Réseaux de Petri IN310 - Modèles des SE

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

3A-SEM - 2010-2011

Introduction

Modèle formel

Analyse de propriétés

Composition

Réseau de Petri et le temps

- ▶ 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.

- 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

00000

Présentation informelle

Éléments de base

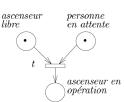
- ▶ Place : interprétée comme condition, état partiel, ensemble de ressources
- ► Transition : associée à un événement qui a lieu dans le système
- ▶ Jeton : indique que la condition associée à la place est vérifiée (ou le nombre d'éléments qui la vérifient)

Présentation informelle

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 $R = \langle P, T, Pre, Post \rangle$

- \triangleright *P* est un ensemble fini de places de dimension *n*;
- ightharpoonup 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 $N = \langle R, M \rangle$

- ▶ R est un réseau de Petri,
- M: P → N est le marquage initial (distribution de jetons dans les places)



Définitions

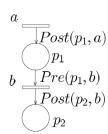
Exemple

$$ightharpoonup R = \langle P, T, Pre, Post \rangle$$

$$P = \{p_1, p_2, p_3\}$$

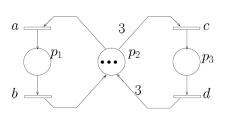
$$T = \{a, b, c, d\}$$

Post
$$(p_1, a) = 1$$
, $Pre(p_1, b) = 1$, $Post(p_2, b) = 1$



Graphe et notation matricielle

Réseau de Petri marqué $N = \langle R, M \rangle$



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

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

 $^{t}M = (0 \ 3 \ 0)$

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

Règle de fonctionnement

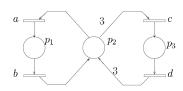
Transition sensibilisée à partir de M

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

Tir d'une transition à partir de M

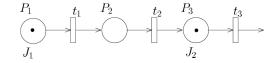
- $ightharpoonup \forall p \in P, \ M'(p) = M(p) Pre(p, t) + Post(p, t)$
- M' = M Pre(., t) + Post(., t) = M + C(., t)

- 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

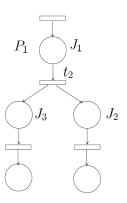
Séquence



- 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 :
 - \triangleright P_i : chariot traverse la section i,
 - t_i : passage d'un chariot d'une section à une autre;

Différentes interactions entre les processus

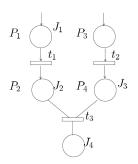
Fork



- à partir de l'activité J₁, deux activités sont crées (J₂ et J₃),
- ▶ J_2 et J_3 évoluent de façon indépendante.

Différentes interactions entre les processus

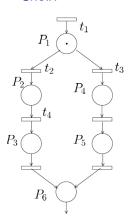
Join



- évolution indépendante de t₁ et t₂ (évolution asynchrone),
- ▶ synchronisme en *t*₃.

Différentes interactions entre les processus

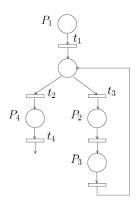
Choix



- ▶ choix entre t_2 (seq. P_2P_3) et t_3 (seq. P_4P_5) : seulement une peut être tirée;
- ▶ les 2 séquences exécuteront P₆.

Différentes interactions entre les processus

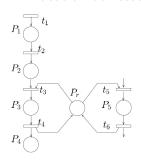
Répétition



- ▶ choix entre t_2 e t_3 ,
- ▶ répéter la séq. P₂P₃ un certain nombre de fois avant de exécuter P₄.

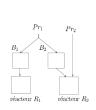
Différentes interactions entre les processus

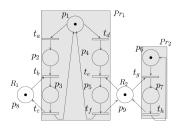
Allocation de ressources



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

- ▶ 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 t₁ et t₂ 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 t₁ et t₂ sont en conflit structurel et sont sensibilisées par le marquage *M*

$$M \geq Pre(., t_1)$$
 et $M \geq Pre(., t_2)$

▶ Parallélisme structurel : si t₁ et t₂ ne possèdent pas de place d'entrée en commun

$$\forall p \in P \quad Pre(p, t_1) Pre(p, t_2) = 0 \text{ ou } Pre(., t_1)^T \times Pre(., t_2) = 0$$

ightharpoonup Parallélisme effectif : t_1 et t_2 sont parallèles structurellement et

$$M \geq Pre(., t_1) e M \geq Pre(., t_2)$$

Séquence de tir

$$\begin{pmatrix}
0 \\
3 \\
0
\end{pmatrix} \xrightarrow{a} \begin{pmatrix}
1 \\
2 \\
0
\end{pmatrix} \xrightarrow{a} \begin{pmatrix}
2 \\
1 \\
0
\end{pmatrix} \xrightarrow{b} \begin{pmatrix}
1 \\
2 \\
0
\end{pmatrix}$$

$$M_0 \qquad M_1 \qquad M_2 \qquad M_1$$

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

$$s: T \to \mathbb{N}$$

 $t \mapsto \text{nombre d'occurrences de } t \text{ dans } s$

Séquence de tir

- Équation fondamentale : 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. $M \xrightarrow{s} M'$?

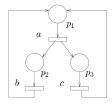
Réseau borné

Introduction

▶ 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

Introduction

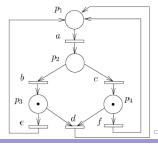
► Transition quasi-vivante :

$$\exists s / M_0 \stackrel{s}{\longrightarrow} M$$
 et $M \stackrel{t}{\longrightarrow}$

► Transition vivante :

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

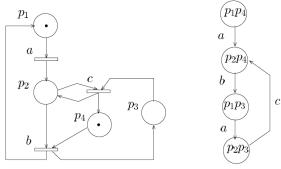
Réseau vivant ssi toutes ses transitions sont vivantes



Réseau réinitialisable

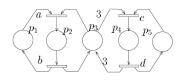
► Réseau marqué réinitialisable s'il est possible de revenir au marquage initial à partir de n'importe quel marquage :

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



Composantes conservatives

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



- ▶ 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
- $M(p_1) + M(p_2) = M_0(p_1) + M_0(p_2) = 1$
- $M(p_2) + M(p_3) + 3.M(p_4) = 3$
- $M(p_4) + M(p_5) = 1$

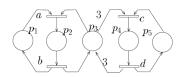
Remarque

- 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
- ▶ $M' = M \Rightarrow C s = 0$ s composante répétitive
- Dans l'exemple :

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

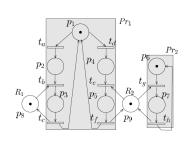


Invariants de transition

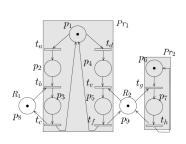
- Séquences s; obtenues à partir du vecteur s
- ▶ Pour s_1 on peut avoir les invariants $s_{11} = ab$ et $s_{12} = ba$
- ▶ Il faut calculer $M \xrightarrow{ab}$ et $M \xrightarrow{ba}$ pour vérifier !

Remarque

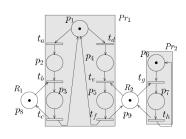
- ▶ Composante répétitive : dépend seulement de la structure !
- Invariant de transition : dépend de la structure et du marquage



- ► $f^1 = {}^t \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$: fabrication de Pr_1
- $f^2 = {}^t (0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0) :$ fabrication de Pr_1
- ► $f^3 = {}^t \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$: état du réacteur R_1
- ► $f^4 = {}^t (0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1)$: état du réacteur R_2



- ▶ $M(p_1) + M(p_2) + M(p_3) + M(p_4) + M(p_5) = 1$: un seul état pour Pr_1 (attente, B_1 , B_2 , R_1 , R_2)
- ▶ $M(p_6) + M(p_7) = 1$: un seul état pour Pr_2 (attente ou R_2)
- $M(p_3) + M(p_8) = 1$: R_1 libre ou prod. Pr_1
- $M(p_5) + M(p_7) + M(p_9) = 1 : R_2 \text{ libre}$ ou prod. Pr_1 ou Pr_2

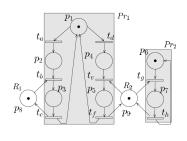


- ▶ $s^1 = {}^t (1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0)$:

 Pr₁ passe par le buffer et est prod. par

 R₁ ; un nouveau cycle peut
 recommencer ;
- $s^3 = {}^t (0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1) :$ début de prod. de Pr_2 et fin

Exemple: Système par lot



Seules séquences effectivement réalisables :

$$ightharpoonup s_1 = t_a t_b t_c$$

$$ightharpoonup s_2 = t_d t_e t_f$$

$$ightharpoonup s_3 = t_g t_h$$

- 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.

Arbre de couverture

- ▶ On part du marquage initial M_0 ,
- ightharpoonup 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 $\mathcal{A}(\mathcal{R},M)$

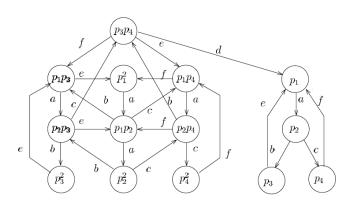
- ▶ Réseau k-borné $\Leftrightarrow \mathcal{A}(\mathcal{R}, M)$ borné
- ▶ Réseau réinitialisable $\Leftrightarrow \mathcal{A}(\mathcal{R}, M)$ fortement connexe

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

▶ Réseau vivant $\Leftrightarrow \mathcal{A}(\mathcal{R}, M)$ fortement connexe et 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 par énumération des marquages



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.

Invariants de place

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

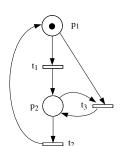
$$f(p)M(p) \leq {}^t f M_0, \qquad M(p) \leq \frac{{}^t f M_0}{f(p)}$$

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



- ► 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

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 :

- \blacktriangleright 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.

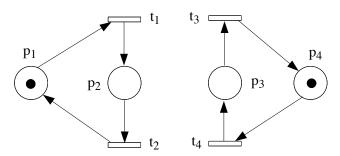
Composition

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

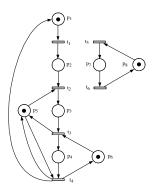
Composition synchrone

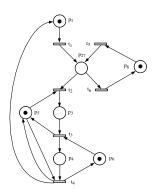
Fusion des transitions t_1 et t_3 (t_{13}) et de t_2 et t_4 (t_{24})



Composition asynchrone

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

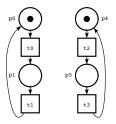




Communication par place

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.



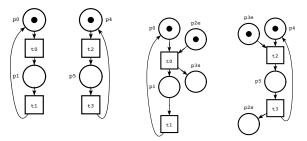
processus indépendants



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

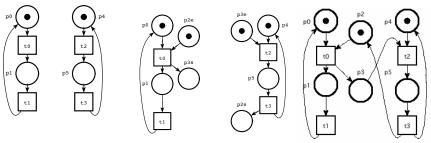
modèle local



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

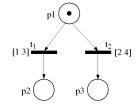
modèle global

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)



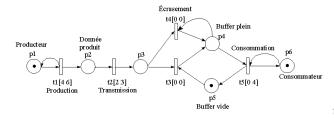
Réseaux de Petri t-temporels

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

- ▶ un réseau de Petri $N = \langle P, T, Pre, Post \rangle$,
- un marquage initial M_0 ,
- une fonction intervalle statique I :

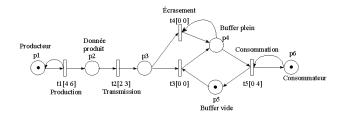
$$I: T \to (Q^+ \cup 0) \times (Q^+ \cup \infty)$$

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.

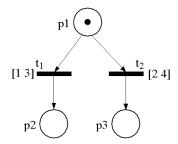


- \triangleright date de sensibilisation d'une transition t_i
- date de début et de fin de l'intervalle de tir,
- ▶ date de franchissement effectif de ti.

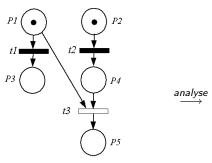


Sémantique

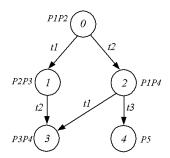
- ▶ si plusieurs transitions franchissables $I(t_i) = [a_i, b_i]$: franchir l'une d'elles avant la fin de l'intervalle de tir des autres transitions.
- ▶ tir de t_1 avant t_2 $(b_2 = 4)$ \rightarrow donc, t_1 [13].
- ▶ tir t_2 avant t_1 $(b_1 = 3)$ → donc, t_2 [23]



Graphe de marquage classique

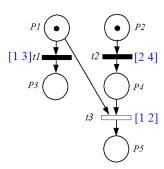


Réseau de Petri atemporel



Graphe de marquage

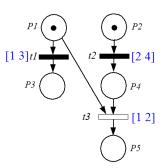
Graphe de classes



Réseau de Petri temporel

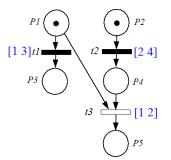
- ▶ Prise en compte du temps :
 - Nombre infini d'états (marquage + temps)
 - Nombre infini de séquences
- Il faut :
 - Regrouper les états en un nombre fini de classes : oublier une partie du passé.
 - ► Classe C: donne les intervalles de tir et les contraintes temporelles que doivent vérifier les transitions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

Graphe de classes

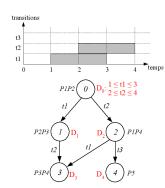


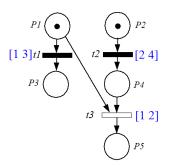
Réseau de Petri temporel

- ▶ 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)

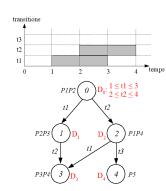


Réseau de Petri temporel





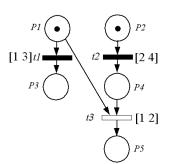
Réseau de Petri temporel



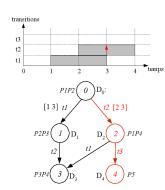
Tous ces états sont-ils atteignables!?

Graphe de classes

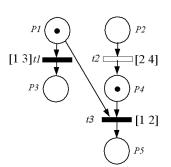
Introduction



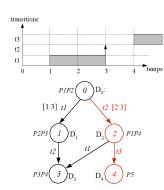
Réseau de Petri temporel



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



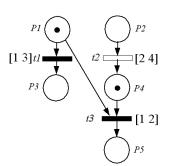
Réseau de Petri temporel



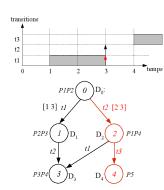
Si t2 est franchie au temps 3, t3 est-elle encore franchissable?

Graphe de classes

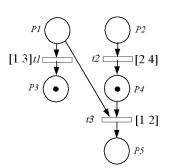
Introduction



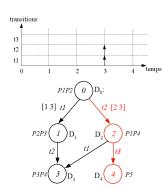
Réseau de Petri temporel



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

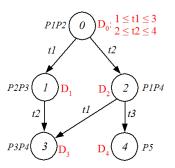


Réseau de Petri temporel

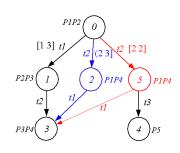


Si t2 est franchie au temps 3, t3 est-elle encore franchissable? Non!

Mode arborescent

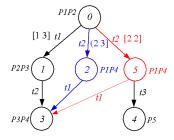


Problème branchement : distinguer les états dans le futur.

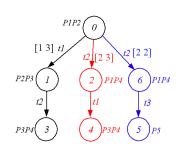


- Problème branchement : résolu
- Mais encore problème de chemin : distinguer les états dans le passé.

Mode C



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



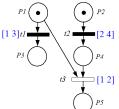
- Problème branchement : résolu
- ▶ Problème de chemin : résolu



Mode linéaire

Graphe de classes :

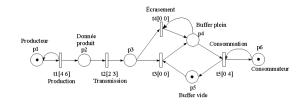
- ▶ noeuds (classes *C_i*) :
 - états avec le même marquage,
 - domaine temporel (union des domaines temporels des états) :
 - intervalle de temps des transitions sensibilisées
 - contraintes temporelles entre couples de transitions sensibilisées (mémoire temporelle depuis la classe où elles étaient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



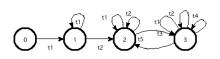
Graphe de classes

Introduction

Protocole unidirectionnel de transfert de données



RdP sans le temps : Graphe de couverture



0 : p1 p5 p6

1 : p1 p2*w p5 p6

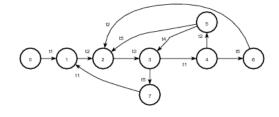
2 : p1 p2*w p3*w p5 p6

3 : p1 p2*w p3*w p4 p6



Protocole unidirectionnel de transfert de données

Les classes (marquage + domaine temporel) :



```
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-t_3]<sub>5</sub> = [1,6]

C6, p1 p2 p5 p6 , t_1=[1,6], t_2=[0,3], [t_1-t_2]<sub>6</sub> = [1,4]

C7, p1 p5 p6 , t_1=[0,4]
```

C0, p1 p5 p6, $t_1=[4,6]$