IN310 - Modèles de systèmes embarqués

Charles Lesire-Cabaniols (ONERA / DCSD)
charles.lesire@onera.fr

3A-SEM - 2010-2011

IN310 - Modèles de systèmes embarqués

Charles Lesire-Cabaniols (ONERA / DCSD)
charles.lesire@onera.fr

3A-SEM - 2010-2011

Qu'est-ce qu'un système embarqué ?

- Un système électronique, comprenant capteurs, actionneurs, processeurs et moyens de communication, piloté par un logiciel, intégré au système qu'il contrôle et
  - soumis à diverses contraintes : d'espace, de consommation, de temps de réponse (système temps réel), de sécurité, de sûreté de fonctionnement ;
  - conception conjointe "embarqueur et embarqué", mêlant différentes spécialités : électronique, traitement du signal, informatique, réseaux, automatique - qui doivent se comprendre et coopérer.

Qu'est-ce qu'un système embarqué ?

- Domaines d'application divers :
  - SE orientés *commande* : transport (Aéronautique, Espace, Automobile, Ferroviaire, Maritime)
  - SE orientés *traitement du signal/calcul* : télécom, multimédia
- Conception $\rightsquigarrow$ modèle :
  - Comment représenter un système embarqué ?
  - Quel point de vue ?
    $\rightarrow$ commande : Système Hybride (variables continues et discrètes)

Types de variables

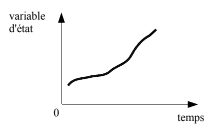
Le modèle mathématique d'un système est caractérisé par :

- la nature de la variable indépendante qui représente le temps
- la nature de ses variables d'état :
  - variables continues : prennent leurs valeurs sur le domaine des réels $\mathbb{R}$
  - variables discrètes : prennent leurs valeurs sur un domaine représenté par un ensemble dont le nombre d'éléments est dénombrable (ex : les entiers naturels $\mathbb{N}$, variables booléennes)

Types de systèmes

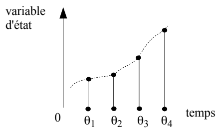
- Les systèmes continus :

- temps : variable continue (temps dense)
- variables d'état continues, évolution dictée par le temps
- équations algébro-différentielles $\dot{x}(t) = Ax(t) + Bu(t)$, transformée de Laplace
- Ex : vitesse de rotation d'un moteur



Types de systèmes

- Les systèmes échantillonnés :
- temps : variable discrète θ_0, $\theta_1 \ldots \theta_{n-1}$ θ_n, θ_{n+1} ...
- variables d'état continues (*observées* à θ_i)
- équations différence $X_{k+1} = A_k.X_k + B_k U_k$, transformée en Z
- Ex : vitesse moteur controlée par un microcontroleur.



Types de systèmes

- Les systèmes à événements discrets :
  - représentés par une suite d'*événements discrets* (ex : un plan)
  - temps : relation de précédence
  - variables d'état discrètes : valeur x(k+1) calculée directement à partir de x(k), sans considérer le temps (fonction des événements)
  - automates, réseaux de Petri
  - ex : nombre de pièces dans un système de manufacture

Types de systèmes

- Les systèmes hybrides :
  - évolution à la fois en fonction du temps continu et des événements discrets
  - variables d'état continues et variables d'état discrètes
  - automates hybrides, réseaux de Petri hybrides
  - ex : contrôle de température : événement (on/off), variable continue (temperature)

Exemple

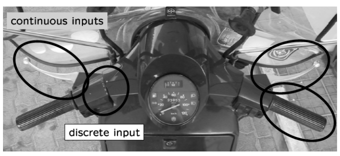
Un réservoir qui peut être rempli ou vidé. Un même système physique, mais deux points de vue :

- Modélisation du niveau de liquide : $S\dot{h}(t) = q_i(t) - u(t).\alpha h(t)$ avec :
  $h(t)$ la hauteur de liquide, $q_i(t)$ la vitesse de remplissage, $u(t)$ l'entrée ($u(t) = 0$: valve fermée, $u(t) = 1$: valve ouverte), S et α deux paramètres
- Modélisation de l'état du réservoir (vide ou plein) :
  Espace d'état X ={*vide, plein*}, contrôle U ={*ouvrir, fermer*}

Exemple

Un réservoir qui peut être rempli ou vidé. Un même système physique, mais deux points de vue :

- Système Hybride : considérer les deux points de vue



- Entrée discrète (rapport de vitesse)
- Entrées continues (frein, gaz)
- Etat dynamique continu (vitesse, vitesse du vent, carburant)

Plan du cours

- Modèles discrets
  - Réseaux de Petri (C. Lesire)
    4 C, 1 BE Tina, 2 BE Lego
  - Automates (J. Brunel)
    4 C, 2 BE Uppaal
- Modèles hybrides (C. Lesire)
  - 1 C (C. Lesire)
  - 2 BE StateFlow (F. Defaÿ)

Réseaux de Petri
IN310 - Modèles des SE

Charles Lesire-Cabaniols (ONERA / DCSD)
charles.lesire@onera.fr

3A-SEM - 2010-2011

Introduction

- 1962, Carl Adam Petri : Communication et composition entre automates
- Outil de modélisation de systèmes dynamiques : permet de raisonner sur les objets, les ressources et leur changement d'état
- Outil mathématique (formel) et outil graphique
  - permet de représenter le vrai parallélisme, la concurrence, contraintes de précédence,
  - analyse de bonnes propriétés (vivacité, borné, etc.) et propriétés structurelles : aide efficiente durant les phases de conception
  - peut être simulé et implémenté directement par un joueur de RdP

Introduction

- Applications :
  - évaluation de performances,
  - analyse et vérification formelles,
  - protocoles de communication,
  - contrôle de systèmes de production,
  - systèmes d'information (organisation d'entreprises),
  - gestion de bases de données,
  - IHM, etc.

Introduction

- Etat : les différentes *phases* par lesquelles passe le système ;
- Variables d'état : ensemble de variables qui permettent de connaître l'état du système.
  - Système continu : les variables d'état évoluent continuellement dans le temps ;
  - Système à événements discrets : les variables d'état changent *brusquement* à certains instants
- Evénement : son occurrence fait changer l'état du système
- Activité : *boîte noire* représente l'évolution du système entre 2 événements
- Processus : séquence d'événements et d'activités
  $\rightsquigarrow$ coopération, compétition, parallélisme

Présentation informelle

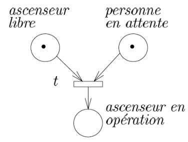
Éléments de base

- Place : interprétée comme condition, état partiel, ensemble de ressources
- Transition : associée à un événement qui a lieu dans le système
- Jeton : indique que la condition associée à la place est vérifiée (ou le nombre d'éléments qui la vérifient)

Présentation informelle

Comportement dynamique

- état : répartition des jetons dans les places,
- occurrence d'un événement : tir de la transition,
  - enlever les jetons des places d'entrée,
  - mettre les jetons dans les places de sortie.



Définitions

- Modèle formel, peut être caractérisé par :
  - graphe avec comportement dynamique ; représentation naturelle pour le concepteur,
  - ensemble de matrices d'entiers : comportement dynamique décrit par un système linéaire : représentation naturel pour l'ordinateur ;
  - système de règles : peut être utilisé avec les techniques d'I.A ;
- Validation par analyse et simulation ;
- Représente : parallélisme, synchronisme, séquence, conflit, concurrence.

Définitions

Réseaux de Petri $R =< P, T, Pre, Post >$

- P est un ensemble fini de places de dimension n ;
- T est un ensemble fini de transitions de dimension m ;
- $Pre : P \times T \rightarrow \mathbb{N}$ est l'application d'*entrée* (places précédentes),
- $Post : P \times T \rightarrow \mathbb{N}$ est l'application de *sortie* (places suivantes),

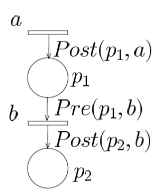
Réseau de Petri marqué $N =< R, M >$

- R est un réseau de Petri,
- $M : P \rightarrow \mathbb{N}$ est le marquage initial (distribution de jetons dans les places)

Définitions

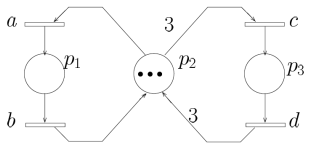
Exemple

- $R =< P, T, Pre, Post >$
- $P = \{p_1, p_2, p_3\}$
- $T = \{a, b, c, d\}$
- $Post(p_1, a) = 1,\ Pre(p_1, b) = 1,$ $Post(p_2, b) = 1$



Graphe et notation matricielle

Réseau de Petri marqué $N = <R, M>$



$$P = \{p_1, p_2, p_3\}, \qquad T = \{a, b, c, d\}$$

$$Pre = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Post = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$^tM = \begin{pmatrix} 0 & 3 & 0 \end{pmatrix}$$

Règle de fonctionnement

Transition sensibilisée à partir de M

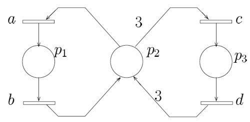
- il y a un nombre suffisant de jetons dans les places d'entrée,
- $\forall p \in P, \quad M(p) \geq Pre(p,t)$
- $M \geq Pre(\,.\,, t)$

Tir d'une transition à partir de M

- $\forall p \in P, \quad M'(p) = M(p) - Pre(p,t) + Post(p,t)$
- $M' = M - Pre(\,.\,, t) + Post(\,.\,, t) = M + C(\,.\,, t)$

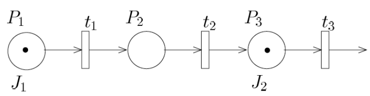
Règle de fonctionnement

- Enlève $Pre(p,t)$ jetons de chaque place précédente p (poids de l'arc d'entrée), et met $Post(p,t)$ jetons à chaque place de sortie p,
- Représente le changement d'état dû à l'ocurrence de l'événement associé à t.



Différentes interactions entre les processus

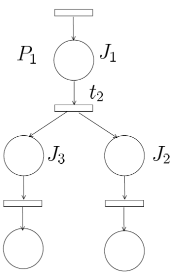
Séquence



- séquence d'un processus de fabrication :
  - P_i : phase i de l'opération sur la pièce,
  - t_i : passage d'une phase à une autre ;
- portion de l'itinéraire d'un système de transport :
  - P_i : chariot traverse la section i,
  - t_i : passage d'un chariot d'une section à une autre ;

Différentes interactions entre les processus

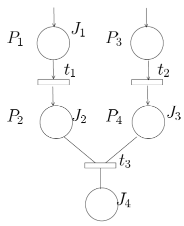
Branchement



- à partir de l'activité J_1, deux activités sont crées (J_2 et J_3),
- J_2 et J_3 évoluent de façon indépendante.

Différentes interactions entre les processus

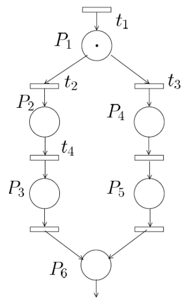
Jonction



- évolution indépendante de t_1 et t_2 (évolution asynchrone),
- synchronisme en t_3.

Différentes interactions entre les processus

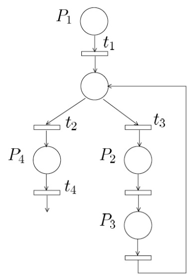
Choix



- choix entre t_2 (seq. P_2P_3) et t_3 (seq. P_4P_5) : seulement une peut être tirée ;
- les 2 séquences exécuteront P_6.

Différentes interactions entre les processus

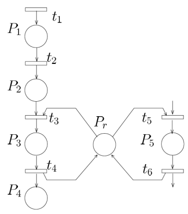
Répétition



- choix entre t_2 e t_3,
- répéter la séq. P_2P_3 un certain nombre de fois avant de exécuter P_4.

Différentes interactions entre les processus

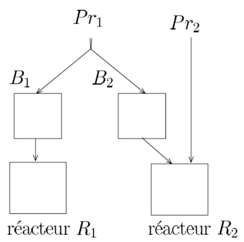
Allocation de ressources



- un même chariot doit servir différentes machines,
- un opérateur doit exécuter différentes activités (une à la fois).

Exemple : Système par lot

- peut produire deux produits (Pr_1 et Pr_2), utilisant 2 réacteurs (R_1 e R_2) de façon concurrente,
- produit Pr_1 : est produit par R_1 ou R_2 ; doit être, au préalable, stocké dans le *buffer* B_1 ou B_2 (respectivement).
- produit Pr_2 : est produit par le réacteur R_2.



Conflit et parallélisme

- Conflit structurel : ssi t_1 et t_2 ont au moins une place d'entrée en commun
$$\exists p \in P, \quad Pre(p, t_1)\, Pre(p, t_2) \neq 0$$
- Conflit effectif : ssi t_1 et t_2 sont en conflit structurel et sont sensibilisées par le marquage M
$$M \geq Pre(., t_1) \text{ et } M \geq Pre(., t_2)$$
- Parallélisme structurel : si t_1 et t_2 ne possèdent pas de place d'entrée en commun
$$\forall p \in P \quad Pre(p, t_1)\, Pre(p, t_2) = 0 \text{ ou } Pre(., t_1)^T \times Pre(., t_2) = 0$$
- Parallélisme effectif : t_1 et t_2 sont parallèles structurellement et
$$M \geq Pre(., t_1) \text{ e } M \geq Pre(., t_2)$$

Séquence de tir

$$\underset{M_0}{\begin{pmatrix}0\\3\\0\end{pmatrix}} \xrightarrow{a} \underset{M_1}{\begin{pmatrix}1\\2\\0\end{pmatrix}} \xrightarrow{a} \underset{M_2}{\begin{pmatrix}2\\1\\0\end{pmatrix}} \xrightarrow{b} \underset{M_1}{\begin{pmatrix}1\\2\\0\end{pmatrix}}$$

- M' accessible à partir de M : $M \xrightarrow{t} M'$
- $M_0 \xrightarrow{a} M_1$, $M_1 \xrightarrow{a} M_2$, $M_2 \xrightarrow{b} M_1$,
- si $M \xrightarrow{t_1} M'$, et $M' \xrightarrow{t_2} M''$, on a $s = t_1 t_2$ et $M \xrightarrow{t_1 t_2} M''$
- dans l'exemple, $M_0 \xrightarrow{s} M_1$, avec $s = aab$, s est dite séquence de tir

$$\begin{aligned} s : \quad & T && \rightarrow \mathbb{N} \\ & t && \mapsto \text{nombre d'occurrences de } t \text{ dans } s \end{aligned}$$

Séquence de tir

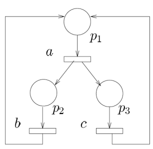
- Équation fondamentale : $\boxed{M' = M + Cs}$
- Etant donné M et une sequence s, existe-t-il M' t.q. $M \xrightarrow{s} M'$?
- Etant donné M et M', existe-t-il s t.q. $M \xrightarrow{s} M'$?

Réseau borné

- Place k-bornée : le nombre maximal de jetons de la place, pour tout marquage accessible, est plus petit que k

$$\forall M' \in \mathcal{A}(\mathcal{R}, M_0), \quad M'(p) \leq k$$

- Si $k = 1$, la place est dite binaire,
- Un réseau marqué est k-borné ssi toutes ses places le sont
- Un réseau marqué est binaire ssi toutes ses places le sont



Réseau vivant

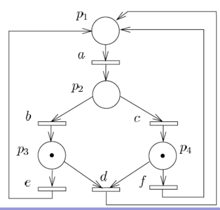
- Transition quasi-vivante :

$$\exists s \ / \ M_0 \xrightarrow{s} M \quad \text{et} \quad M \xrightarrow{t}$$

- Transition vivante :

$$\forall M \in \mathcal{A}(\mathcal{R}, M_0), \ \exists s \ / \ M \xrightarrow{st}$$

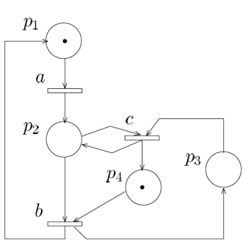
- Réseau vivant ssi toutes ses transitions sont vivantes

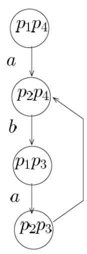


Réseau réinitialisable

- Réseau marqué réinitialisable s'il est possible de revenir au marquage initial à partir de n'importe quel marquage :

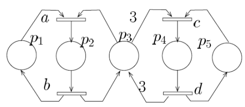
$$\forall M \in \mathcal{A}(\mathcal{R}, M_0), \quad \exists s \ / \ M \xrightarrow{s} M_0$$





Composantes conservatives

- circuit formé par p_1, p_2, a, b : $M(p_1) + M(p_2)$ est constant
  - $M_0 = {}^t\begin{pmatrix} 1 & 0 & 3 & 0 & 1 \end{pmatrix}$
  - $M_0 \xrightarrow{a} M' = {}^t\begin{pmatrix} 0 & 1 & 2 & 0 & 1 \end{pmatrix}$
  - $M' \xrightarrow{b} M'' = M_0$



- Marquage obtenu après une séquence de tir : $M' = M + Cs$
- Composante conservative : $f \ / \ {}^t f \ C = 0$
- dans l'exemple :

$${}^t f^1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \end{pmatrix}, {}^t f^2 = \begin{pmatrix} 0 & 1 & 1 & 3 & 0 \end{pmatrix}, {}^t f^3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Invariants de place

- Invariant de place = composante conservative + marquage
- ${}^tf\ C = 0 \Rightarrow {}^tf\ M = {}^tf\ M_0 \quad \forall M \in \mathcal{A}(\mathcal{R}, M_0)$
- $M(p_1) + M(p_2) = M_0(p_1) + M_0(p_2) = 1$
- $M(p_2) + M(p_3) + 3.M(p_4) = 3$
- $M(p_4) + M(p_5) = 1$

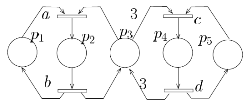
Remarque

- composante conservative : dépend seulement de la structure !
- invariant de place : dépend de la structure et du marquage

Composantes répétitives stationnaires

- Sous-réseau formé par c et d, et places p_3, p_4 et p_5 : le tir de $s = cd$ à partir de M_0 ramène au même marquage
- Transitions c et d forment une composante répétitive stationnaire
- $M' = M \Rightarrow C\, s = 0 \quad s$ composante répétitive
- Dans l'exemple :

$$s^1 = {}^t\begin{pmatrix}1 & 1 & 0 & 0\end{pmatrix},\ s^2 = {}^t\begin{pmatrix}0 & 0 & 1 & 1\end{pmatrix}$$



Invariants de transition

- Séquences s_i obtenues à partir du vecteur s
- Pour s_1 on peut avoir les invariants $s_{11} = ab$ et $s_{12} = ba$
- Il faut calculer $M \xrightarrow{ab}$ et $M \xrightarrow{ba}$ pour vérifier !

Remarque

- Composante répétitive : dépend seulement de la structure !
- Invariant de transition : dépend de la structure et du marquage

Analyse des propriétés

Analyse par énumération des marquages le graphe des marquages accessibles est calculé, vérifiant si le réseau est borné, vivant et réinitialisable.

Analyse structurelle calcul des composantes conservatives et répétitives stationnaires et des invariants correspondants ; ne permet pas toujours d'avoir une réponse, mais dans certains cas, permet d'obtenir une réponse simples et rapide des propriétés du réseau.

Analyse par réduction si le réseau est trop grand ou non borné, on peut réduire la taille du réseau, en utilisant certaines règles de réduction.

Analyse par énumération des marquages

Arbre de couverture

- On part du marquage initial M_0,
- On crée une branche pour chaque transition sensibilisée par M_0,
- La construction d'une branche est interrompue quand on rencontre un marquage
  - déjà calculé,
  - strictement supérieur à un marquage *de la branche qui est en train d'être explorée.*

Si le réseau est non borné, on introduit le symbole ω pour rendre l'arbre fini.

Analyse par énumération des marquages

Recherche des propriétés sur $\mathcal{A}(\mathcal{R}, M)$

- Réseau k-borné $\Leftrightarrow \mathcal{A}(\mathcal{R}, M)$ borné
- Réseau réinitialisable $\Leftrightarrow \mathcal{A}(\mathcal{R}, M)$ fortement connexe

$$\forall M_i, M_j \in \mathcal{A}(\mathcal{R}, M),\ \exists s \,/\, M_i \xrightarrow{s} M_j$$

- $\mathcal{A}(\mathcal{R}, M)$ fortement connexe $\Rightarrow$
  Réseau vivant $\Leftrightarrow$ Chaque transition étiquette au moins un arc

$$\forall t \in T, \exists M_i, M_j \in \mathcal{A}(\mathcal{R}, M),\ /\ M_i \xrightarrow{t} M_j$$

Analyse structurelle

Composantes conservatives

- Toute place qui appartient à une composante conservative est bornée
- Une place p qui n'appartient à aucune composante conservative ($f(p) = 0$) peut être bornée
- Une place non bornée n'appartient à aucune composante conservative

Un réseau de Petri pour lequel il existe une couverture de composantes conservatives ($f > 0$) est k-borné, *peu importe son marquage initial.*

Analyse structurelle

Invariants de place

${}^t f\, M = {}^t f\, M_0$ permet de calculer une limite pour chaque place p

$$f(p)M(p) \leq {}^t f\, M_0, \qquad M(p) \leq \frac{{}^t f\, M_0}{f(p)}$$

Analyse structurelle

Composantes répétitives

Réseau de Petri répétitif : il existe une couverture de composantes répétitives ($s > 0$)

- un réseau de Petri borné et vivant est répétitif
- un réseau non répétitif ($\exists t,\ s(t) = 0$) est non vivant ou non borné

Analyse par réduction

- On cherche à réduire la taille du réseau à analyser, en conservant ses propriétés ;
- Plusieurs types de réduction :
  - Place substituable
  - Place implicite
  - Transition neutre
  - Transitions identiques

Analyse par réduction

Place substituable

- Si p n'a qu'une transition en entrée t_e et qu'une en sortie t_s :

$$Post(p, t_e) = Pre(p, t_s)$$
$$\forall p' \in P,\ p' \neq p \Rightarrow Pre(p', t_s) = 0$$

- Sa réduction fournit un réseau sans p et où t_e et t_s sont remplacées par t_{es} :

$$\begin{aligned} \forall p' \in P,\ Pre(p', t_{es}) &= Pre(p', t_e) \\ Post(p', t_{es}) &= Post(p', t_e) + Post(p', t_s) \end{aligned}$$

Analyse par réduction

Place implicite

- Son marquage est une combinaison linéaire du marquage d'autres places : elle est donc inutile !
- Deux cas triviaux :
  - Deux places sont identiques (même transitions, Pre et Post) ;
  - Une place n'est connectée que par des boucles élémentaires (implicite / $\emptyset$).

Analyse par réduction

Transition neutre

- Une transition neutre est une transition connectée uniquement par des boucles élémentaires :

$$Pre(., t) = Post(., t)$$

- Pour conserver la propriété de vivacité, il faut qu'il existe une transition dans le réseau réduit permettant de vérifier cette vivacité :
  - Soit $\exists t_1,\ Post(., t_1) = Pre(., t)$
  - Soit $\exists t_2,\ Pre(., t_2) = Pre(., t)$

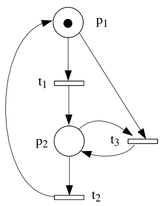
Analyse par réduction

Transitions identiques

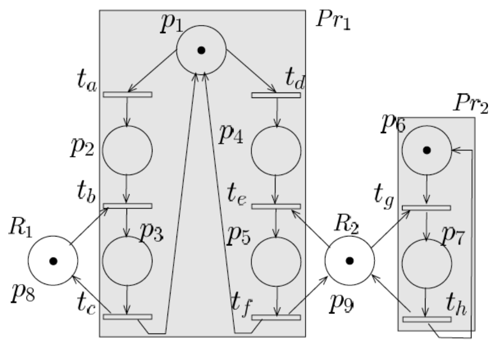
- Si deux transitions t_1 et t_2 sont identiques on peut supprimer l'une des deux

$$Pre(., t_1) = Pre(., t_1) \text{ et } Post(., t_1) = Post(., t_1)$$

Exemple



Exemple : Système par lot



- Ouvrir une fenêtre de commandes en ligne ; taper "nd" (NetDraw)
- Faire Help, Setup (5eme ascenceur) : indiquer le nombre de boutons de la souris
- Faire Help, Help : comment créer les places, les transitions, les arcs, comment changer les propriétés d'une place (marquage, label), d'un arc (poids), d'une transition (label). Comment déplacer, effacer les éléments...
- Editer le RdP des lecteurs écrivains
- Générer le graphe d'accessibilité (tools/reachability analysis/ cocher "marking graph", utiliser comme sortie "lts(.aut)".
- Dessiner ce graphe : click droit, "open file in nd" ; edit/draw ; déplacer les noeuds pour que le graphe soit lisible, et "séparer" les doubles flèches
- Générer le graphe d'accessibilité (tools/reachability analysis/ cocher "marking graph", utiliser comme sortie "verbose".

Caractéristiques de la représentation par RdP

- Modularité : est-il possible de décomposer un système complexe ?
- Composition : si les modules ont les bonnes propriétés, la composition de ces modules garde-t-elle les bonnes propriétés ? Ou est-il nécessaire d'analyser le système globale (composé) ?
- Calculabilité : existe-t-il des algorithmes pour l'analyse ?

Bloc bien-formé

un réseau de Petri avec :

- une transition d'entrée t_e et une transition de sortie t_s,
- réseau borné, vivant et réinitialisable si l'on rajoute une place p tel que $Pre(p, t_e) = Post(p, t_s) = 1$.
  Ex : séquence, if-then-else, do-while, fork-join.

Raffinement

Processus d'abstraction fait en deux temps :

- modélisation d'une première ébauche (*vision abstraite du système global*), avec des transitions *abrégées* (associées à des tâches complexes),
- à partir de ce RdP, remplacer les transitions *abrégées* par des blocs bien-formés représentant une vision détaillée de ces tâches complexes
- conception *top-down*
- analyse : si le réseau abstrait est un bloc bien formé, et les transitions sont représentées par des blocs bien formés, alors le réseau global est bien formé.

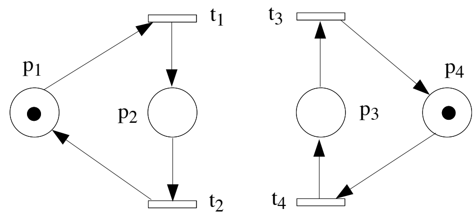
Composition

Processus "à objets" :

- modélisation détaillée de deux objets dès le départ
- construction du réseau global à partir de la composition de ces objets
  - Composition asynchrone (fusion des places)
  - Composition synchrone (fusion des transitions)
- conception *bottom-up*
- analyse : le réseau global composé ne conserve pas forcement les propriétés de chaque bloc

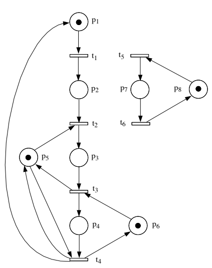
Composition synchrone

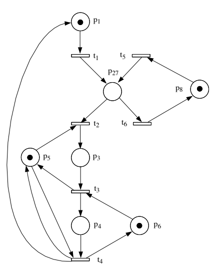
Fusion des transitions t_1 et t_3 (t_{13}) et de t_2 et t_4 (t_{24})



Composition asynchrone

Fusion des places p_2 et p_7 (p_{27})

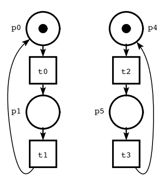




Communication par place

Deux processus A et B exécutant chacun une opération doivent se communiquer :

- A ne peut commencer qu'après la fin de B ;
- B doit attendre que A commence pour pourvoir commencer.

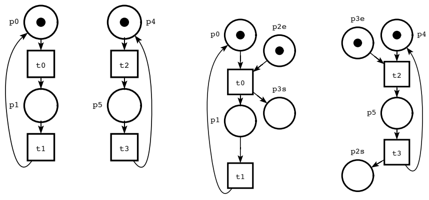


processus
indépendants

Communication par place

Deux processus A et B exécutant chacun une opération doivent se communiquer :

- A ne peut commencer qu'après la fin de B ;
- B doit attendre que A commence pour pourvoir commencer.

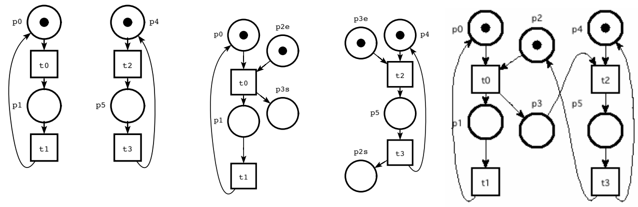


processus indépendants — modèle local

Communication par place

Deux processus A et B exécutant chacun une opération doivent se communiquer :

- A ne peut commencer qu'après la fin de B ;
- B doit attendre que A commence pour pourvoir commencer.



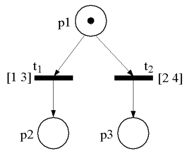
processus indépendants — modèle local — modèle global

Réseau de Petri et le temps

Plusieurs extensions de RdP pour prendre en compte le temps :

- temps associé aux arcs,
- temps associé aux places,
- temps associé aux transitions (RdP temporel, RdP temporisé)

Réseaux de Petri temporels (Merlin, 1974)



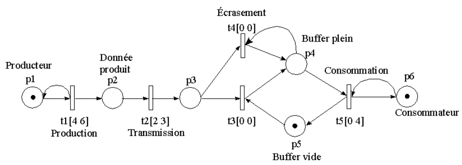
Réseaux de Petri t-temporels

Un réseau de Petri t-temporel $< N, M_0, I >$ est défini par :

- un réseau de Petri $N =< P, T, Pre, Post >$,
- un marquage initial M_0,
- une fonction intervalle statique I :

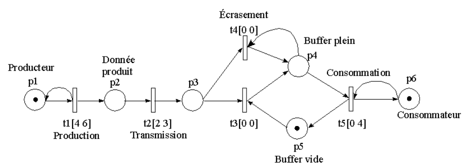
$$I : T \rightarrow (Q^+ \cup 0) \times (Q^+ \cup \infty)$$

Protocole unidirectionnel de transfert de données :



Réseaux de Petri t-temporels

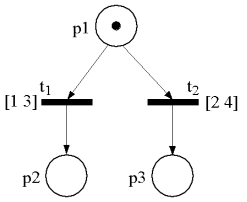
Intervalle temporel $I(t_i) = [a_i, b_i]$: dates de tir possibles de t_i à partir de sa date de sensibilisation.



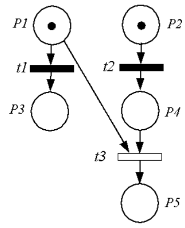
- date de sensibilisation d'une transition t_i
- date de début et de fin de l'intervalle de tir,
- date de franchissement effectif de t_i.

Sémantique

- si plusieurs transitions franchissables $I(t_i) = [a_i, b_i]$: franchir l'une d'elles avant la fin de l'intervalle de tir des autres transitions.
- tir de t_1 avant t_2 ($b_2 = 4$) $\longrightarrow$ donc, t_1 [1 3].
- tir t_2 avant t_1 ($b_1 = 3$) $\longrightarrow$ donc, t_2 [2 3]

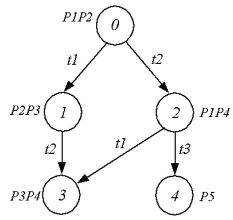


Graphe de marquage classique



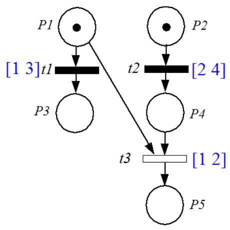
Réseau de Petri atemporel

analyse $\longrightarrow$



Graphe de marquage

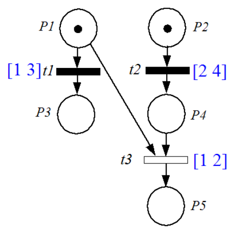
Graphe de classes



Réseau de Petri temporel

- Prise en compte du temps :
  - Nombre infini d'états (marquage + temps)
  - Nombre infini de séquences
- Il faut :
  - Regrouper les états en un nombre fini de classes : *oublier* une partie du passé.
  - Classe $\mathcal{C}$: donne les intervalles de tir et les contraintes temporelles que doivent vérifier les transitions vis-à-vis des franchissements passés.

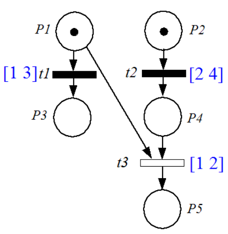
Graphe de classes



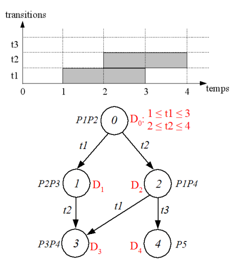
Réseau de Petri temporel

- **État** : $\{M, I(t)\}$
  - M : Marquage
  - $I(t)$: Fonction temporelle
- **Classe d'états** : composée par tous les états que sont atteignables par une séquence de tir.
- Plusieurs définitions de classe, selon le type de propriétés à prouver :
  - mode Linear, (B.Berthomieu)
  - mode Arborescent, (B.Berthomieu)
  - mode C, (J. Cardoso, R. Valette)

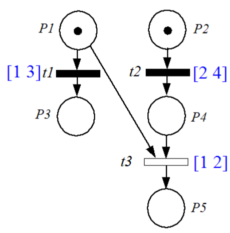
Mode linéaire



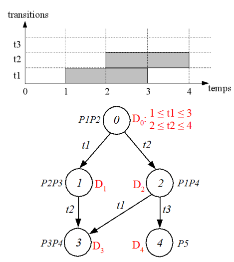
Réseau de Petri temporel



Mode linéaire

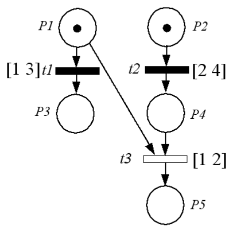


Réseau de Petri temporel

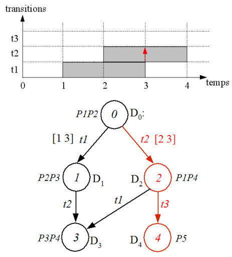


Tous ces états sont-ils atteignables ! ?

Mode linéaire

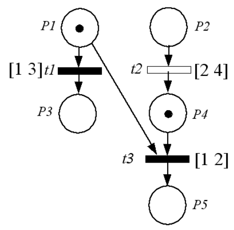


Réseau de Petri temporel

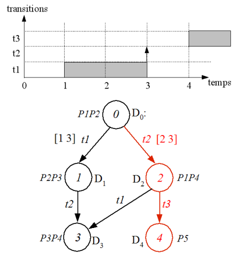


Si $t2$ est franchie au temps 3, $t3$ est-elle encore franchissable ?

Mode linéaire

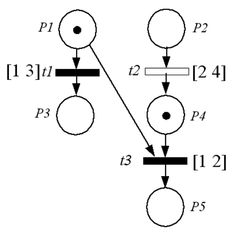


Réseau de Petri temporel

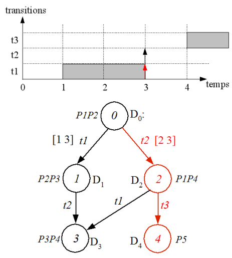


Si $t2$ est franchie au temps 3, $t3$ est-elle encore franchissable ?

Mode linéaire

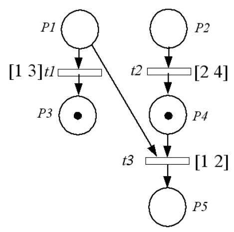


Réseau de Petri temporel

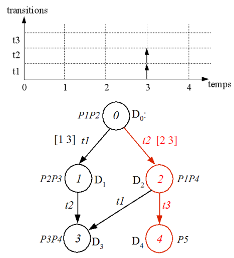


Si $t2$ est franchie au temps 3, $t3$ est-elle encore franchissable ?

Mode linéaire

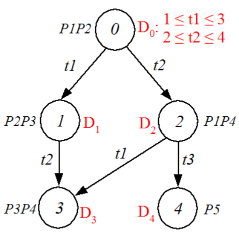


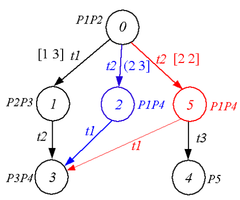
Réseau de Petri temporel



Si $t2$ est franchie au temps 3, $t3$ est-elle encore franchissable ? Non !

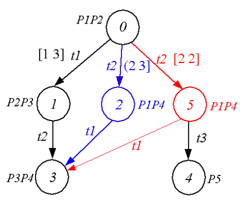
Mode arborescent

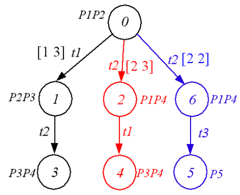




- Problème branchement : distinguer les états dans le futur.
- Problème branchement : résolu
- Mais encore problème de chemin : distinguer les états dans le passé.

Mode C



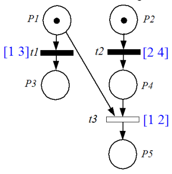


- Problème branchement : résolu
- Mais encore problème de chemin : distinguer les états dans le passé.
- Problème branchement : résolu
- Problème de chemin : résolu

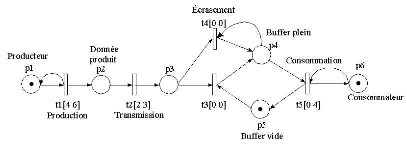
Mode linéaire

Graphe de classes :

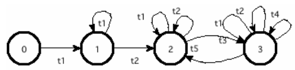
- noeuds (classes C_i) :
  - états avec le même marquage,
  - domaine temporel (union des domaines temporels des états) :
    - intervalle de temps des transitions sensibilisées
    - contraintes temporelles entre couples de transitions sensibilisées (mémoire temporelle depuis la classe où elles étaient sensibilisées) ;
- arcs (C_i, C_j) : intervalle de tir de t, avec $C_i \xrightarrow{t} C_j$



Protocole unidirectionnel de transfert de données



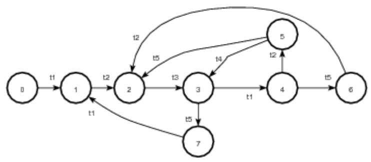
RdP sans le temps : Graphe de couverture



0 : p1 p5 p6
1 : p1 p2*w p5 p6
2 : p1 p2*w p3*w p5 p6
3 : p1 p2*w p3*w p4 p6

Protocole unidirectionnel de transfert de données

Les classes (marquage + domaine temporel) :



C0, p1 p5 p6 , t_1=[4,6]
C1, p1 p2 p5 p6 , t_1=[4,6]; t2 =[2,3]
C2, p1 p3 p5 p6 , t_1=[1,4], t_3=[0,0]
C3, p1 p4 p6 , t_1=[1,4], t_5=[0,4]

C4, p1 p2 p4 p6 , t_1=[4,6], t_2=[2,3], t_5=[0,3]
C5, p1 p3 p4 p6 , t_1=[1,4], t_4=[0,0], t_5=[0,1], $[t_1\text{-}t_5]_5$ = [1,6]
C6, p1 p2 p5 p6 , t_1=[1,6], t_2=[0,3], $[t1\text{-}t2]_6$ = [1,4]
C7, p1 p5 p6 , t_1=[0,4]

Systèmes Hybrides
IN310 - Modèles de systèmes embarqués

Charles Lesire-Cabaniols (ONERA / DCSD)
charles.lesire@onera.fr

3A-SEM - 2010-2011

Types de variables

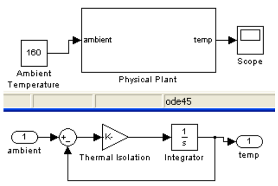
Le modèle mathématique d'un système est caractérisé par :

- la nature de ses variables d'état :
  - variables continues : prennent leurs valeurs sur le domaine des réels $\mathcal{R}$.
  - variables discrètes : prennent leurs valeurs sur un domaine représenté par un ensemble dont le nombre d'éléments est fini (ex : les entiers naturels $\mathcal{N}$, variables booléennes)/
- la nature de la variable indépendante qui représente le temps

Types de systèmes

Systèmes continus

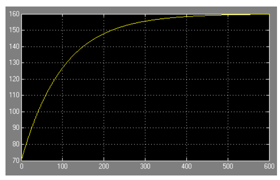
- temps : variable continue (temps dense)
- variables d'état continues, évolution dictée par le temps
- équations algébro-différentielles $\dot{x}(t) = Ax(t) + Bu(t)$, transformée de Laplace
- Ex : température d'une pièce



Types de systèmes

Systèmes échantillonnés

- temps : variable discrète $\theta_0, \theta_1 \dots \theta_{n-1}\ \theta_n, \theta_{n+1} \dots$
- variables d'état continues (*observées* à θ_i)
- équations aux différences $X_{k+1} = A_k.X_k + B_k U_k$, transformée en Z



Types de systèmes

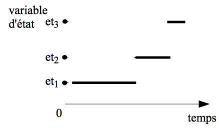
Systèmes à événements discrets

- représentés par une suite d'*événements discrets* (ex : un plan)
- temps : relation de précédence
- variables d'état discrètes : valeur x(k+1) calculé directement à partir de x(k), sans considérer le temps (fonction des événements)
- automates, réseaux de Petri
- ex : nombre de pièces dans un système de manufacture

Types de systèmes

Systèmes discrets

- temps : variable continue (temps dense)
- variables d'état discrètes (ex : machine libre ou occupée, ventilateur ON/OFF)
- automates (temporisés), réseaux de Petri (temporels)



Types de systèmes

Systèmes hybrides

- évolution à la fois en fonction
  - du temps continu et
  - des événements discrets
- variables d'état continues et variables d'état discrètes
- automates hybrides, réseaux de Petri hybrides ; Simulink et StateFlow

Systèmes Hybrides

- Evolution technologique :
  - systèmes distribués, réseaux : sous-systèmes interconnectés ;
  - 98% des microprocesseurs sont intégrés sur des systèmes physiques : BMW (72 μP en réseau), Boeing 777 (1280 μP en réseau) ;
  - avancées sur les technologies de capteurs et d'actionneurs
- 2 visions :
  - Théorie du Contrôle : systèmes continus, controllabilité, stabilité, atteignabilité, robustesse, . . .Résultats en modélisation : switched control system, supervisory control system, piecewise affine systems (PWA)
  - Informatique : systèmes de état-transition, composition et abstraction, concurrence, . . .Résultats en modélisation : automates hybrides, réseaux de Petri hybrides

Systèmes Hybrides

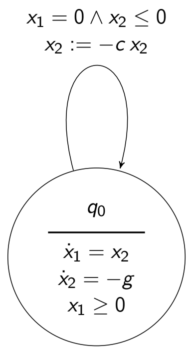
- juin 1991 : First workshop on Hybrid Systems, R.L. Grossman and A. Nerode, Cornell University, USA
- oct. 1992 : 2nd workshop, Technical University of Lyngby, Danemark
- 1993 : Hybrid Systems (Grossman et al.), Lecture Notes in Computer Science
- avril 1998 : First Hybrid Systems : Computationand Control (HSCC) (Henzinger and Sastry), Berkeley
- 2003 : IFAC conference on Analysis and Design of Hybrid Systems (ADHS), France
- de nombreux problèmes ouverts : modélisation, analysé, vérification, synthèse de controleur, simulation, génération de code, complexité, . . .

Automates hybrides

- Les automates temporisés décrivent un type de systèmes hybrides
- Automates hybrides pour représenter des horloges asynchrones !
- À l'origine : Alur, Henzinger, Sifakis, Yovine, ...
- École française performante : Sifakis, Yovine, Maler (VeriMAG), Asarin (LIAFA / Paris 7)
- Inspiration d'un cours de Claire Tomlin (`http://www.stanford.edu/class/aa278a/`)
- Références, approfondissemnets → "Google is your friend !"

Exemples

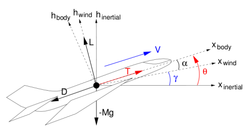
Bouncing ball

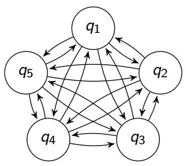


- x_1 : position verticale de la balle ;
- x_2 : vitesse vertical de la balle ;
- g : accélération ;
- $c \in [0, 1]$: coefficient de restitution ;
- transitions discrètes lors des rebonds ;
- Propriétés :
  - non-bloquant ;
  - si $c < 1$, l'automate est Zénon.

Exemples

Pilote automatique





- Équations de la dynamique du vol,
- Modes de vol, i.e. stratégies de contrôle.

Automates Hybrides

Un Automate hybride (autonome) est défini par :

- un ensemble d'états discrets Q,
- un espace d'état continu $X \subset \mathbb{R}^n$,
- un ensemble d'état initiaux $Init \subset Q \times X$,
- des invariants $Inv \subset Q \times X$,
- une dynamique continue $f : Q \times X \to X$,
- une dynamique discrète $R : Q \times X \to 2^{Q \times X}$.

Comportement dynamique d'un AH

Une succession de phases séparées par des événements

- au cours d'une phase : les variables discrètes n'évoluent pas ; les variables continues évoluent continûment dans le temps
- lors d'un événement : les variables discrètes évoluent ; les variables continues peuvent changer de valeur ($\rightsquigarrow$ discontinuité, variable non dérivable en ce point) ; la structure du modèle continu peut être modifiée ! !

Ensemble temporel hybride

Un ensemble temporel hybride (Hybrid Time Set) est une séquence d'intervalles $\tau = \{I_i\}_{i \geq 0}$ telle que $\forall\, i$:

- $I_i = [\tau_i, \tau_i']$
- $\tau_i \leq \tau_i' = \tau_{i+1}$

- $< \tau >= \sup i$ (N ou $+\infty$) ; $||\tau|| = \sum_{i=0}^{<\tau>} (\tau_i' - \tau_i)$
- si $< \tau >= \infty$ et $||\tau|| < \infty$, τ est Zénon.

Trajectoire hybride

L'exécution d'un automate hybride est une trajectoire hybride (τ, q, x) telle que :

- $(q_0, x_0) \in Init$,
- $(q_{i+1}(\tau_{i+1}), x_{i+1}(\tau_{i+1})) \in R((q_i(\tau_i'), x_i(\tau_i')))$,
- $\forall i$:
  - $q_i : I_i \rightarrow Q$ est constante (i.e., $q_i(t) = q_i(\tau_i) \forall t \in I_i$),
  - $x_i : I_i \rightarrow X$ est une solution de l'équation différentielle
    $$\dot{x}_i = f(q_i(t), x_i(t))$$
  - $\forall t \in [\tau_i, \tau_i'[$, $x_i(t) \in Inv(q_i(t))$.

Propriétés
Trajectoires

Non-bloquant
Un automate hybride est non-bloquant si $\forall (q_0, x_0) \in Init$, il existe une trajectoire infinie partant de (q_0, x_0).

Déterminisme
Un automate hybride est déterministe si $\forall (q_0, x_0) \in Init$, il existe au plus une trajectoire maximale partant de (q_0, x_0).

Propriétés
Trajectoires

Dans le cas continu :

- non-bloquant $\Leftarrow f$ continue
- déterministe $\Leftarrow f$ lipschitzienne

Dans le cas hybride :

- Plus de "souplesse" : une transition discrète peut débloquer l'automate !
- non-bloquant $\Leftarrow \forall (q, x) \in Trans \cap Reach, \exists q' \in Q \,/\, R(q, x) = (q', x')$
  - $Trans$ ensemble des états de transition : l'équation différentielle n'admet pas de solution dans l'invariant ;

    $Q \times Inv$ ouvert et f localement lipshitzienne $\Rightarrow Trans = (Q \times Inv)^c$

Propriétés
Zénon

Un automate est Zénon si pour $(q_0, x_0) \in Init$, toutes les trajectoires infinies τ depuis (q_0, x_0) sont Zénon :

- $< \tau >= \infty$
- $||\tau|| < \infty$

Du à la modélisation du système, qui est une abstraction du système réel.
Pose des problèmes pour la simulation (et donc pour la vérification).

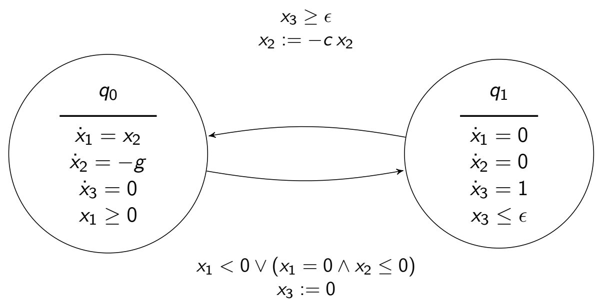
Propriétés
Zénon

Régularisation
On peut régulariser un automate hybride Zénon H en construisant une famille d'automates H_ϵ.

- $\phi : Q_\epsilon \times X_\epsilon \rightarrow Q \times X$ fait correspondre un état de H_ϵ à un état de H ;
- H_ϵ "tend vers H" quand $\epsilon \rightarrow 0$.

Propriétés

Zénon



$$\phi(q_0, (x_1, x_2, x_3)) = \phi(q_1, (x_1, x_2, x_3)) = (q, (x_1, x_2))$$

Propriétés

États

Accessibilité

Un état $(q, x) \in Q \times X$ est accessible s'il existe une trajectoire finie σ qui finie en (q, x) (i.e. $<\tau>= N < \infty$ et $(q_N(\tau'_N), x_N(\tau'_N)) = (q, x)$).

Invariants

L'ensemble $M \subset Q \times X$ est appelé invariant si $\forall (q_0, x_0) \in M$ et pour toute trajectoire σ partant de (q_0, x_0) :

$$\forall\, i,\ \forall\, t \in I_i,\ (q_i(t), x_i(t)) \in M$$

Automates hybrides rectangulaires

Rectangle

Un ensemble $R \subset \mathbb{R}^n$ est un rectangle si $R = \Pi_{i=1}^n R_i$ où R_i est un intervalle dont les bornes sont rationnelles.

Automate rectangulaire

Un automate rectangulaire est un automate hybride tel que :

- $Q = \{q_1, \ldots, q_m\}$;
- $Init = \bigcup_{i=1}^m \{q_i\} \times Init(q_i)$ où $Init(q_i)$ est un rectangle ;
- $f(q, x) = F(q)$ où $F(q)$ est un rectangle ;
- $Inv(q)$ est un rectangle.

$\longrightarrow$ Automates rectangulaires initialisés : plus grande classe d'AH pour laquelle l'accessibilité est décidable (mais PSPACE) !

Stabilité

Equilibre

L'état continu x_e est un point d'équilibre de H si :

- $f(q, x_e) = 0$ pour tout $q \in Q$;
- $R(q, x_e) \subset Q \times \{x_e\}$.

Equilibre stable

L'état continu x_e est un point d'équilibre stable si $\forall \epsilon > 0$, $\exists \delta > 0$ tel que pour toute trajectoire $(\tau, (q, x))$ partant de (q_0, x_0),

$$||x_0 - x_e|| < \delta \Rightarrow \forall t \in \tau,\ ||x(t) - x_e|| < \epsilon$$

x_e stable pour $f(q)$, pour tout $q \not\Rightarrow x_e$ stable pour H !!
même si les variables ne sont pas réinitialisés ($R(q, x) \subseteq Q \times \{x\}$).

Stabilité

Théorème de Lyapunov pour les SH

Soit H un automate tel que x_e est un point d'équilibre et $R(q, x) \in Q \times \{x\}, \forall q$. Soit D un ouvert de $\mathbb{R}^n$ tel que $x_e \in D$. x_e est stable s'il existe $V : D \to \mathbb{R}$ une fonction C^1 telle que :

- $V(x_e) = 0$
- $V(x) > 0 \forall x \in D \backslash \{(x_e)\}$
- $\frac{\partial V(x)}{\partial x} f(q, x) \leq 0, \forall x \in D, \forall q \in Q$

Système linéaire par morceaux

Pour un système linéaire par morceaux, i.e. $f(q_i, x) = A_i x$. x_e est stable s'il existe une solution P symétrique définie positive au système de LMI

$$\forall i,\ A_i^T P + PA_i < 0$$

Contrôle

Automate hybride

Un automate hybride H est défini par :

- Q, X,
- $Init \subset Q \times X$ un ensemble d'états initiaux,
- In un ensemble fini de variables d'entrées, $In = \Sigma \cup W$,
  - $\Sigma = \Sigma_U \times \Sigma_D$, avec Σ_U les entrées discrètes et Σ_D les perturbations discrètes,
  - $W = U \times D$, avec U les entrées continues (contrôlables) et D les perturbations continues (bruits),
- $f : Q \times X \times W \to \mathbb{R}^n$ un champ vectoriel,
- $Inv : Q \to 2^{X \times W}$ les invariants de H,
- $R : Q \times X \times In \to 2^{Q \times X}$ la fonction de transition,
- $Out = P \cup Y$ l'ensemble des variables de sortie, discrètes (P) et continues (Y).

Contrôle

Hypothèses

On fait les hypothèses suivantes :

- f est lipshitzienne sur X et continue sur W ;
- $\forall q,\ Inv(q)$ est un ouvert ;
- $\forall (q,x), \forall (\sigma_u, u) \in \Sigma_U \times U,\ \exists (\sigma_d, d) \in \Sigma_D \times D$ tel que

$$(x, (\sigma_u, \sigma_d), (u, d)) \in Inv(q) \ \vee \ R(q, x, (\sigma_u, \sigma_d), (u, d)) \neq \emptyset$$

Contrôle

Principe

On veut satisfaire une propriété de sûreté, i.e. calculer l'invariant contrôlé maximal $F_C \subset F \subset Q \times X$.
On appréhende le problème comme un jeu, dans lequel :

- les perturbations essaient de fuir F en
  1. faisant des sauts (discrets) hors de F,
  2. tirant (continument) le système hors de F ;
- le contrôleur essaie de rester dans F en
  1. tirant le système dans F (en évitant les sauts),
  2. faisant des sauts dans F (évitant les sorties continues).

Contrôle

Principe

- $Pre_u : 2^{Q \times X} \to 2^{Q \times X}$ prédécesseurs contrôlables : une action contrôlable peut forcer à rester dans K

$$Pre_u(K) = \{(q,x) \in K / \exists u \in In_U, \forall d \in In_D,\ (x,u,d) \notin Inv(q) \wedge R(q,x,u,d) \subset K\}$$

- $Pre_d : 2^{Q \times X} \to 2^{Q \times X}$ prédécesseurs incontrôlables : des actions incontrôlables peuvent forcer à sortir de K

$$Pre_d(K) = \{(q,x) \in K / \forall u \in In_U, \exists d \in In_D,\ R(q,x,u,d) \cap K^c \neq \emptyset\} \cup K^c$$

- $Reach : 2^{Q \times X} \times 2^{Q \times X} \to 2^{Q \times X}$: états depuis lesquels G est accessible sans atteindre E

$$Reach(G,E) = \{(q(0),x(0)) / \forall u \in \mathcal{U}, \exists d \in \mathcal{D},\ (q(t),x(t)) \in G \text{ and } (q(s),x(s)) \in Inv \setminus E\ \forall s \in [0,t]\}$$

où $(q(s), x(s))$ est la trajectoire d'équation $\dot{x} = f(q(s), x(s), u(s), d(s))$.

Contrôle

Principe

Algorithme

Point fixe de l'équation :

$$\begin{aligned} W^0 &= F \\ W^{i-1} &= W^i \backslash Reach(Pre_d(W^i), Pre_u(W^i)) \end{aligned}$$

Calcul de Pre : inversion de la fonction R.
Calcul de $Reach$: résolution d'un équation d'Hamilton-Jacobi sous contrainte.
$\rightarrow$ algorithme semi-décidable pour des automates hybrides linéaires
f linéaire en x, gardes et resets (R) polyédriques

Filtrage

À partir des observations y de l'état continu, estimer l'état hybride $(\hat{q}, \hat{x})$.
Approches :

- Filtres de Kalman étendus sur des automates hybrides probabilistes concurrents (Hofbaur & Williams, 2004)
- Filtrage particulaire sur automates hybrides (Koutsoukos *et al.*, 2006 ; Funiak & Williams, 2003 ; Pfeffer & Dearden, 2007 ; ...)
- Filtre ensembliste sur automates hybrides (Benazera, 2004).

Outils

- HyTech (U. Berkeley) : vérification de propriété temporelle (LTL) sur des automates hybrides linéaires (composés) ;
- d/dt (VeriMAG) : accessibilité dans les AH linéaires ; synthèse de contrôleur discret ;
- CHARON (U. Pennsylvanie) : hiérarchisation, simulation.