rôle de d'observer les évènements générés par le procédé afin d'agir sur le comportement de celui-ci par interdiction ou autorisation des évènements contrôlables (les évènements incontrôlables sont toujours autorisés). Ainsi dans un état du système, un événement peut être généré par le procédé supervisé s'il est physiquement possible et s'il est autorisé par le superviseur.

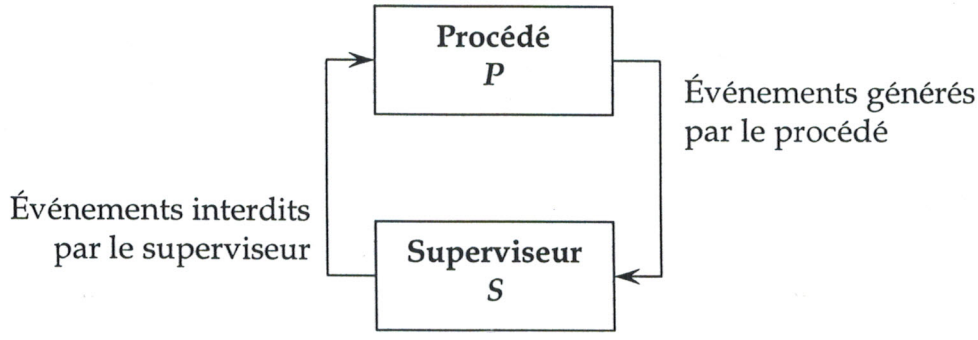


Figure 5.1. Schéma de principe de la synthèse de contrôleur.

L'objectif de la théorie de Ramadge et Wonham est de synthétiser le superviseur qui permet de restreindre le comportement du procédé modélisé par le langage L(A) à un comportement modélisé par un langage K tel que $K \subset L(A)$. Ce comportement K est généralement modélisé par un automate décrivant les contraintes à imposer en terme de séquences d'événements possibles et des états à interdire. Cependant l'existence d'événements incontrôlables implique que l'on ne

peut pas restreindre le comportement d'un système à événements discrets à n'importe quel sous-langage K de L(A). Le problème est donc de déterminer quel est le plus grand sous-langage de L(A) qui respecte la spécification. La solution de ce problème est liée à la notion de commandabilité.

Un langage $K \subseteq L(A)$ est contrôlable si toute occurrence d'un événement incontrôlable physiquement possible entraîne une évolution vers un état qui est également dans K. La commandabilité du langage K est une condition nécessaire et suffisante pour l'existence d'un superviseur tel que couplé au procédé, le fonctionnement en boucle fermée soit K. En effet, les évènements contrôlables physiquement possibles peuvent toujours être interdits par le superviseur lorsqu'ils entraînent une évolution vers un état n'appartenant pas à K.

S'il s'avère que K n'est pas contrôlable, le problème sera de chercher le sous-langage de K qui est contrôlable et qui est le plus permissif possible, ce langage, dit langage suprême contrôlable sup(K), représente la restriction minimale de K.

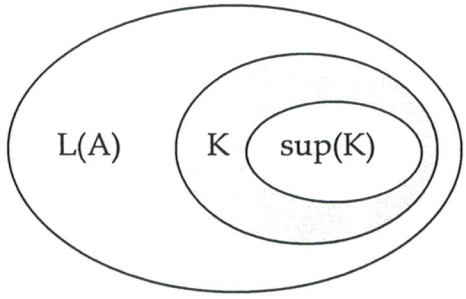


Figure 5.2. Langage suprême contrôlable.

Exemple 5.1 : Pour illustrer les concepts de la commandabilité et de supervision, prenant l'exemple d'un système manufacturier composé de deux machines identiques (Figure 5.3). Chacune des deux machines peut prendre trois états : A (arrêt), M (marche) et P (panne). La transition d'un état à un autre se fait par occurrence de l'un des quatre événements début du travail (α), fin du travail (β), panne de la machine (γ) et réparation de la machine (λ).Les événements α, α', λ et λ' sont considérés comme contrôlables ; les autres événements (β, β', γ et γ') sont donc incontrôlables. L'état initial (A, A') est le seul état marqué.

Le comportement désiré K du procédé est décrit par l'automate de la figure 5.4 qui impose d'usiner les pièces d'abord sur la machine 1 et ensuite sur la machine 2.

La machine 1 doit avoir accompli sa tâche (occurrence de l'événement β') avant que la machine 2 ne puisse commencer la sienne α'). L'absence de l'événement β sur la transition rebouclant de l'état 1 indique que le stock entre les deux machines est de capacité 1. Les transitions qui existent à partir d'un état de l'automate de contraintes (comportement désiré) correspondant aux événements autorisés lors du fonctionnement en boucle fermée.

Le superviseur correspondant au comportement contrôlable suprême de K est présenté en figure 5.5 à partir de l'état initial (A, A'), seul l'événement α, lié à la mise en service de la machine 1, est autorisé et permet l'évolution de l'automate vers l'état (M,A'). Dans cet état, si l'opération sur la machine 1 se termine normalement, l'événement β fait évoluer l'automate vers l'état (A, A', s) où le superviseur autorise la mise en marche de la machine 2 par l'intermédiaire de l'événement α'. Par contre si la machine 1 tombe en panne, l'automate passe à l'état (P, A') et revient à l'état (A, A') après la réparation. Dés l'instant où la machine 2 est en marche (état (A, M')) le superviseur autorise la mise en service de la machine 1 pour l'usinage d'une nouvelle pièce, et ainsi de suite. Les états marqués par s correspondent à l'existence d'une pièce dans de stock intermédiaire entre les deux machines.

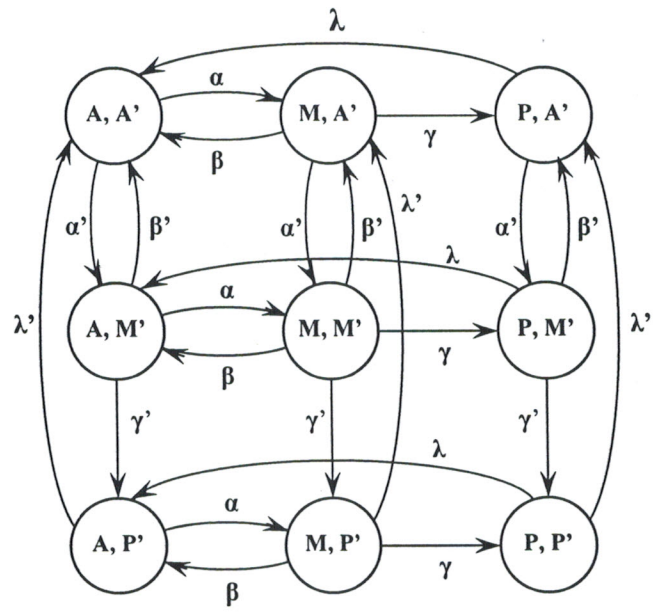


Figure 5.3. Système manufacturier comportant deux machines identiques.

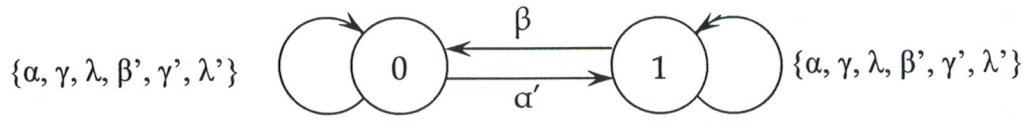


Figure 5.4. Comportement désiré.

Par rapport à l'évolution libre sans contraintes des deux machines (Figure 5.3), on remarque que le superviseur agit principalement sur l'ordonnancement des événements. α' et β. Ainsi, par exemple, l'état (A, A', s) est rajouté pour s'assurer que l'événement α' doit s'intercaler entre β et α.

5.2 La synthèse de contrôleur pour les systèmes à évènements discrets temporisés

L'approche classique de supervision utilise l'énumération des évènements et leurs ordre d'occurrence. Les contraintes temporelles quantitatives ne peuvent pas

être prises en compte directement dans cette approche. Parmi les nombreuses extensions de la théorie de base initiée par Ramadge et Wonham, certaines concernent le temps. L'objectif est d'intégrer le temps dans cette théorie. Cet objectif est justifié par l'intérêt croissant porté aux systèmes temps réel. Cela donne la plupart du temps un contrôleur moins contraignant.

La théorie concernant la synthèse de contrôleur pour les automates hybrides est bien établie. Les extensions de la théorie de ramadge et Wonham pour les systèmes à évènements discrets temporisés peuvent être classées en deux catégories :

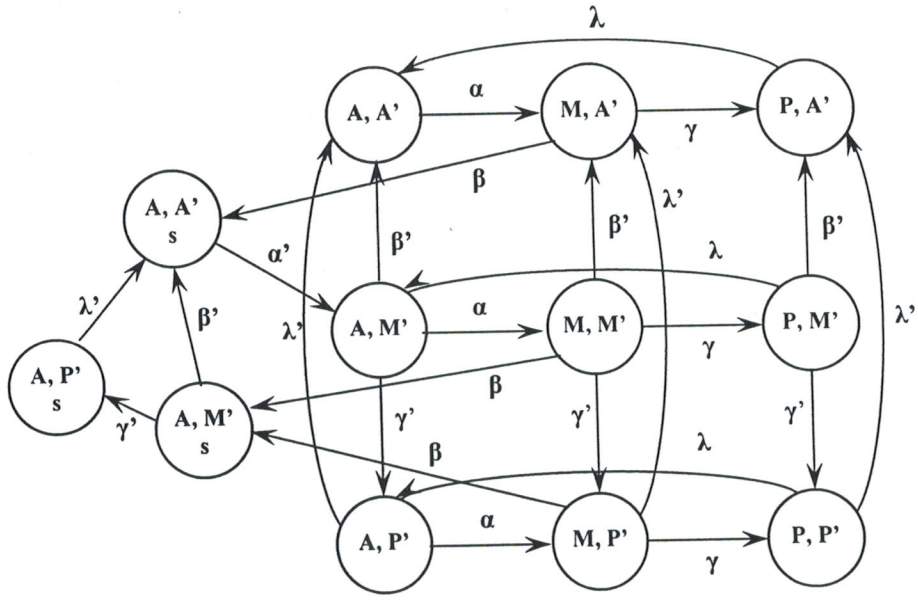


Figure 5.5. Comportement en boucle fermée.

1. La commande par supervision en temps discret

Dans ces approche on suppose que le temps est mesuré par une horloge digitale qui incrémente un compteur de top d'horloge, chaque top d'horloge, dit évènement Tick est traité d'une manière quasi identique à celle des autres évènements. Ce qui permet de définir d'une manière simple un langage temporisé (un langage comportant les séquences d'évènements ainsi que leurs dates d'occurrence).

Le comportement en boucle fermée, le concept de contrôlabilité d'un langage ainsi que le calcul du langage suprême contrôlable sont traités de la même façon que dans la théorie de Ramagde et Wonham. Cette approche donne une bonne solution au problème de commande par supervision des systèmes à événements discrets temporisés. La matérialisation de l'écoulement du temps à travers un événement spécifique inclut la dimension temporelle au sein même d'un langage comparable au

langage d'un automate non temporisé. Néanmoins, elle a un inconvénient majeur au niveau de la modélisation. Dans la plupart des cas, la nature discrète de l'espace et du temps engendre l'explosion combinatoire du nombre d'états du modèle.

2. La commande par supervision en temps continu

Plusieurs approches de commande par supervision ont été proposées pour pallier au problème de l'explosion combinatoire du nombre d'états engendrée par une nature discrète du temps. Ces approches, basées principalement sur l'outil automate temporisé, considèrent que le temps évolue d'une manière continue. Parmi ces approches nous citons : La commande supervisée par détemporisation, qui propose de réaliser le superviseur en détemporisant le modèle. Elle utilise les automates de régions et les automates de τ-régions.

5.3 La synthèse de contrôleur pour les systèmes dynamiques hybrides

Dans un système hybride le procédé à commander contient une partie événementielle et une continue. On peut donc avoir deux types de spécifications sur le comportement d'un tel système, à savoir :

1. Des spécifications sur la partie continue :

On peut exiger de l'espace d'état continu de rester dans une région particulière. Ce type de contraintes peut être introduit comme des invariant de sommet q. Ce qui implique que l'état continu ne doit pas quitter la région défini par *Inv* (q) quand le système séjourne dans q.

Exemple 5.2. Considérons le système manufacturier présenté en exemple 1.3 est dont le schéma est repris en Figure 5.6. Les machines 1, 2 et 3 ont des dureés de fabrication de 10, 7 et 22 respectivement.

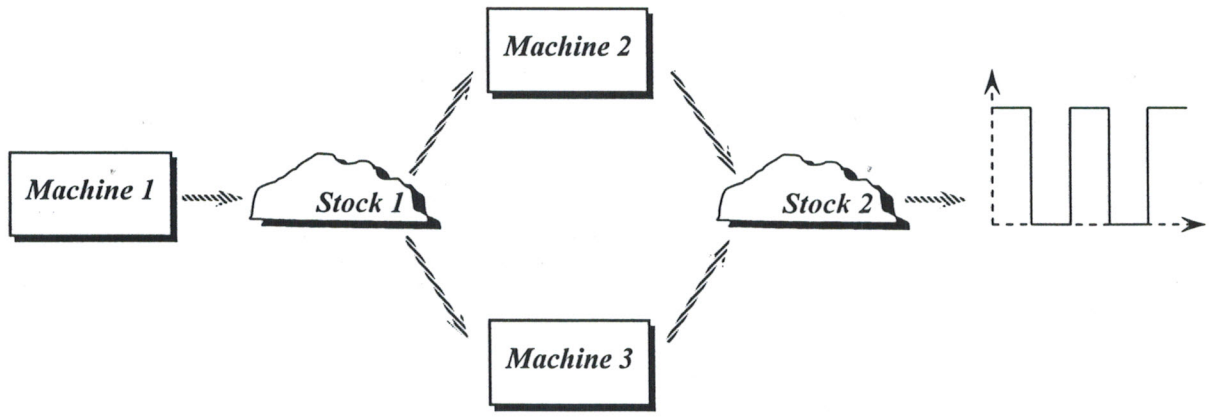


Figure 5.6 Système manufacturier

Exiger du stock 2 d'avoir un niveau comprit entre deux seuils S_{min} et S_{max} ; est un exemple de contrainte sur la partie continue qui exige la modification d'invariant de sommets.

2. Des spécifications sur la partie événementielle :

Les spécifications sur le partie événementielle sont ceux traités par la commande supervisés des systèmes à évènements discrets temporisés. Il ont comme objectif soit d'assurer que le système n'atteint jamais un sommet indésirable, soit d'assurer l'atteignablilité d'un ensemble de sommet à partie d'un espace d'état initial.

Conclusion

L'algorithme de traduction des RdPH D-élémentaires en automates hybrides s'effectue d'une manière hiérarchique, dans une première étape, la partie discrète est traduite en automate temporisé, ensuite la prise en considération de la partie continue s'effectue pour chaque régime discret. Cette forme hiérarchique peut être exploiter pour une synthèse de contrôleur hiérarchique, d'abord la partie discrète est utilisé pour satisfaire les contraintes discrètes, les contraintes sur la partie continue seront ensuite considérées et vont à nouveau modifier les dates de commutation entre les sommets.

Conclusion et perspectives

Ce travail s'articule autour de la modélisation en vue du contrôle des systèmes dynamique hybrides. Une classe particulière des systèmes hybrides a été considérée, à savoir la classe des systèmes hybrides positifs et linéaires par morceaux. Cette classe englobe plusieurs problèmes réalistes, comme les procédés batch et les systèmes manufacturiers traitant une quantité importante de produits.

Pour cela nous avons adopté la démarche suivante :

- Tout d'abord nous avons introduit un nouveau modèle pour la représentation de cette classe de systèmes hybrides. Ce modèle dit RdPH D-élémentaire, se distingue du RdPH classique [DA01] dans deux points :

1) dans un RdPH D-élémentaire, la partie discrète est représentée par un RdP T-temporel et non par un RdP T-temporisé comme dans le cas des RdPH. Les RdP T-temporels introduisent un indéterminisme quand aux dates d'occurrences des évènements discrets.

2) dans un RdPH D-élémentaires la partie événementielle commande le comportement de la partie continue, tandis que cette dernière n'a aucune influence sur la partie continue. Ce qui correspond, dans plusieurs situations, à un comportement réaliste.

- Nous avons ensuite proposé un algorithme pour la traduction des RdPH D-élémentaires en automates hybrides. La traduction des RdPH D-élémentaires en automates vise à associer la puissance de modélisation des réseaux de Petri à la puissance d'analyse et de manipulation formelle des automates. L'algorithme de traduction proposé s'effectue d'une manière hiérarchique et structurelle. Pour traduire un RdPH D-élémentaire en automate temporisé, nous avons séparé les parties discrète et continue. La partie discrète traduite en automate temporisé a son comportement qui est indépendant, elle peut être étudiée sans considération de la partie continue. A chaque sommet de cet automate temporisé correspond un comportement de la partie continue, qui peut être modélisé par un automate hybride, ce qui donne un modèle automate hybride sous une forme hiérarchique.

- L'automate hybride modélisant le comportement d'un RdPH D-élémentaire peut être utilisé pour l'analyse de manière générale de ce dernier, et plus précisément, pour la synthèse de contrôleur. En effet, la nature hiérarchique de ce

modèle peut être exploitée dans une synthèse hiérarchique de contrôleur, seul le modèle discret (automate temporisé) est utilisé pour répondre aux spécifications discrètes. Le modèle global sera utilisé satisfaire les contraintes sur la partie continue.

La traduction de la partie continue (RdPCC) en automate hybride s'effectue aussi d'une manière structurelle, sans considération pour le marquage initial, ce qui peut résulter en des sommets de l'automate hybride qui ne seront jamais visités. L'outil HyTech a été utilisé pour éliminer ces sommets ainsi que toutes les transitions qui ont ces sommets comme sommets sources ou sommets buts.

La continuité immédiate de ce travail consiste en la conception d'une approche et d'un algorithme qui permettent la synthèse de contrôleur pour la classe des systèmes dynamiques hybrides considérée. Le modèle RdPH D-élémentaire modélisera le procédé à commander (procédé en boucle ouverte). Un des avantages de ce modèle est qu'il permet de découpler les dynamiques discrètes et continues. Cet avantage peut être étendu à la synthèse de contrôleur pour les système à flux continu, en effet, la théorie de Ramadge et Wonham, ainsi que toutes ses extensions présentent l'inconvénient de l'explosion de la taille du modèle. Cet inconvénient disparaît dans un modèle qui approche le flot discret de produits par un flux continu. A titre d'illustration si on impose a un niveau d'un stock d'être compris entre deux seuils S_1 et S_2, la taille du contrôleur ne dépend pas des valeurs numériques des seuils, si le niveau du stock est considéré comme une variable continue, alors qu'elle est fortement dépendante dans la théorie de Ramadge et Wonham et ses extensions où le niveau du stock est considéré comme étant une variable discrète. Dans la détermination du contrôleur, on s'inspirera fortement des résultats établis pour la vérification dans la communauté informatique.

Références bibliographiques

[1] Bibliography on Hybrid Petri Nets. http://www.diee.unica.it/_hpn.

[AA98] M. Allam et H. Alla, "*From hybrid Petri nets to hybrid automata*", Journal Européen des Systèmes Automatisés (APII-JESA), 32(9-10) : pp. 1165-185, 1998.

[ACH+95] R. Alur, C. Courkoubetis, N. Halbwachs, T.A. Henzinger, P.H. Ho, A. Olivero, J. Sifakis et S. Yovine, "*The algorithmic Analysis of Hybrid Systems*" Theoretical computer science, (138): 3-34, 1995.

[AD94] R. Alur et D. Dill, "*The theory of timed automata*", Theoretical Computer Science, 126: pp. 183-235, 1994.

[AD98] H. Alla et R. David, "*Continuous and Hybrid Petri Nets*", Journal of Circuits, Systems and Computer, 8(1) : pp. 159-188, 1998.

[AKZ98] P. Antsaklis, X. Koutsoukos et J. Zaytoon, "*On hybrid control of complex systems: a survey*", Journal Européen des Systèmes Automatisés (APII-JESA), 32(9-10) : pp. 1023-1045, 1998.

[All98] Mohammed Allam, "*Sur l'analyse quantitative des RdP hybrides : une approche basée sur les automates hybrides*", PhD thesis, Laboratoire d'Automatique de Grenoble Institut National Polytechnique de Grenoble, 1998.

[And96] D. ANDREU, "*Commande et supervision des procédés discontinus : une approche hybride*", Thèse de doctorat, Université Paul Sabatier, Toulouse (France), 1996.

[Ant00] P. Antsaklis, "*A brief introduction to the theory and applications of hybrid systems*", proceedings of IEEE, Special Issue on Hybrid Systems: Theory and Applications, 88(7) : pp. 879-887, 2000.

[ASL93] P. Antsaklis, J. Stiver et M. Lemmon, "*Hybrid system modeling and autonomous control systems*", in Hybrid Systems, Grossman R.L., et al. (Edit), Lecture Notes in Computer Science, Springer-Verlag, Berlin (Allemagne), 736 : pp. 366-392, 1993.

[BBM94] M. Branicky, V. Borkar et S. Mitter, "*A unified framework for hybrid control*", proceedings of IEEE Conference and Decision and Control (CDC), Lake Buena Vista (USA) : pp. 4228-4234, 1994.

[BC98] J. Buisson et H. Comerais, "*Descriptor systems for knowledge modelling and simulation of hybrid physical systems*", Journal Européen des Systèmes Automatisés (APII-JESA), 32(9-10) : pp. 1047-1072, 1998.

[BCG87] G. Berry, P. Couronne et G. Gonthier, "*Programmation synchrone des systèmes réactifs : le langage Esterel*", Technique et Science Informatique, 4 : pp. 305-316, 1987.

[BD91] Bernard Berthomieu et Michel Diaz, "*Modeling and verification of time dependant systems using Time Petri Nets*", IEEE Transactions on Software Engineering, 17(3):259_273, Mars, 1991.

[BGG90] A. Benveniste, P. Guernic et G. Gonthier, "*Hybrid dynamical systems theory and the signal language*", IEEE Transaction on Automatic and Control, 35 : pp. 535-546, 1990.

[BGK+02] J. Bengtsson, W. Griffioen, K. J. Kristoffersen, K. G. Larsen, F. Larsson, P. Pettersson et W. Yi, "*Automated analysis of an audio control protocol using UPPAAL*", Journal of Logic and Algebraic Programming, 52{53 :163{181, Juillet-Août 2002.

[BGM93] A. Back, J. Guckenheimer et M. Myers, "*A dynamical simulation facility for hybrid systems*", in Hybrid Systems, Grossman R.L., et al. (Edit), Lecture Notes in Computer Science, Springer-Verlag, Berlin (Allemagne), 736 : pp. 255-267, 1993.

[BL99] P. Berlioux et M. Levy, "*Théorie des langages*", Note de cours, ENSIMAG, 1999.

[BLL+95] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi. "*Uppaal a Tool Suite for Automatic Verification of Real-Time Systems*". In Proc. of Workshop on Veri_cation and Control of Hybrid Systems III, volume 1066 of Lecture Notes in Computer Science, pages 232{243. Springer{Verlag, October 1995.

[Boy01] M. Boyer, "*Contribution à la modélisation des systèmes à temps contraint et application au multimédia*". Thèse de doctorat, Université Toulouse 3, 2001.

[Bra82] G.W. Brams, "*Réseaux de Petri : Théorie et Pratique*", Tome 1, Théorie et analyse, Masson, 1982.

[Bra83] G.W. Brams, "*Réseaux de Petri : Théorie et Pratique*", Tome 2, modélisation et applications, Masson, 1983.

[Bra95] M. Branicky, "*Studies in hybrid systems: modelling, analysis, and control*", Ph.D., Electrical Engineering and Computer Science, Massachusetts Institute of Technology (MIT), Boston (USA), 1995.

[Bro93] R. BROCKETT, "*Hybrid models for motion control systems*", in Essays in Control, Trentelman H. et Willems J. (Edit), Birkhauser, Boston (USA) : pp. 29-53, 1993.

[BST98] S. Bornot, J. Sifakis, et S. Tripakis, "*Modeling urgency in timed systems*". Lecture Notes in Computer Science, 1536 : 103-129, 1998.

[Cas96] Chritos. G. Cassandras et Stephane Lafortune, "*Introduction to disrete event systems*" Kluwer Academic Publishers, 1999. ISBN # 0-7923-8609-4

[Cer99] A. Cerone et A. Maggiolo-Schettini, *Time-based expressivity of time Petri nets for system specification*". Theoretical Computer Science, 216. Elsevier, 1999.

[Cha98] R. Champagnat, "*Supervision des systèmes discontinus : définition d'un modèle hybride et pilotage en temps réel*", Thèse de doctorat, Université Paul Sabatier, Toulouse (France), 1998.

[CMS] http ://www.lsv.ens-cachan.fr/_fl/cmcweb.html

[CPA+98] R. Champagnat, H. Pingaud, H. Alla, C. Valentin-Roubinet, J.M. Flaus et R. Valette, "*A gas storage example as a benchmark for hybrid modelling: a comparative study*", Journal Européen des Systèmes Automatisés (APII-JESA), 32(9-10) : pp. 1233- 1254, 1998.

[CPH+87] P. Caspi, D. Pilaud, N. Halbwatchs et J. Plaice, "Lustre a declarative language for programming synchronous systems", proceedings of 14th ACM Symposium on Principles of Programming Languages, Munich (Allemagne) : pp. 178-188, 1987.

[CR03] F. Cassez et O.H. Roux, "*Traduction structurelle des réseaux de Petri temporels vers les automates temporisés*". In 4ième Colloque Francophone sur la Modélisation des Systèmes Réactifs, (MSR'03), Metz, France, octobre 2003.

[CR04] F. Cassez et O.H. Roux. "*Structural translation from time Petri nets to timed automata*". In The 4th International Workshop on Automated Verification of Critical Systems (*AVoCS 2004*), London, United Kingdom, Septembre 2004.

[CR83] J.E. Coholahan et N. Roussopoulos, "*Timed requirements for timed driven systems using augmented Petri nets*", IEEE Transactions in Software Engineering, 9. IEEE Computer Society, 1983.

[DA01] R. David et H. Alla, "*On hybrid Petri Nets*", Discrete Event Dynamic Systems: Theory and Applications, Vol. 11, Numbers 1/2, pp. 9-40, 2001.

[DA05] R. David et H. Alla, " *Discrete, Continuous, and Hybrid Petri Nets*", Springer, 2005.

[DA87] R. David et H. Alla, "*Continuous Petri Nets*" Dans les proceedings of the eight European workshop on application and theory of Petri nets, Pages 275-294, Zaragoza (Espagne), Juin 1987.

[DA92] R. David et H. Alla, "*Du Graphcet aux Réseaux de Petri*", Hermès, 1962.

[DAD94] E. Dubois, H. Alla et R. David, "*Les réseaux de Petri Continus à Vitesse fonction du temps*", Journal Européen des systèmes automatisés (APII-JESA), 28(5) : pp 425-443, 1994.

[DAP93] I. Demongodin, N. Aubry et F. Prunet, "*Batch Petri Nets*", proceedings of IEEE International Conference on Systems, Man and Cybernetics, Le Touquet (France) : pp. 607-617, 1993.

[Dav91] R. David, "*Modeling of Dynamic Systems by Petri Nets*", Proceedings of 1st European control conference (ECC), Grenoble (France) : pp. 136-147,1991.

[DK98] I. Demongodin et N. Koussoulas, "*Differential Petri nets: representing continuous systems in a discrete-event world*", IEEE transactions on Automatic and Control, 43(4) : pp. 573-578, 1998.

[DR03] I. Demongodin et S. Rouibia, "*Modélisation par Réseaux de Petri Lots et Analyse de L'état Stable Par Automates Hybrides*", 4e conférence francophone de modélisation et simulation MOSIM'03, (Toulouse) France, 2003.

[DY96] C. Daws et S. Yovine, "*Reducing the number of clock variables of timed automata*". In 1996 IEEE Real-Time Systems Symposium (RTSS'96), pages 73{81, Washington, DC, USA, december 1996. IEEE Computer Society Press.

[ECO93] H. Elmqvist, F. Cellier et M. Otter, "*Object-oriented modeling of hybrid systems*", proceedings of European Simulation Symposium (ESS'93), Delft, (Pays-Bas) : pp. 31-41, 1993.

[Fla98] J.M. Flaus, "*Discrete supervisor synthesis for a class of continuous systems*", proceedings of IEEE Conference on Decision and Control, Tempa (USA) : pp. 31-37, 1998.

[GM99] Stéphane Gaubert et Jean Mairesse, "*Asymptotic analysis of heaps of pieces and application to timed Petri nets*", In Peter Bucholz et Manuel Silva, editors, Proceedings of the 8th international workshop on Petri Nets and Performance Models (PNPM'99), pages 158_169, Zaragoza, Spain, September 1999.

[GS98] H. Genrich et J. Schuart, "*Modelling and verification of hybrid systems using hierarchical coloured Petri Nets*", proceedings of 3rd International Conference on Automation of Mixed Processes (ADPM'98), Reims (France) : pp. 17-24, 1998.

[GZ01] H. Geguen et J. Zaytoon, "*Vérification des systèmes hybrides*", In J. Zaytoon, "*Systèmes dynamiques hybrides*", Hermes, Paris, 2001.

[Hal93] N. Halbwachs, "Delay analysis in synchronous programs", Proceedings of 5th Int. Conference on computer-aided Verification, Springer-Verlag, LNCS 697,pp. 333-346, 1993.

[HHW95a] T.A. Henzinger, P.H. Ho, et H. Wong-Toi, "*HYTECH : the Next Generation*", Dans les proceedings de 16th IEEE Real-Time Systems Symposium (RTSS'95), pp. 56–65. IEEE Computer Society press, 1995.

[HHW95b] T.A. Henzinger, P.H. Ho, et H. Wong-Toi, "*A User Guide to HYTECH*", Dans les proceedings du 1st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'95), vol. 1019 of Lecture Notes in Computer Science, pp. 41–71. Springer-Verlag, 1995.

[HHW97] T.A. Henzinger, P.H. Ho, et H. Wong-Toi, "HyTech : A Model-Checker for Hybrid Systems", Journal of Software Tools for Technology Transfer, vol. 1(1–2) :110–122, Octobre 1997.

[HKP+95] T.A. Henzinger, P.W. Kopke, A.Puri et P. Varaiya, "*What's decidable about hybrid automata? The algorithmic analysis of hybrid systems*", Proceedings of 27th annual ACM Symposium on theory of computing, pp. 373-382, 1995.

[HLS99] K. Havelund, K. G. Larsen, and A. Skou, "Formal verification of a power controller using the real-time model checker UPPAAL". In 5th International AMAST Workshop on Real-Time and Probabilistic Systems, 1999.

[HR98] T.A. Henzinger, V. Rusu, "*reachability verification for hybrid automata*", Proceedings of 1st Int. Workshop on hybrid systems: Computation and control, Springer-Verlag, LNCS1386, pp. 190-204, 1998.

[HV87] M.A. Holliday et M.K. Vernon, "A generalized timed Petri net model for performance analysis". IEEE Transactions in Software Engineering, 13. IEEE Computer Society, 1987.

[HyT] http ://www-cad.eecs.berkeley.edu/_tah/HyTech/

[Kha97] W. Khansa, "*Réseaux de Petri p-temporels : contribution à l'étude des systèmes à événements discrets*". Thèse de doctorat, Université de Savoie, 1997.

[KLL+97] K.J. Kristoffersen, F. Laroussinie, K.G. Larsen, P. Pettersson et W.A. Yi, "Compositional Proof of a Real-Time Mutual Exclusion Protocol", dans les proceedings du 7th International Joint Conference on Theory and Practice of Software Development (TAPSOFT'97), vol. 1214 of Lecture Notes in Computer Science, pp. 565–579. Springer-Verlag, 1997.

[KRO] http ://www-verimag.imag.fr/TEMPORISE/kronos/

[LAD91] J. Le Bail, H. Alla et R. David, "Hybrid Petri Nets", proceedings of European Control Conference (ECC), Grenoble (France), pp. 1472-1477, 1991.

[LBR96] G. Labinaz, M. M. Bayoumi et K. Rudie, "Modeling and control of hybrid systems: a survey", proceedings of 13th IFAC world congress, San Francisco (USA) : pp. 293-304, 1996.

[LeB92] J. Le Bail, "Sur les réseaux de Petri continus et hybrides" Thèse de Doctorat, Institut National Polytechnique de Grenoble (France), 1992.

[Lil99] J. Lilius, "*Eficient state space search for time Petri nets*", In MFCS Workshop on Concurrency '98, volume 18 of ENTCS. Elsevier, 1999.

[Lim04] Didier Lime, "*Vérification d'applications temps réel à l'aide de réseaux de Petri temporels étendus*", Thèse de PhD, Université de Nante, 2004.

[LL95] F. Laroussinie et K.G. Larsen, "Compositional Model-Checking of Real-Time Systems", Dans les proceedings du 6th International Conference on Concurrency Theory (CONCUR'95), vol. 962 of Lecture Notes in Computer Science, pp. 529–539. Springer-Verlag, 1995.

[LL98] F. Laroussinie et K.G. Larsen, "CMC : A Tool for Compositional Model-Checking of Real-Time Systems"Dans les proceedings IFIP Joint International Conference Formal Description Techniques & Protocol Specification, Testing, and Verification (FORTE-PSTV'98), pp. 439–456. Kluwer Academic, 1998.

[LLPY97] K.G. Larsen, F. Larsson, P. Pettersson et W. Yi,"Efficient Verification of Real-Time Systems : Compact Data Structure and State-Space Reduction" Dans les proceedings du 18th IEEE Real-Time Systems Symposium (RTSS'97), pp. 14–24. IEEE Computer Society Press, 1997.

[LPY01] M. Lindahl, P. Pettersson et W. Yi, "Formal Design and Analysis of a Gearbox Controller", Springer International Journal of Software Tools for Technology Transfer (STTT), 3(3) :353{368, 2001.

[LR03] Didier Lime et Olivier H. Roux, "*State class timed automaton of a time Petri net*", In 10th International Workshop on Petri Nets and Performance Models, (PNPM'03). IEEE Computer Society, September 2003.

[Lue79] David. G. Luenberger, "Introduction to dynamic systems", John Wiley, New York, 1979.

[Mer74] P.M. Merlin, "*A study of recoverability of communication protocols*". Ph.D. Thesis, Department of Computer Science, University of California, 1974.

[MF76] Merlin

[Mos97] P. Mosterman, "*Hybrid dynamic systems: a hybrid bond graph modeling paradigm and its application in diagnosis*", Ph.D., Vanderbilt University, Nashville (USA) : 233 p., 1997.

[Mur89] T. Murata, "*Petri-nets: Properties, Analysis and Applications*", Dans les proceedings de IEEE, 77(4) : 541-580, 1989.

[NK93] A. Nerode et W. Kohn, "*Models for hybrid systems: automata, topologies, controllability, observability*", in Hybrid Systems, Grossman R., et al. (Edit), Lecture Notes in Computer Science, Springer-Verlag, Berlin (Allemagne). 736 : pp. 317-356, 1993

[NY00] P. Niebert et S. Yovine, "Computing optimal operation schemes for chemical plants in multi-batch mode". In HSCC, pages 338-351, 2000.

[Pet62] C.A. Petri, "Kommunication mit Automaten", Thèse de PhD, Université de Bonn, Allemagne, 1962.

[Pet81] J.L. Peterson, "Petri nets theory and the modeling of systems", Prentice-Hall, 1981.

[QBC96] Y. Quenec'hdu, J. Buisson et H. Cormerais, "Les systèmes dynamiques hybrides: le point de vue continu. Une application à l'électronique de puissance" Journée d'Étude Analyse et Supervision des Systèmes Dynamiques Hybrides, Lyon, France. 1996.

[Ram74] C. Ramchandani, "*Analysis of asynchronous concurrent systems using Petri nets*", Ph.D. Thesis, Project MAC, MAC-TR 120, MIT, 1974.

[RH80] C.V. Ramamoorhy et G.S. Ho, "*Performance evaluation of asynchronous concurrent systems using Petri nets*". IEEE Transactions in Software Engineering, 6. IEEE Computer Society, 1980.

[RW87] J.G. Ramadge et W.M. Wonham, "*Supervisory control of a class of discrete event* processes", SIAM J., Control and Optimisation, 25 :206{230, 1987.

[RW89] J.G. Ramadge et W.M. Wonham, "*The control of discrete event systems*", Proceedings of the IEEE, 77(1) :81{97, 1989.

[SA01] Alexandru Tiberiu Sava et Hassane Alla, "*Commande par supervision des systèmes à évènements discrets temporisés*", Modélisation des systèmes réactifs (MSR 2001), p.p, 71–86, 2001.

[Sav01] Alexandru Tiberiu Sava, "*Sur la synthèse de la commande des systèmes à événements discrets temporisés*", PhD thesis, Institut National polytechnique de Grenoble, Grenoble, France, novembre 2001.

[Sif77] J. Sifakis, "*Use of Petri nets for performance evaluation*". Measuring modelling and evaluating computer systems. North-Holland, 1977.

[Sif79] J. Sifakis. "*Performance evaluation of systems using nets. Net theory and applications*", Advanced course on general net theory of processes and systems, LNCS 84. Springer, 1979.

[Sta78] P. H. Starke. Free Petri nets languages. Mathematical Foundations of Computer Science, LNCS 64. Springer, 1978.

[STS93] J.E. Stromberg, J. Top et U. Soderman, "Variable causality in bond graphs caused by discrete effects", Proc. Intl. Conf. on Bond Graph Modeling, pp . 115-119, San Diego, CA, 1993 .

[SY96] I. Sifakis et S. Yovine "Compositional specification of timed systems (extended abstract)". In 13th Syposium on Theoretical Aspects of Computer Science, pages 347-359, Grenoble, France, february 1996. Springer-Verlag.

[Tav87] L. Tavernini, "Differential automata and their discrete simulators", Nonlinear Analysis, Theory, Methods, and Applications, 11(6) : pp. 665-683, 1987.

[Thé00] L. Thévenon, "Représentation des systèmes hybrides complexes par flux de données : développement d'un outil de modélisation et de simulation des procédées batch", Thèse de Doctorat, Institut National Polytechnique, Grenoble (France), 2000.

[TY98] S. Tripakis et S. Yovine, "Verification of the fast reservation protocol with delayed transmission using the tool Kronos", Dans les Proceedings of the 4th IEEE Real-Time Technology and Applications Symposium, RTAS'98, pages 165-, 1998.

[UPP] http ://www.uppaal.com/

[Val98] C. Valentin-Roubinet, "Modelling of hybrid systems: DAE supervised by Petri nets the example of a gas storage", proceedings of 3rd International Conference on Automation of Mixed Processes (ADPM'98), Reims (France) : pp.142-149, 1998.

[VS00] A. Van Der Schaft et H. Schumacher, "An Introduction to Hybrid Dynamical Systems", Lecture Notes in Control and Information Sciences, Springer-Verlag, Berlin (Allemagne), Londres (Angleterre), 251 : 175 p., 2000.

[Wal83] B. Walter, "*Timed Petri-nets for modelling and analyzing protocols with real-time characteristics*". Third IFIP workshop on protocols specification, testing and verification. North-Holland, 1983.

[Zay01] J. Zaytoon, "Systèmes dynamiques hybrides, Productique", Hermès, Paris (France), 2001.

[ZCN+97] J. Zaytoon, V. Carre-Menetrier, M. NicletM et P. De Loor, "*On the recent advances in grafcet*", proceedings of IFAC Workshop on Manufacturing Systems Modelling, Management and Control, Vienne (Autriche) : pp. 419-424, 1997.