Partie 2
Chapitre 1

CentraleSupélec

Modélisation

Transparents du cours

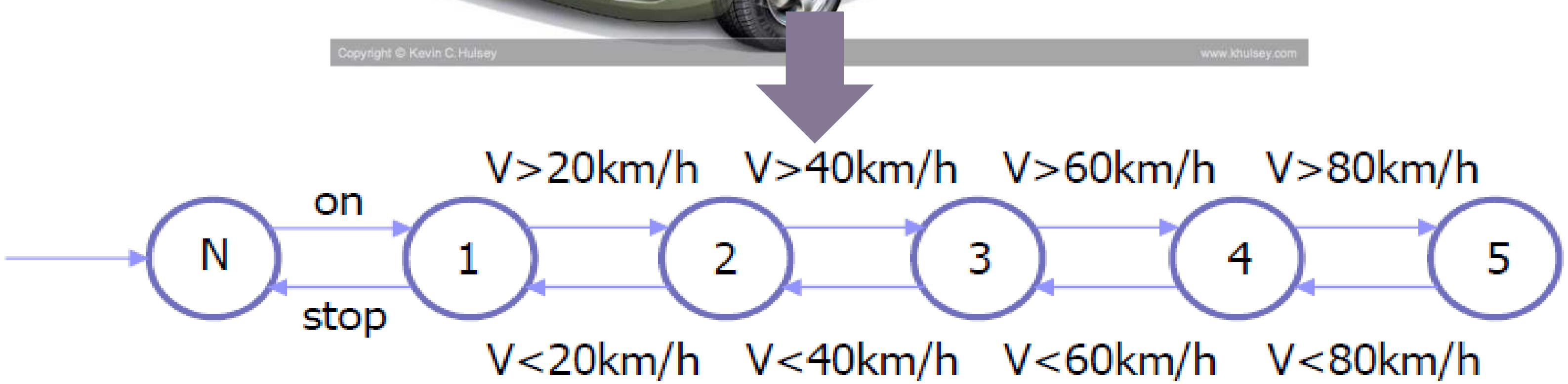
Equipe pédagogique : M. Cicic, S. Font, V. Letort-Le Chevalier, H. Lhachemi, C. Stoica Maniu, G. Sandou, C. Vlad

Cristina.Vlad@centralesupelec.fr

modelisation.cours@centralesupelec.fr

Plan du cours

- Introduction
- Partie I : Modélisation des systèmes à état continu
- Partie II :



Plan du cours

- Introduction
- Partie I : Modélisation des systèmes à état continu
- Partie II : Modélisation des systèmes à état discret
  - Introduction
  - Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation des systèmes à évènements discrets temporisés
- Partie III : Méthodes pour l'analyse, l'identification paramétrique et l'évaluation des modèles

Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets (SED) non temporisés
  - Modélisation par automates non temporisés
  - Modélisation par réseaux de Petri non temporisés
- Chapitre 2 : Modélisation des SED temporisés
  - Modélisation par automates temporisés, automates temporisés avec gardes, systèmes hybrides et réseaux de Petri temporisés

II. Introduction

- Partie I : **Approche traditionnelle** permettant la prise en compte des aspects continus dans la modélisation des systèmes

  → systèmes à état continu (temps continu/discret)

  représentation d'état, fonction de transfert, …

  Contexte plus réaliste : systèmes composés de sous-processus continus démarrés / reconfigurés / arrêtés par une commande logique, à état discrets

- Partie II : Prise en compte des aspects séquentiels

  → systèmes à évènements discrets (SED)

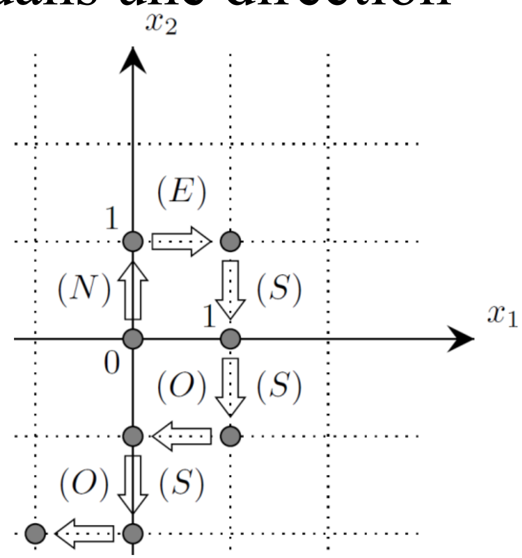
  arrivée d'un signal, fin d'une tâche précédente, libération d'une ressource, changement d'un mode opératoire…

  ex : réseaux informatiques, systèmes de production et de logistique

II. Introduction

Exemple introductif : marche aléatoire – jeu à 4 joueurs

- Chaque joueur peut faire progresser un mobile dans une direction
  - joueur N : $x_2(t)$ croissant
  - joueur E : $x_1(t)$ croissant
  - joueur S : $x_2(t)$ décroissant
  - joueur O : $x_1(t)$ décroissant



- Position initiale (0,0), état du jeu $\left(x_1(t), x_2(t)\right)$
- Evolution du jeu entièrement définie

par une *séquence d'évènements* ➡ e.g. N→E→S→S→O→S→O

- Temps non explicite (pas d'influence de la date à laquelle se produit un évènement sur l'évolution du mobile)

➡ représentation par SED non temporisé

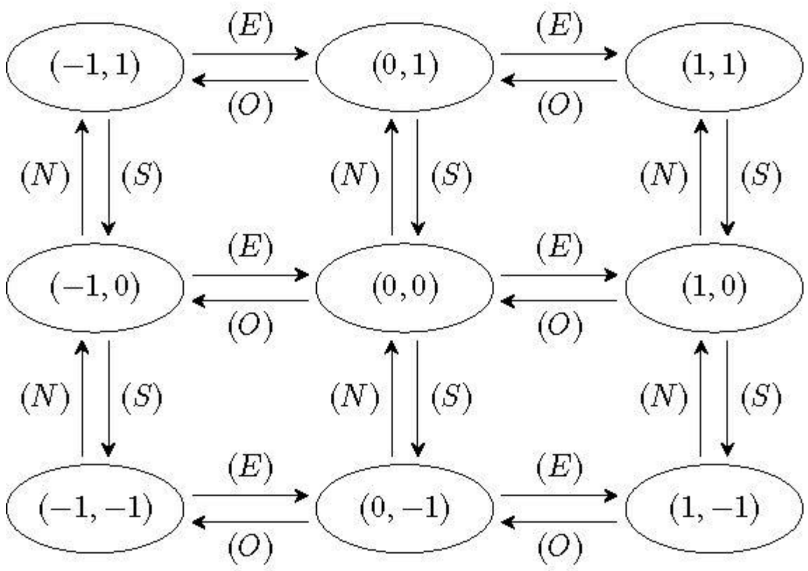
Exemple introductif : marche aléatoire – jeu à 4 joueurs

- Si le mobile doit rester dans la zone définie par

$$-1 \leq x_1(t) \leq 1$$

$$-1 \leq x_2(t) \leq 1$$

Hypothèse : pas d'appui simultané par 2 joueurs



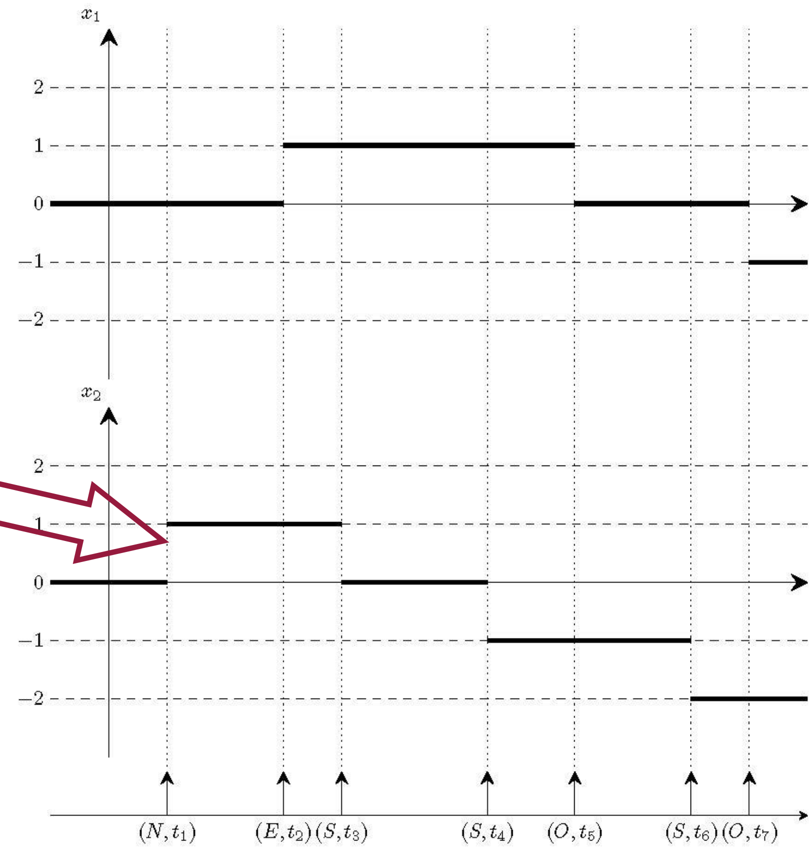
- Taille finie de l'espace des états

→ représentation de SED non temporisé par un automate fini

II. Introduction

Exemple introductif : marche aléatoire – jeu à 4 joueurs

- Si temps limité, avec délai maximum de réflexion par joueur



Ex : état initial (0,0) à l'instant t_0

➡ état (0,1) à l'instant t_1

➡ prise en compte des instants de temps dans le jeu

➡ représentation de SED temporisé par un chronogramme

II. Introduction

Exemple introductif : marche aléatoire – jeu à 4 joueurs

- Si la pression du bouton n'engendre plus un saut dans l'espace, mais une translation rectiligne uniforme dans leur direction d'action
  - si le joueur N appuie sur son bouton à l'instant t_0 (avec $a > 0$ fixé)

$$\dot{x}_1(t) = 0 \rightarrow x_1(t) = x_1(t_0)$$
$$\dot{x}_2(t) = +a \rightarrow x_2(t) = x_2(t_0) + a(t - t_0)$$

modèle jusqu'à l'action d'un autre joueur

- Arrivée d'un évènement ➡ modification instantanée de la dynamique continue du système

➡ représentation de SED temporisé par un système hybride

- Si changement de l'évolution du système à une date aléatoire par une 5$^{\text{ème}}$ personne

➡ représentation de SED par un automate stochastique

II. Introduction

- **Définition**
  - Système à Evénements Discrets (SED) : système défini par un *espace d'états discrets* et des *évolutions* (trajectoires) fondées sur une *succession d'états et de transitions*
    - Espace d'**état** : état à t_0 suffisant pour en prédire le comportement futur dès lors que l'on connait les entrées
      - Espace d'état discret : nombre dénombrable (fini ou infini) des états possibles
    - **Evènements** discrets : à des instants discrets du temps
    - Symboles (évènements) : éléments d'un alphabet, associés aux **transitions**

- **Classification** de SED
  - SED non-temporisés
    - automates à états finis, réseaux de Petri non-temporisés, Grafcet, etc.
  - SED temporisés déterministes
    - automates temporisés, réseaux de Petri temporisés, algèbre min-max, etc.
  - SED temporisés stochastiques fondés sur des hypothèses statistiques
    - chaînes de Markov, réseaux de files d'attente, processus semi-markoviens généralisés, réseaux de Petri stochastiques, etc.

Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation par automates non temporisés
    - Définition, propriétés et composition des automates
    - Analyse d'un SED non temporisé via un automate
  - Modélisation par réseaux de Petri non temporisés
- Chapitre 2 : Modélisation des systèmes à évènements discrets temporisés

II.1. Définition des automates non temporisés

Poly 7.1.1

- SED non temporisés : seule la séquence des évènements est considérée, les dates d'occurrences n'ayant pas d'effet
- *Automate à états finis* ou *automate fini* $\mathcal{A} = (Q, \Sigma, \phi, Q_0)$
  - Q : ensemble fini de sommets (i.e. les états discrets)
  - Σ : ensemble de symboles (évènements) appelés alphabet
  - ϕ : relation de transition définie comme
    - $\phi : Q \times \Sigma \to Q$ t.q. $q_2 = \phi(q_1, \sigma)$
    - ou comme une partie de $Q \times \Sigma \times Q$ t.q. $(q_1, \sigma, q_2) \in \phi$
  - $Q_0 \subset Q$: ensemble d'états initiaux
  - $Q_f \subset Q$: ensemble d'états marqués (facultatif)

→ **Exemple : loueur de 2 véhicules électriques**

II.1. Définition des automates non temporisés

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c , n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule

II.1. Définition des automates non temporisés

Poly 7.1.1

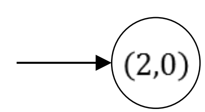
- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :

    nombre de véhicules déchargés

    - Etat $X = (n_c , n_d)$

      nombre de véhicules disponibles chargés

    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule



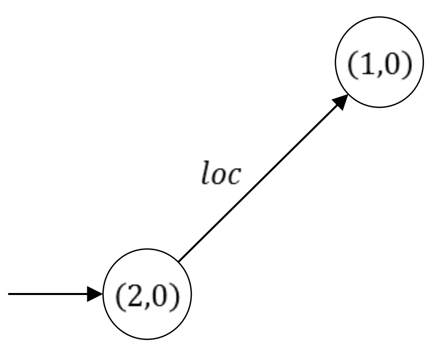
II.1. Définition des automates non temporisés

Poly 7.1.1

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c, n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule



CentraleSupélec

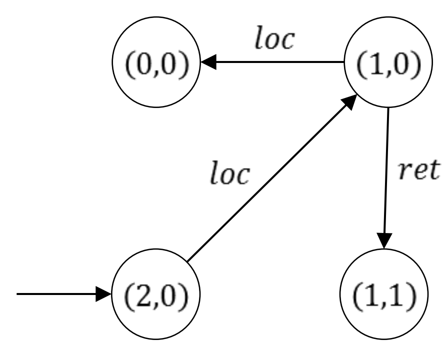
II.1. Définition des automates non temporisés

Poly 7.1.1

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c , n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule

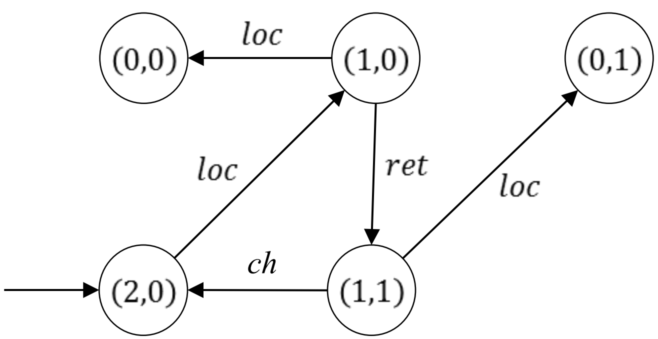


II.1. Définition des automates non temporisés

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c , n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule

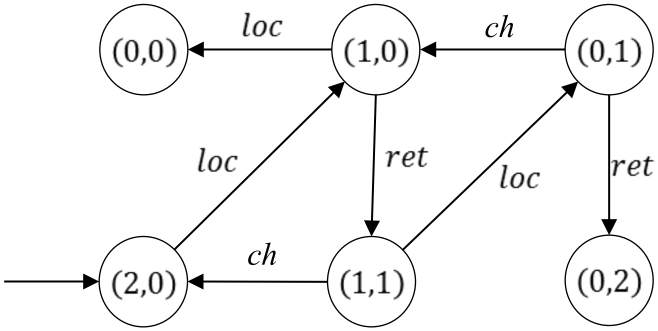


II.1. Définition des automates non temporisés

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c , n_d)$

      n_c : nombre de véhicules disponibles chargés

      n_d : nombre de véhicules déchargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule



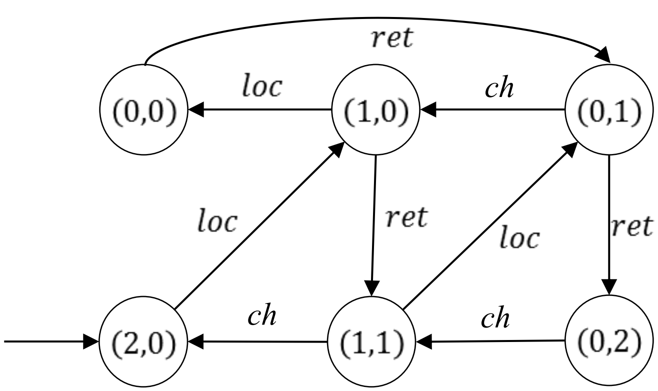
II.1. Définition des automates non temporisés

Poly 7.1.1

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c, n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule



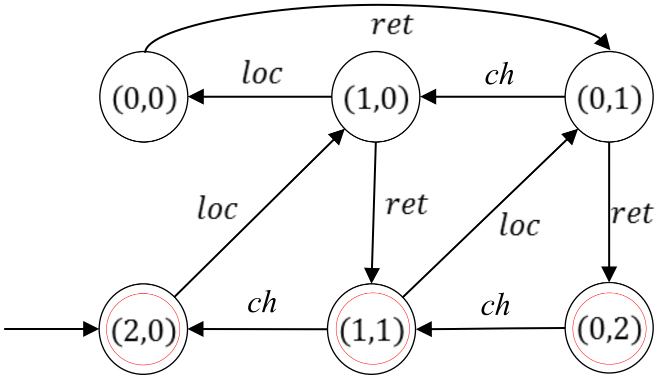
II.1. Définition des automates non temporisés

Poly 7.1.1

- Exemple : loueur de 2 véhicules électriques
  - Hypothèses :
    - Etat $X = (n_c , n_d)$

      nombre de véhicules déchargés

      nombre de véhicules disponibles chargés
    - Initialement le loueur possède 2 véhicules chargés prêt à la location
    - Véhicules loués un par un
    - Un véhicule loué nécessite forcément une recharge avant de pouvoir être loué à nouveau
    - Tous les véhicules sont retournés le soir
  - Evènements :
    - *loc* : location d'un véhicule
    - *ret* : retour d'un véhicule
    - *ch* : recharge d'un véhicule



II.1. Définition des automates non temporisés

- Exemple : loueur de 2 véhicules électriques
  - $Q = \{(0,0), (0,1), (1,0), (1,1), (2,0), (0,2)\}$
  - $\Sigma = \{loc, ch, ret\}$
  - $\phi = \{((2,0), loc, (1,0)),\ ((1,0), loc, (0,0)),\ ((0,0), ret, (0,1)),$
    $((0,1), ch, (1,0)),\ ((1,0), ret, (1,1)),\ ((1,1), ch, (2,0)),$
    $((0,1), ret, (0,2)), ((0,2), ch, (1,1)), ((1,1), loc, (0,1))\}$
  - $Q_0 = \{(2,0)\}$
  - $Q_f = \{(2,0), (1,1), (0,2)\}$

II.1. Propriétés des automates non temporisés

Poly 7.1.2

- Caractère **déterministe / non déterministe (stochastique)**
  - *Automate déterministe* : un seul état initial $Q_0 = \{q_0\}$ et

  à partir d'un état, un même symbole conduit dans un seul état

  $$\forall q \in Q,\ \forall \sigma \in \Sigma,\ \left((q,\ \sigma,\ q_1) \in \phi \ \&\ (q,\ \sigma,\ q_2) \in \phi \right) \Rightarrow q_1 = q_2$$

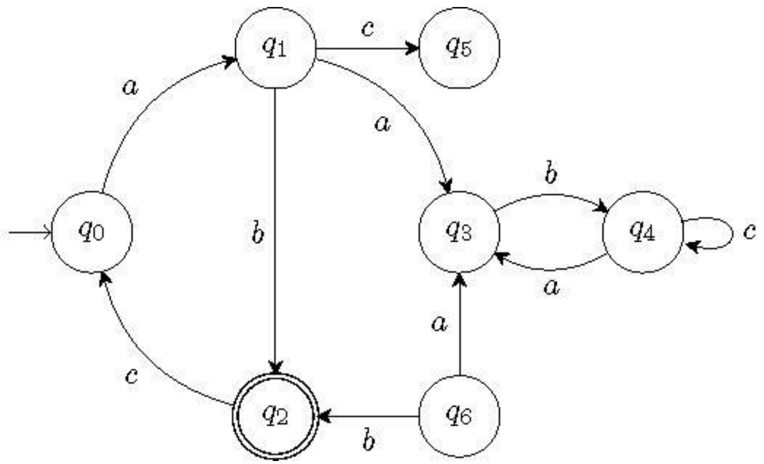
- **Accessibilité, co-accessibilité, complément**
  - *Etat q accessible* depuis q_0 : si $\exists\ \sigma_1\ \sigma_2 \ldots \sigma_n$ t.q. $\phi(q_0\ ,\ \sigma_1\ \sigma_2 \ldots \sigma_n) = q$
    - *Automate accessible* : dont tous les états sont accessibles
  - *Etat q co-accessible* : si $\exists\ \sigma_1\ \sigma_2 \ldots \sigma_n$ t.q. $\phi(q\ ,\ \sigma_1\ \sigma_2 \ldots \sigma_n) = q_m,\ q_m \in Q_f$
    - *Automate co-accessible* : si l'automate obtenu en éliminant la partie non co-accessible est identique à l'automate de départ

      i.e. pas de partie non co-accessible

    - *Automate élagué (émondé)* : à la fois accessible et co-accessible

II.1. Propriétés des automates non temporisés

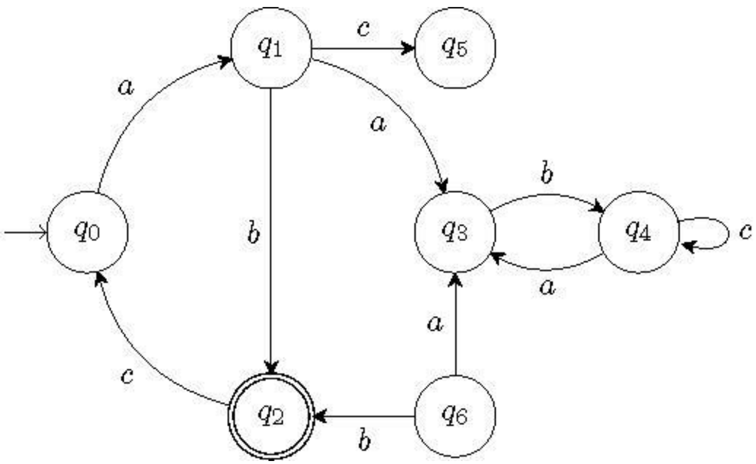
Automate initial



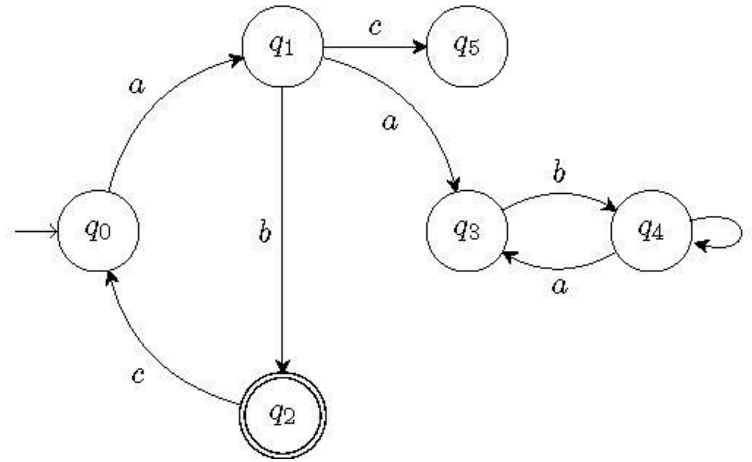
II.1. Propriétés des automates non temporisés

Poly 7.1.2

Automate initial



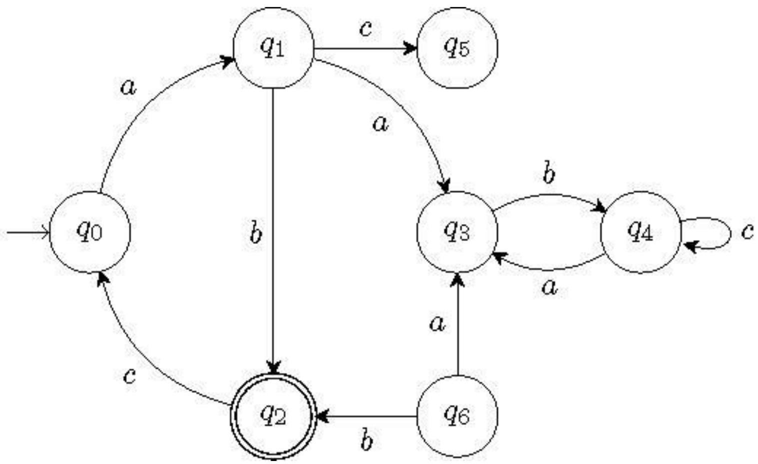
Automate accessible



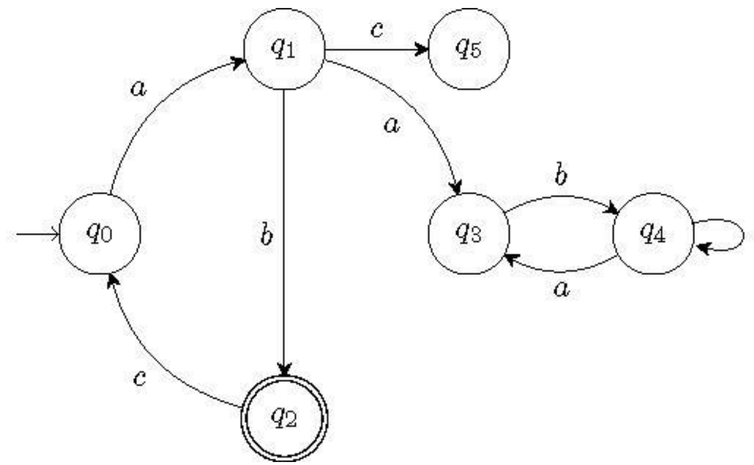
II.1. Propriétés des automates non temporisés

Poly 7.1.2

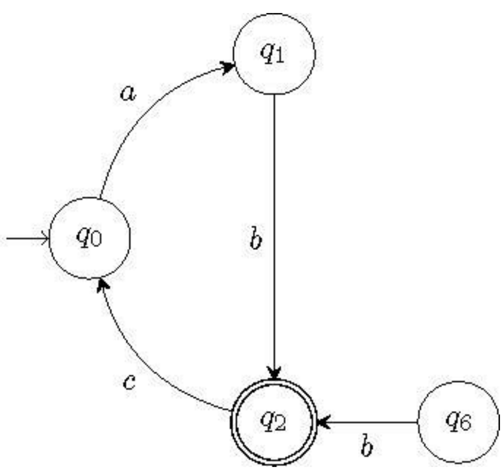
Automate initial



Automate accessible



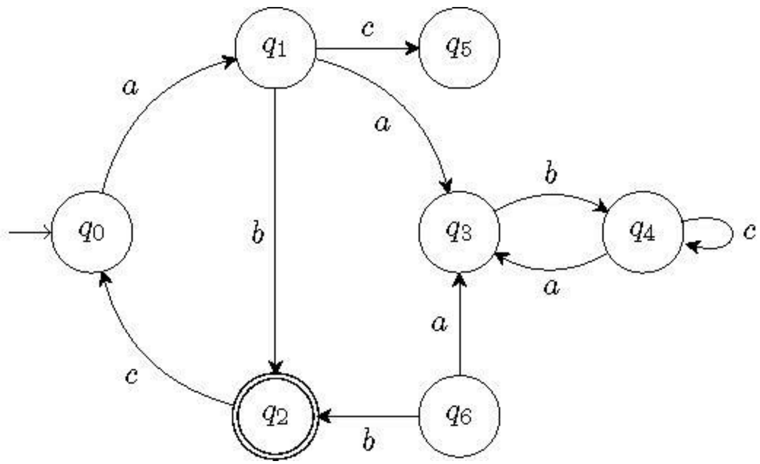
Automate co-accessible



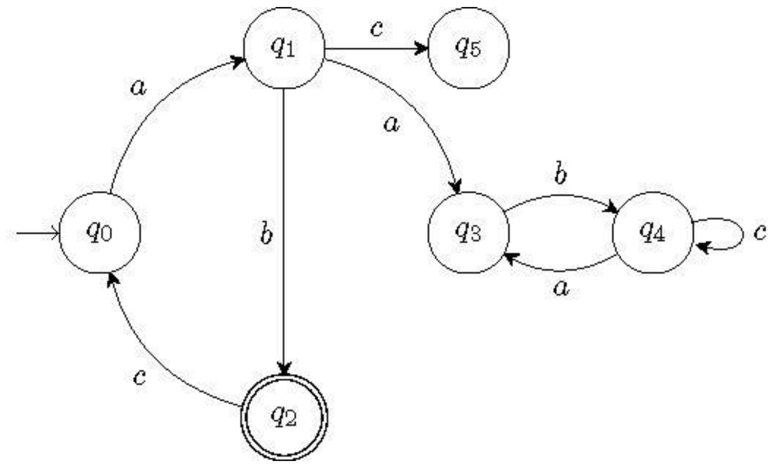
II.1. Propriétés des automates non temporisés

Poly 7.1.2

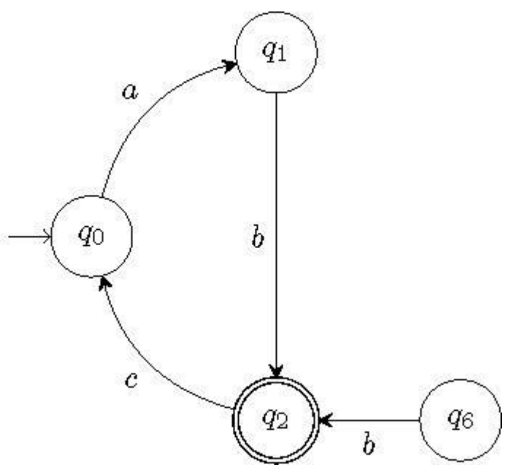
Automate initial



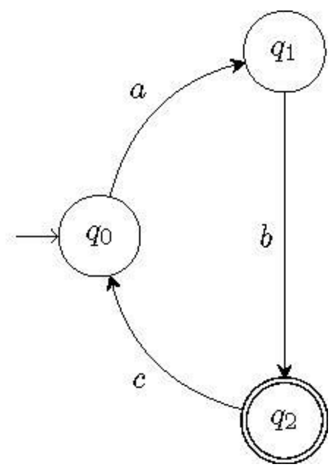
Automate accessible



Automate co-accessible



Automate élagué

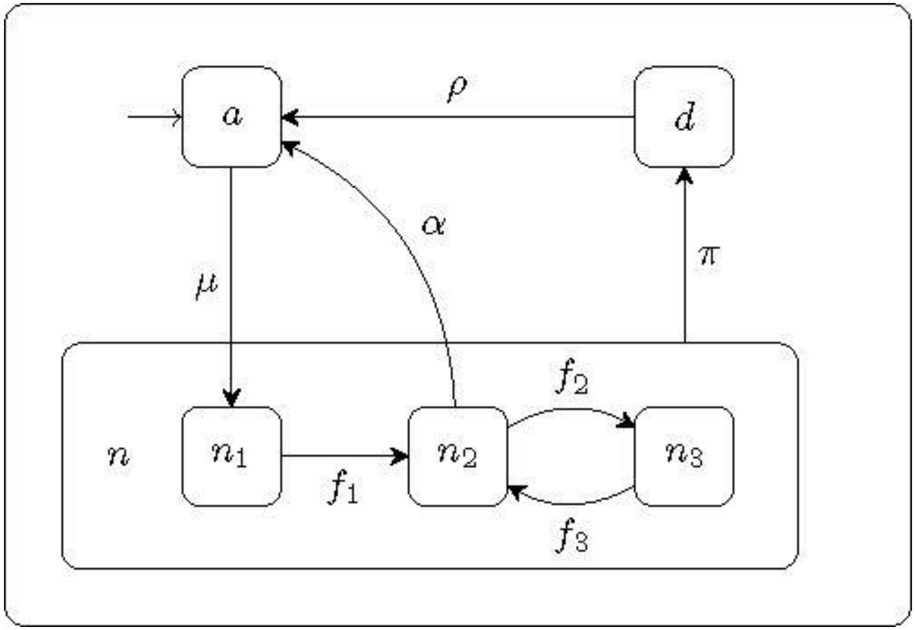


II.1. Composition des automates non temporisés

Poly 7.1.3

- **Automate hiérarchique**
  - Certains de ses états sont eux-mêmes des automates
- Exemple

Fonctionnement d'une machine



II.1. Composition des automates non temporisés

- **Produit synchrone d'automates**

$$A_1 \times A_2 = Ac(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \phi_{prod}, Q_{01} \times Q_{02}, Q_{f1} \times Q_{f2})$$

un évènement arrive exactement au même instant dans les 2, entrainant une transition simultanée

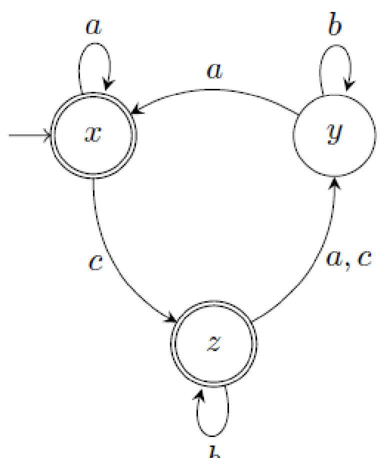
Poly 7.1.3

avec

$$\phi_{prod}((q_1,q_2),e) = \begin{cases} (\phi_1(q_1,e), \phi_2(q_2,e)) & \text{si les deux transitions sont définies} \\ \text{non définie sinon} \end{cases}$$

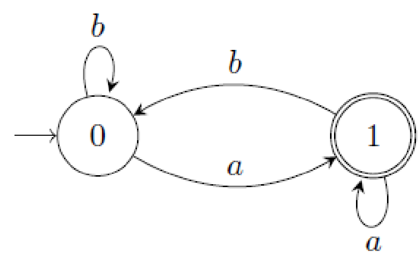
seulement les évènements appartenant à l'intersection de 2 ensembles

- Exemple



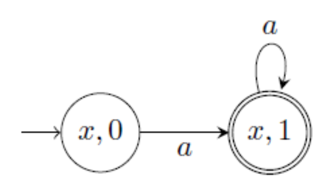
A_1

$\times$



A_2

$=$



$A_1 \times A_2$

II.1. Composition des automates non temporisés

Poly 7.1.3

- **Parallélisation d'automates**

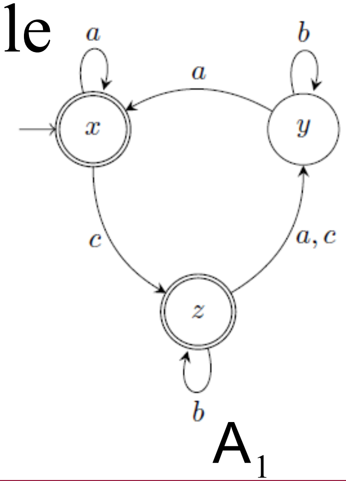
partiellement synchronisée : un évènement présent simultanément sur les deux automates doit se déclencher exactement au même instant

$$A_1 \parallel A_2 = Ac(Q_1 \times Q_2,\ \Sigma_1 \cup \Sigma_2,\ \phi_{par},\ Q_{01} \times Q_{02}, Q_{f1} \times Q_{f2})$$

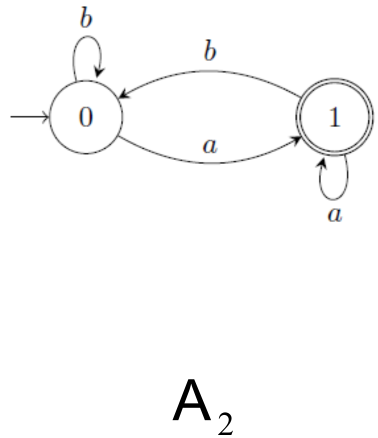
avec

$$\phi_{par}((q_1,q_2),e) = \begin{cases} (\phi_1(q_1,e), \phi_2(q_2,e)) & \text{si les deux transitions sont définies} \\ (\phi_1(q_1,e), q_2) & \text{si la transition n'est définie que pour } A_1 \\ (q_1, \phi_2(q_2,e)) & \text{si la transition n'est définie que pour } A_2 \\ \text{non définie sinon} & \end{cases}$$

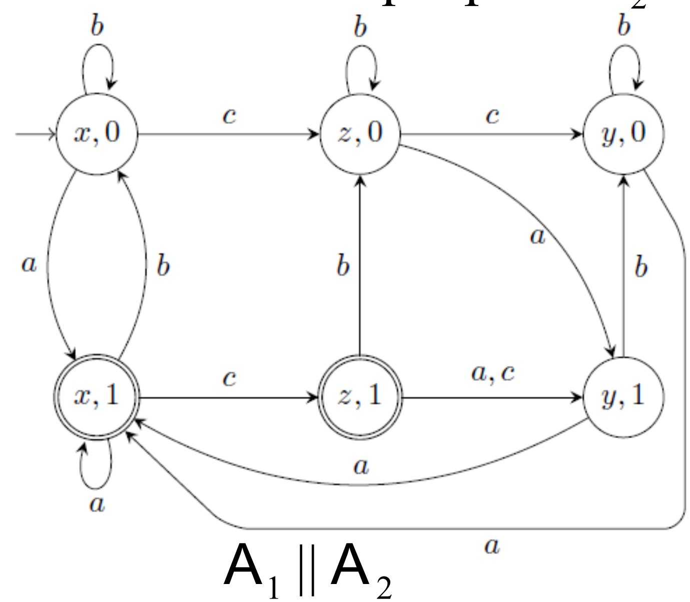
- Exemple



$\parallel$



$=$



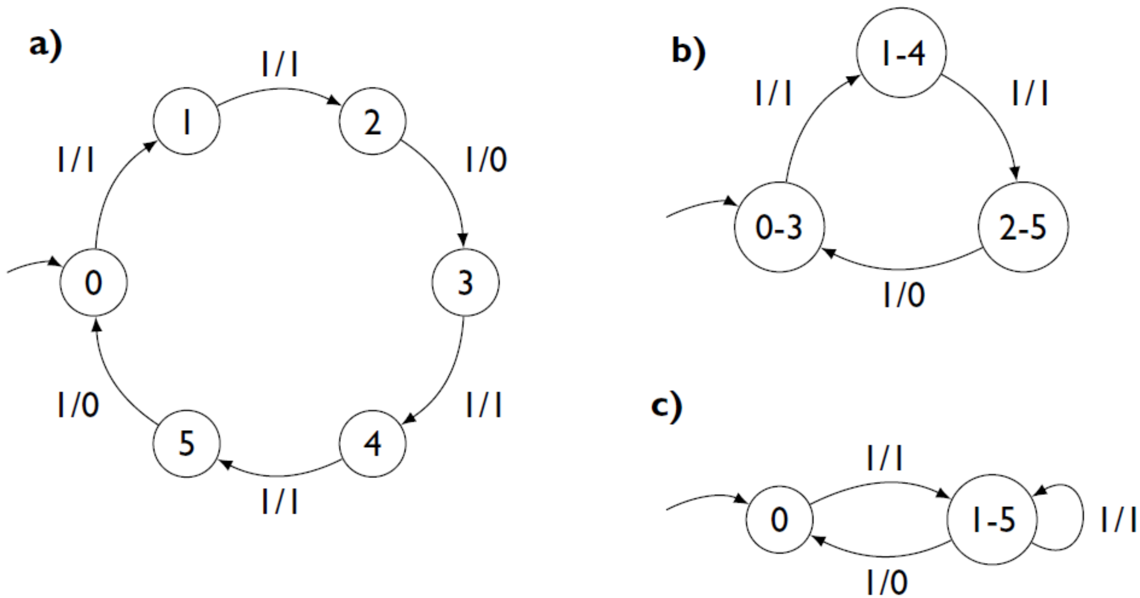
Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation par automates non temporisés
    - Définition, propriétés et composition des automates
    - Analyse d'un SED non temporisé via un automate
  - Modélisation par réseaux de Petri non temporisés
- Chapitre 2 : Modélisation des systèmes à évènements discrets temporisés

II.1. Analyse d’un SED non temporisé via un automate

- **Simulation et bisimulation d’automates**
- Exemple
  - Bisimulation : forme d’équivalence entre les automates
  - Simulation : forme d’abstraction

Poly 7.1.4



b) bisimule a)
i.e. a) simule b) et b) simule a)

c) simule a) et b)

II.1. Analyse d'un SED non temporisé via un automate

Poly 7.1.4

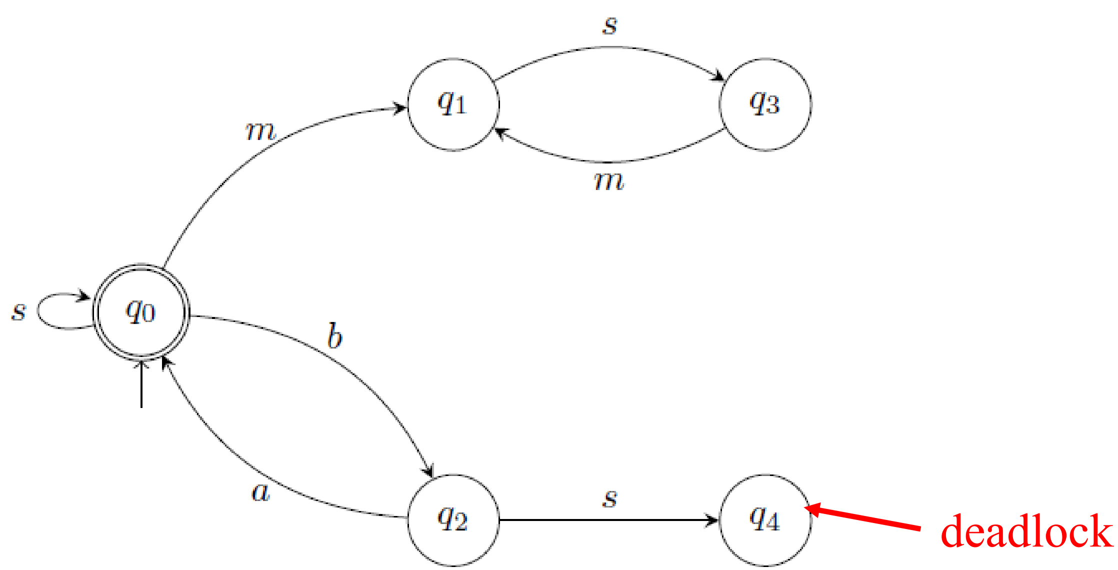
- **Simulation et bisimulation d'automates**
  - A_1 *simule* A_2 si toute suite d'évènements exécutés par A_2 peut également être exécutée par A_1
  - A_1 et A_2 sont **bisimilaires** si A_1 *simule* A_2 et A_2 *simule* A_1
  - $\mathsf{A}_1 = (Q_1,\ \Sigma,\ \phi_1,\ Q_{01}, Q_{f1})$ et $\mathsf{A}_2 = (Q_2,\ \Sigma,\ \phi_2,\ Q_{02}, Q_{f2})$ **bisimilaires**

  si $\exists \mathsf{R}$ relation binaire de $Q_1 \times Q_2$ avec $(q_1, q_2) \in \mathsf{R} \Leftrightarrow q_1 \mathsf{R}\ q_2$

- $Q_{01} \mathsf{R}\ Q_{02}$
- $\forall q_1 \in Q_1, \exists q_2 \in Q_2$ t.q. $q_1 \mathsf{R}\ q_2$; $\forall q_2 \in Q_2, \exists q_1 \in Q_1$ t.q. $q_1 \mathsf{R}\ q_2$
- si $q_1 \mathsf{R}\ q_2$ et $\sigma \in \Sigma$, si $q_3 = \phi_1(q_1, \sigma)$ est défini $\Rightarrow q_4 = \phi_2(q_2, \sigma)$ est défini et $q_3 \mathsf{R}\ q_4$
- si $q_1 \mathsf{R}\ q_2$ et $\sigma \in \Sigma$, si $q_4 = \phi_2(q_2, \sigma)$ est défini $\Rightarrow q_3 = \phi_1(q_1, \sigma)$ est défini et $q_3 \mathsf{R}\ q_4$
- pour les états marqués, si $q_1 \mathsf{R}\ q_2$, alors $\left(q_1 \in Q_{f1} \Leftrightarrow q_2 \in Q_{f2}\right)$

Poly 7.1.4

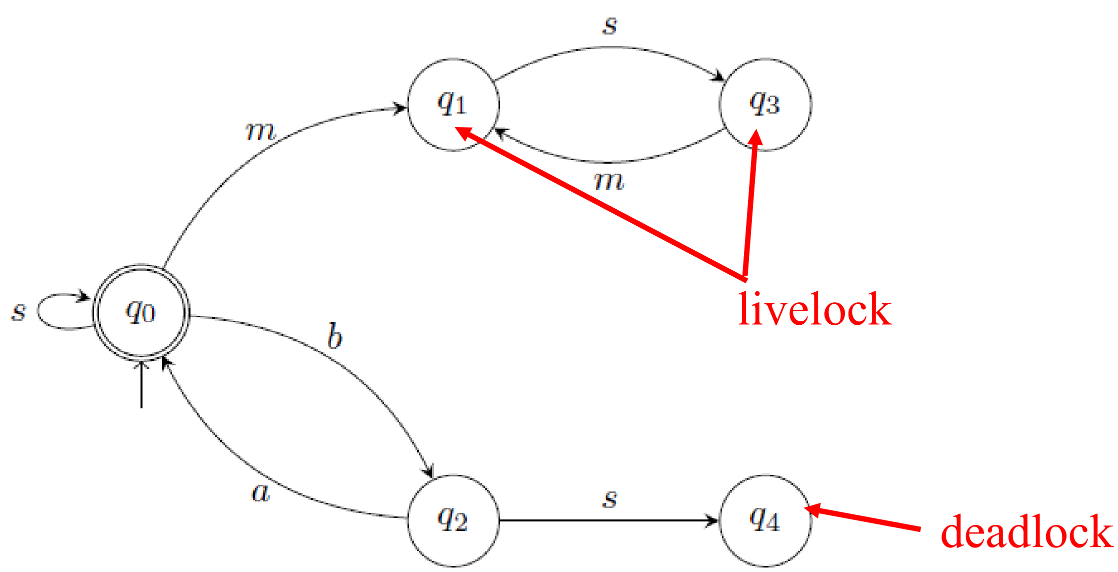
- **Deadlock et livelock**
  - *Deadlock* ou *blocage fatal* : situation où il n'y a plus d'évolution possible alors que l'automate est arrivé dans un état non marqué
  - *Livelock* ou *blocage vivant* : situation où un sous-ensemble d'états non marqués devient le seul accessible



II.1. Analyse d'un SED non temporisé via un automate

Poly 7.1.4

- **Deadlock et livelock**
  - *Deadlock* ou *blocage fatal* : situation où il n'y a plus d'évolution possible alors que l'automate est arrivé dans un état non marqué
  - *Livelock* ou *blocage vivant* : situation où un sous-ensemble d'états non marqués devient le seul accessible



Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation par automates non temporisés
  - Modélisation par réseaux de Petri non temporisés
    - Définitions et propriétés des réseaux de Petri
    - Analyse des réseaux de Petri par réduction et analyse linéaire
    - Réseaux de Petri non autonomes synchronisés
- Chapitre 2 : Modélisation des systèmes à évènements discrets temporisés
- Chapitre 3 : Automates stochastiques

II.1. Modélisation par réseaux de Petri non temporisés

Poly 7.2

- Motivation de la modélisation par réseaux de Petri
  - Alternative aux automates à états finis pour la représentation des systèmes à évènements discrets
    - Bien adaptés à la modélisation des processus ayant des activités parallèles
    - Développés pour besoin d'analyse et de validation des programmes conçus pour les machines multiprocesseurs
    - Adaptés au comportement des systèmes industriels, des réseaux de communication

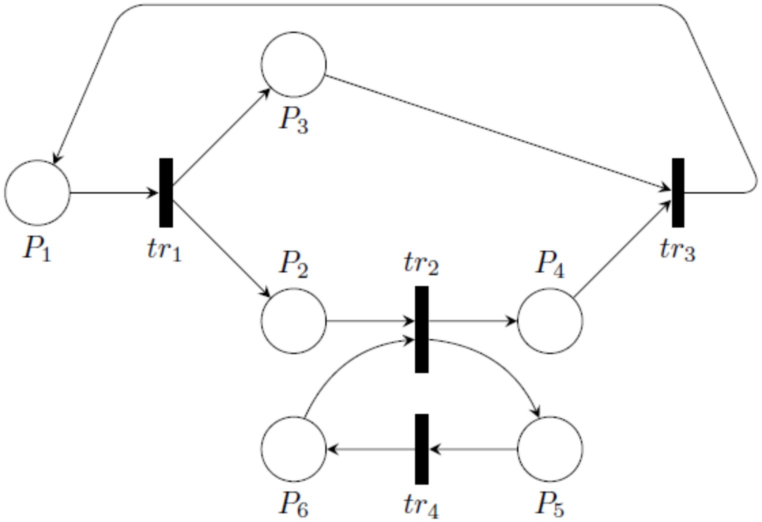
Tout automate peut se représenter sous la forme d'un réseau de Petri.

La réciproque est fausse !

II.1. Définitions des réseaux de Petri non temporisés

Poly 7.2.1

- **Réseau de Petri autonome (non marqué)**
  - graphe $R = (P; T; Arc)$
    - P : *places* symbolisées par des cercles
    - T : *transitions* symbolisées par des traits
    - Arc : *arcs orientés*
      reliant 2 noueux de nature différente



$$P = \{P_1, \ldots, P_6\}$$

$$T = \{tr_1, \ldots, tr_4\}$$

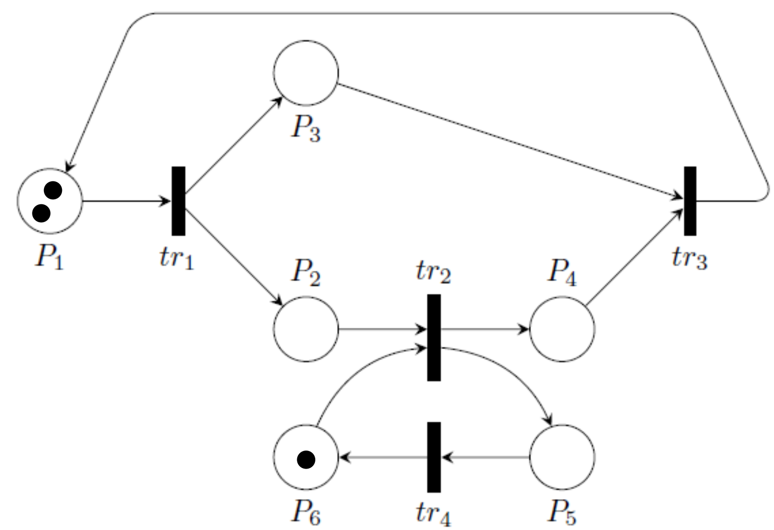
II.1. Définitions des réseaux de Petri non temporisés

Poly 7.2.1

- **Réseau de Petri autonome (non marqué)**
  - graphe $R = (P; T; Arc)$
    - P : places symbolisées par des cercles
    - T : transitions symbolisées par des traits
    - Arc : arcs orientés
- **Modélisation de la dynamique**

des systèmes via *marquages*

- **Marquage** :
  - Application $\mathrm{m} : P \to \mathbb{N}$
    qui associe un nombre naturel
    à toute place du réseau
  - Symbolisé sur le graphe par des jetons



$$\mathrm{m}(P_1) = 2,\ \mathrm{m}(P_6) = 1,$$
$$\mathrm{m}(P_i) = 0,\ i \in \{2, 3, 4, 5\}$$

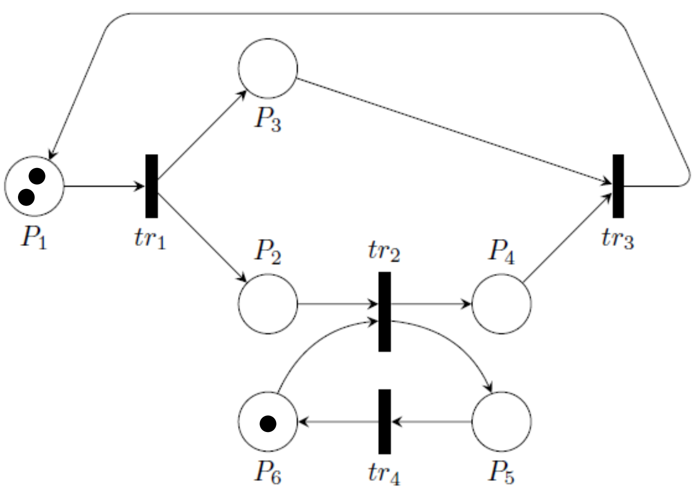
II.1. Définitions des réseaux de Petri non temporisés

Poly 7.2.1

- **Transition validée**
  - si toutes ses places d'entrée sont marquées (au moins un jeton)
- **Transition franchie**
  - si elle est validée, en retirant un jeton de chaque place d'entrée et en ajoutant un jeton dans chaque place de sortie
- **Réseau de Petri discret autonome marqué** $R = (P,\ T,\ Pre,\ Post, m_0)$
  - $P = \{P_1, \ldots, P_n\}$: ensemble fini, non vide de places
  - $T = \{tr_1, \ldots, tr_m\}$: ensemble fini, non vide de transitions, $P \cap T = \varnothing$
  - $Pre : P \times T \to \mathbb{N}$: application d'incidence avant, où $Pre(P_i, tr_j)$ contient la valeur entière associée à l'arc allant de P_i à tr_j
  - $Post : P \times T \to \mathbb{N}$: application d'incidence arrière, où $Post(P_i, tr_j)$ contient la valeur entière associée à l'arc allant de tr_j à P_i
  - $m_0 : P \to \mathbb{N}$: marquage initial du réseau

II.1. Définitions des réseaux de Petri non temporisés

- **Exemple**
  - Trouver l'application d'incidence avant, l'application d'incidence arrière et le marquage initial



$$Pre : \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad Post : \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad m_0 : \begin{pmatrix} 2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

II.1. Définitions des réseaux de Petri non temporisés

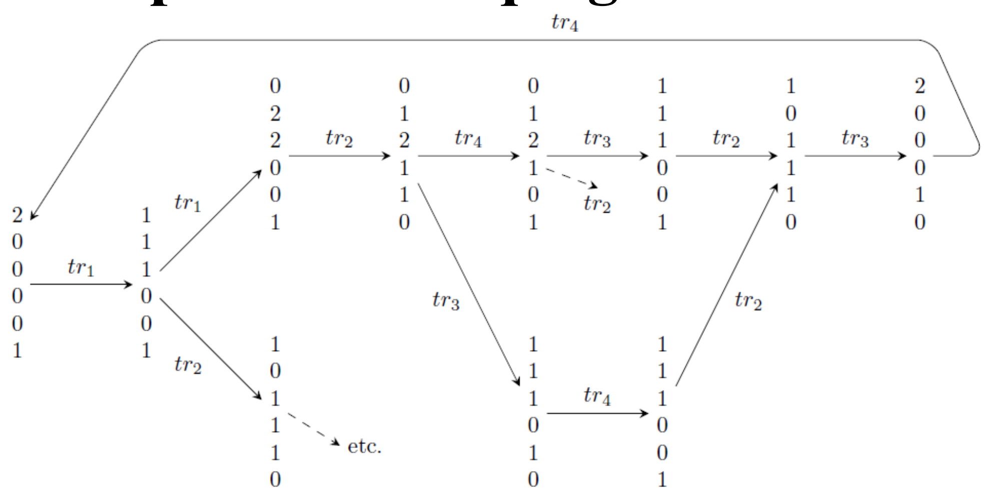
Poly 7.2.1

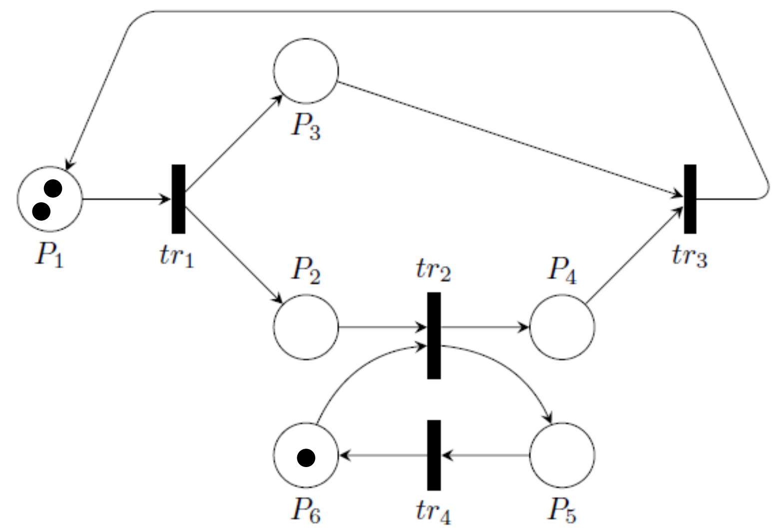
- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

1. déterminer les transitions franchissables
2. choisir une séquence de franchissement
3. franchir la première transition de cette séquence

- **Graphe des marquages**





II.1. Définitions des réseaux de Petri non temporisés

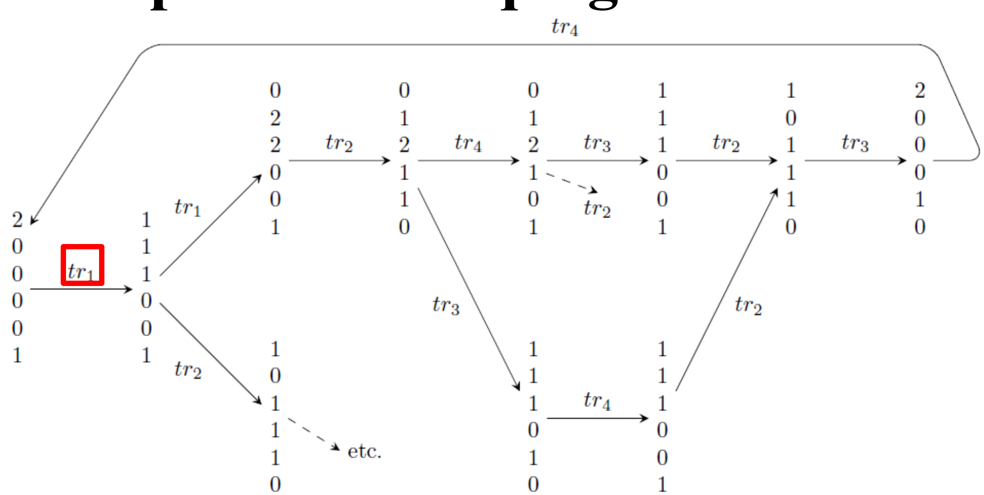
Poly 7.2.1

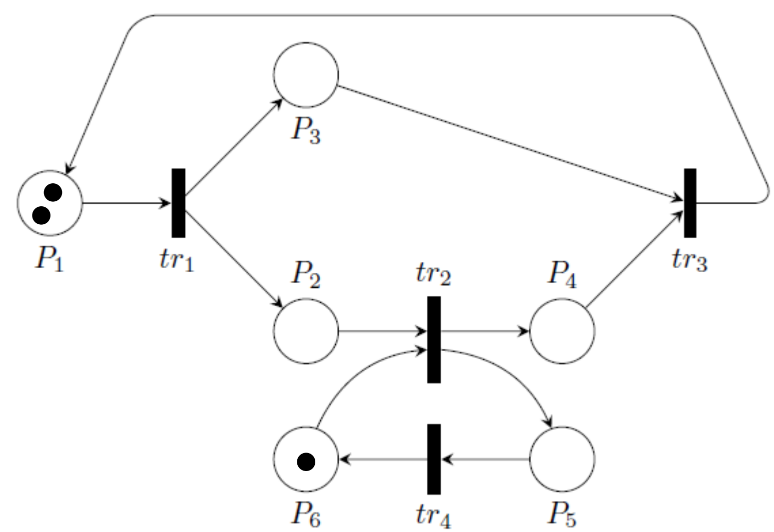
- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

1. déterminer les transitions franchissables
2. choisir une séquence de franchissement
3. franchir la première transition de cette séquence

- **Graphe des marquages**





II.1. Définitions des réseaux de Petri non temporisés

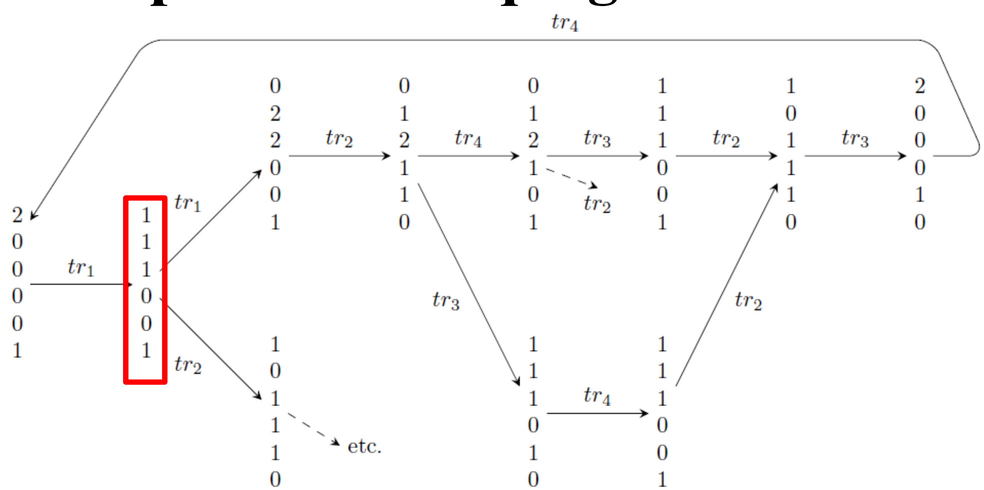
Poly 7.2.1

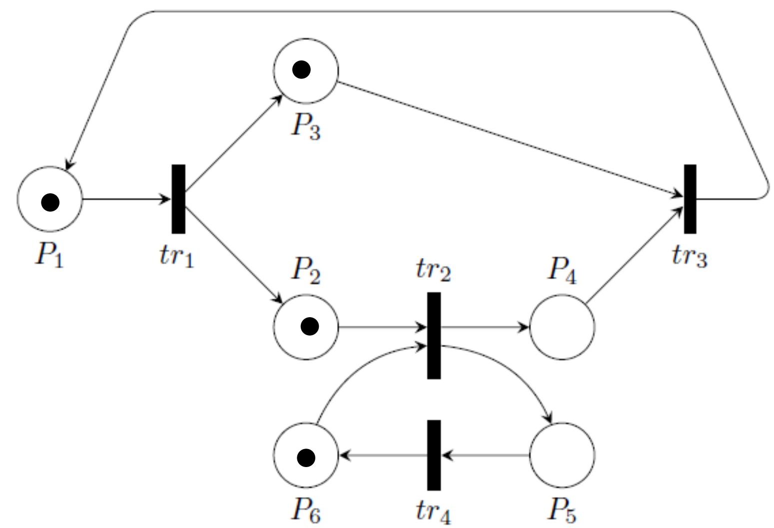
- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

1. déterminer les transitions franchissables
2. choisir une séquence de franchissement
3. franchir la première transition de cette séquence

- **Graphe des marquages**





II.1. Définitions des réseaux de Petri non temporisés

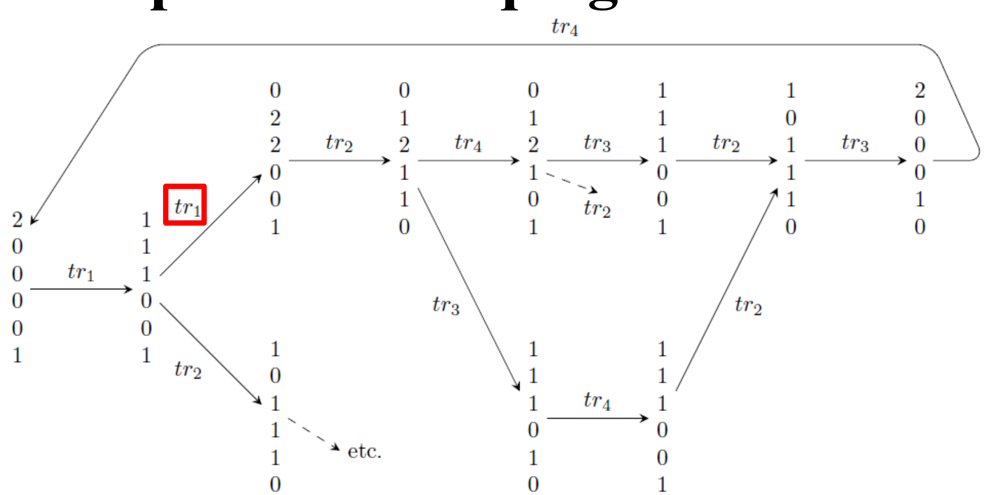
Poly 7.2.1

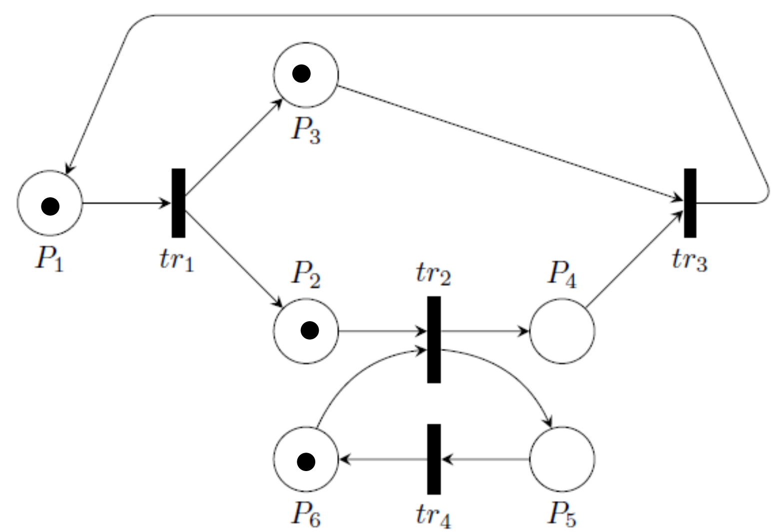
- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

1. déterminer les transitions franchissables
2. choisir une séquence de franchissement
3. franchir la première transition de cette séquence

- **Graphe des marquages**





II.1. Définitions des réseaux de Petri non temporisés

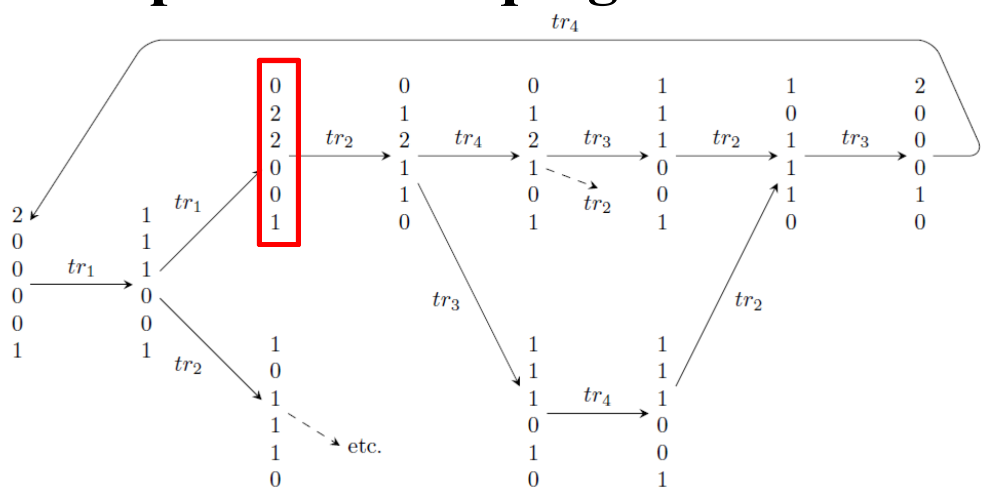
Poly 7.2.1

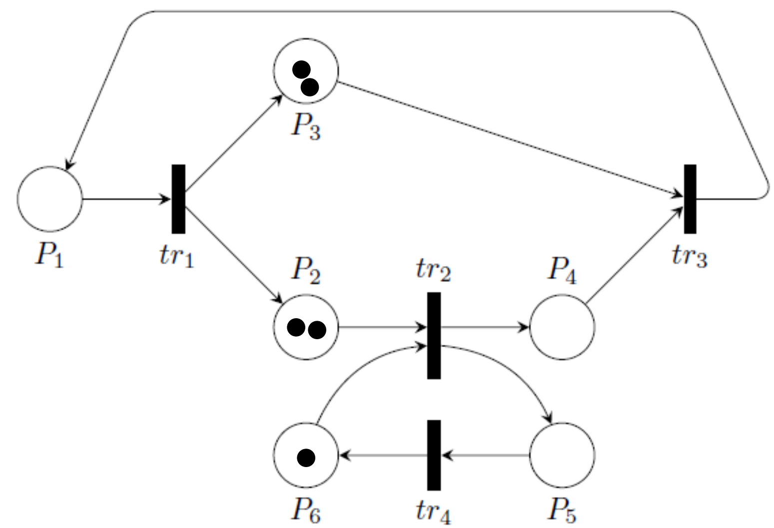
- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

  1. déterminer les transitions franchissables
  2. choisir une séquence de franchissement
  3. franchir la première transition de cette séquence

- **Graphe des marquages**





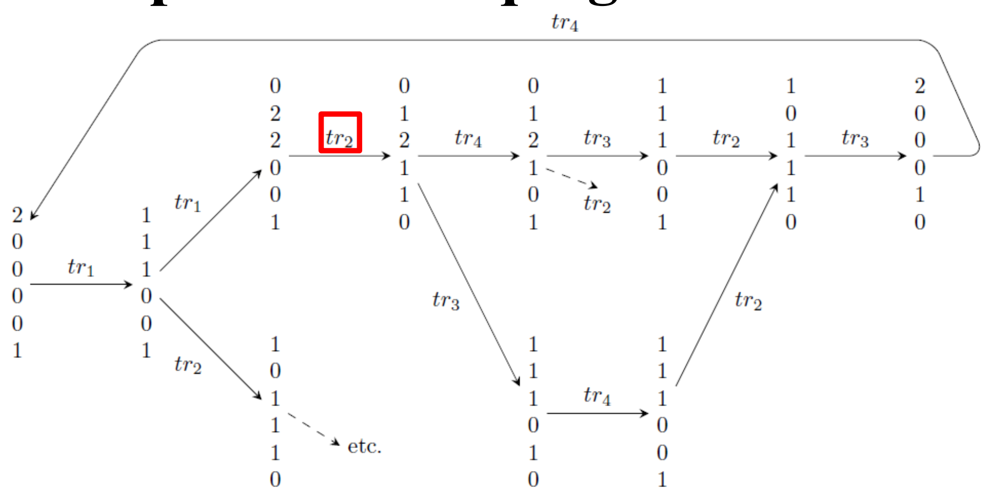
II.1. Définitions des réseaux de Petri non temporisés

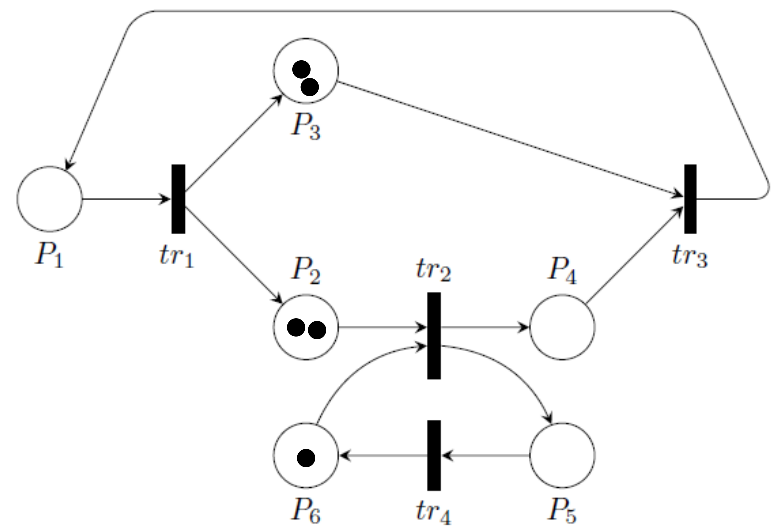
Poly 7.2.1

- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

  1. déterminer les transitions franchissables
  2. choisir une séquence de franchissement
  3. franchir la première transition de cette séquence
- **Graphe des marquages**



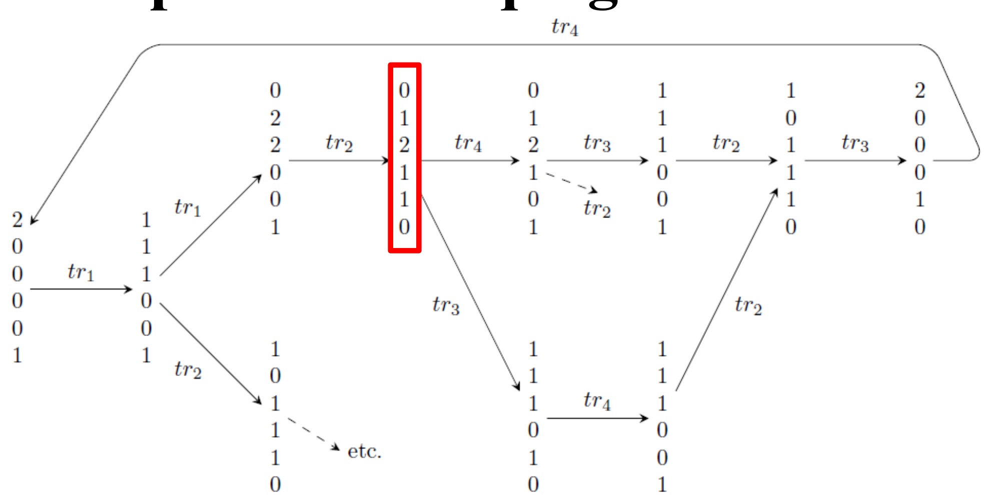


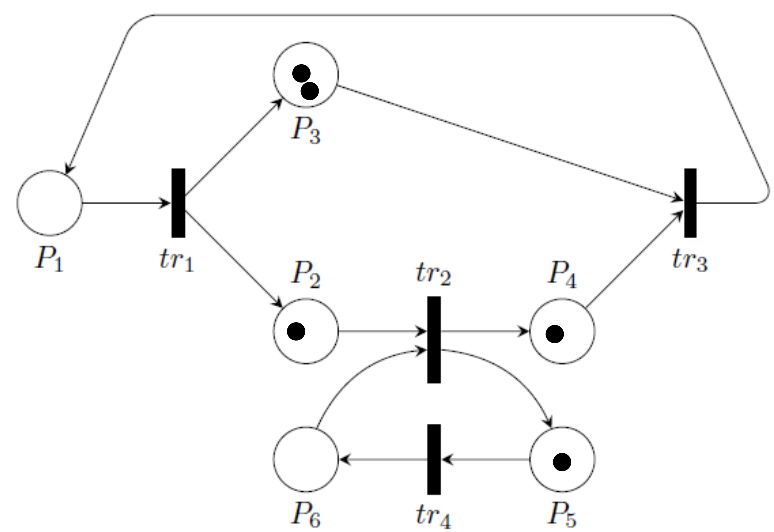
Poly 7.2.1

- **Séquence de franchissements**
  - Plusieurs transitions franchissables, franchies successivement
- **Algorithme d'interprétation d'un réseau de Petri**

répéter

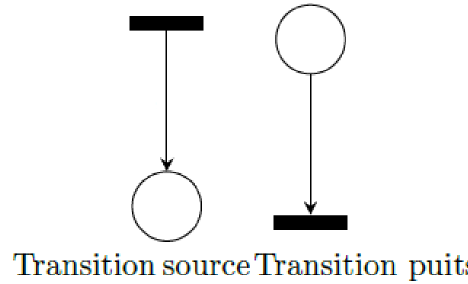
  1. déterminer les transitions franchissables
  2. choisir une séquence de franchissement
  3. franchir la première transition de cette séquence
- **Graphe des marquages**



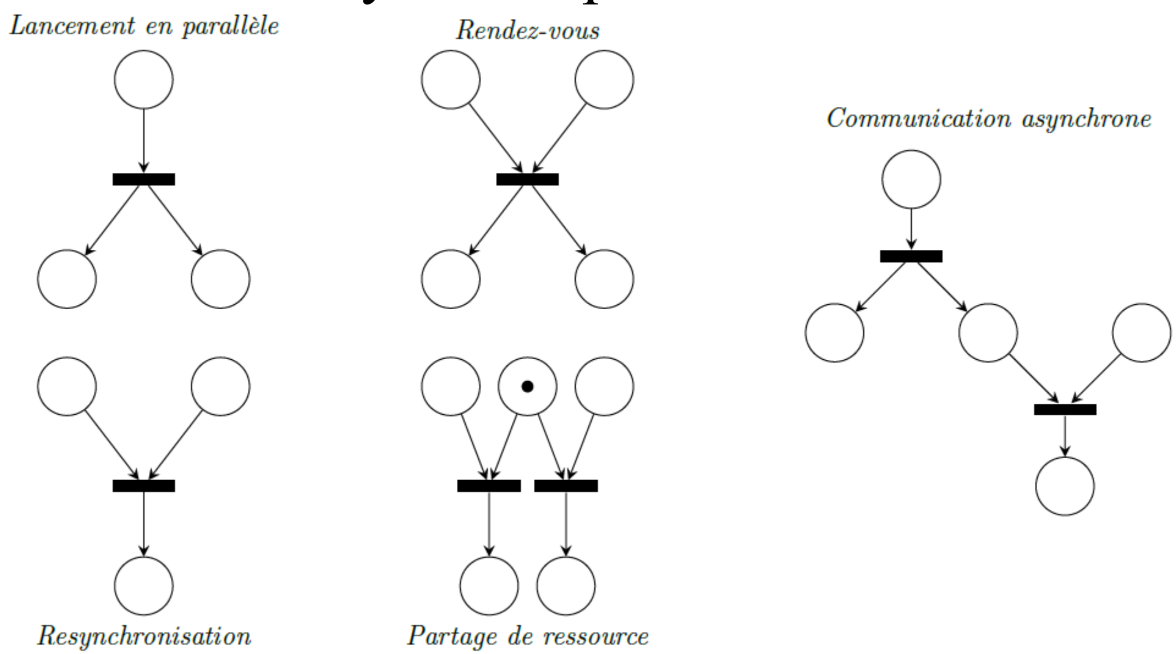


II.1. Définitions des réseaux de Petri non temporisés

- **Transition source**
  - pas de place d'entrée et toujours franchissable
- **Transition puits**
  - pas de place de sortie et font disparaître des marques



- **Exemple :** évolution de systèmes parallèles



Poly 7.2.1

II.1. Propriétés des réseaux de Petri non temporisés

- **Source d'erreur dans la modélisation par réseau de Petri**
  - erreurs de connaissance du système modélisé
  - erreurs de modélisation, car mauvaise utilisation du réseau de Petri

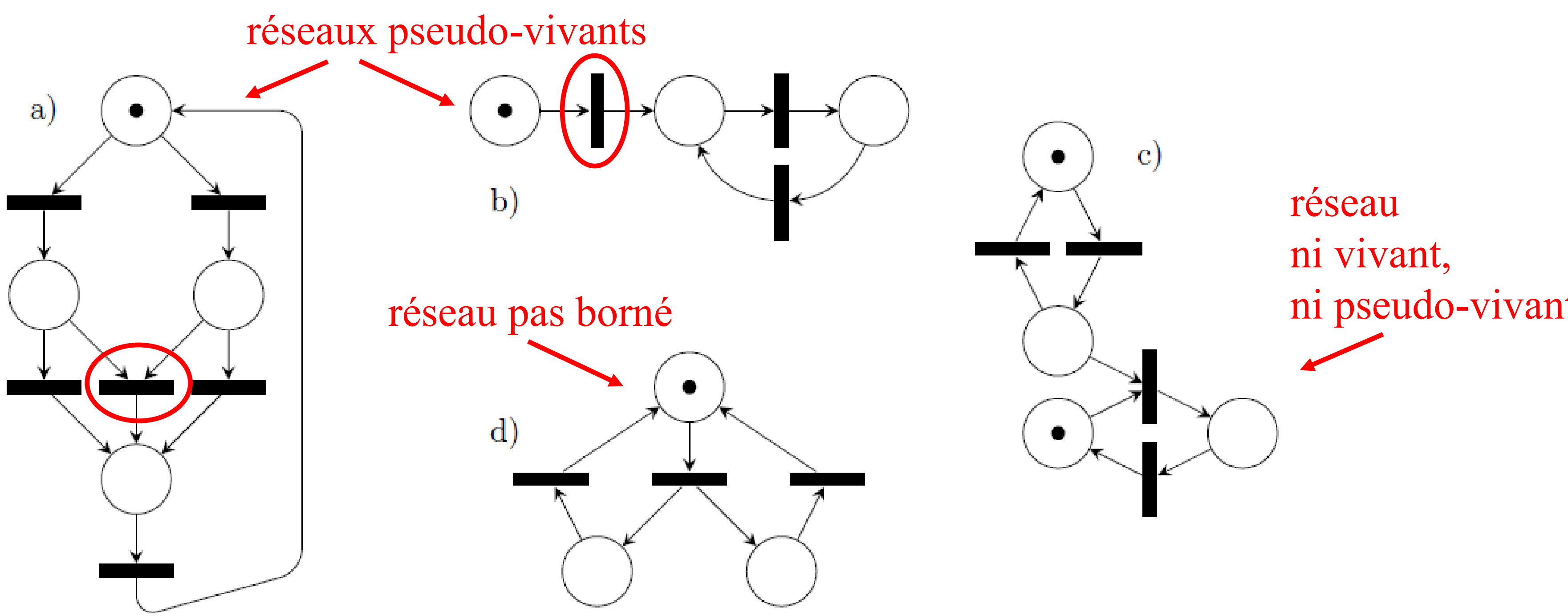
Poly 7.2.2

➡ Trouver des propriétés identifiables du système modélisé

  - **Réseau vivant**
    - à partir du marquage initial et de tout marquage consécutif, toute transition peut être incluse dans une séquence de franchissement
  - **Réseau pseudo-vivant**
    - pour tout marquage accessible, au moins une transition peut être franchie
  - **Réseau borné**
    - pour tout marquage accessible, nombre de marques dans chaque place < k
  - **Réseau sauf**
    - réseau vivant dont la borne est égale à 1

Poly 7.2.2

■ Exemples

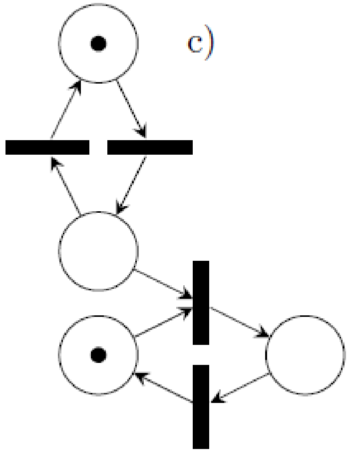


II.1. Propriétés des réseaux de Petri non temporisés

Poly 7.2.2

- **Exemples**

Réseau ni vivant, ni pseudo-vivant

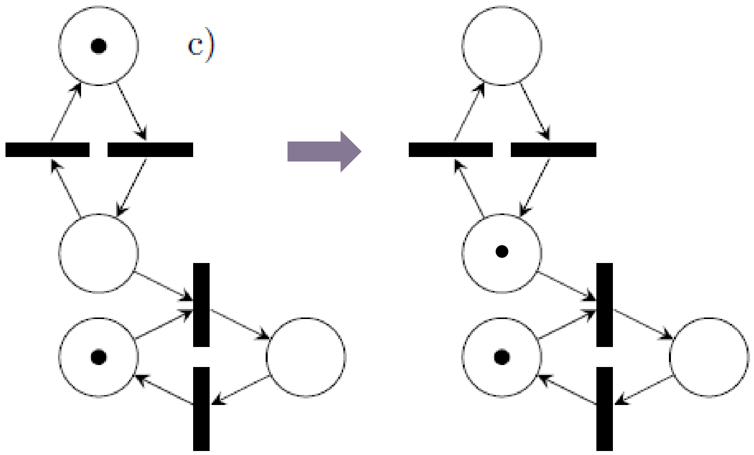


c)

Poly 7.2.2

- **Exemples**

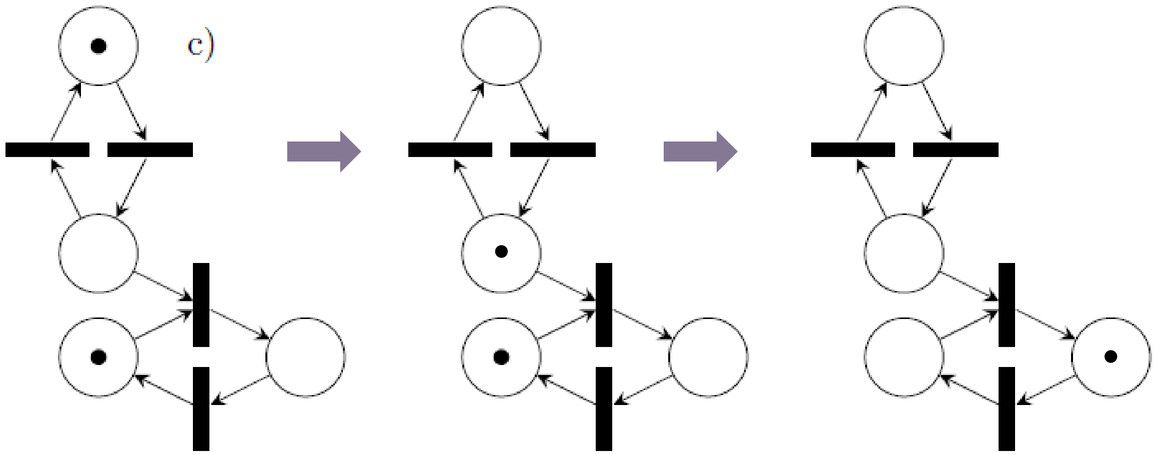
Réseau ni vivant, ni pseudo-vivant



Poly 7.2.2

- **Exemples**

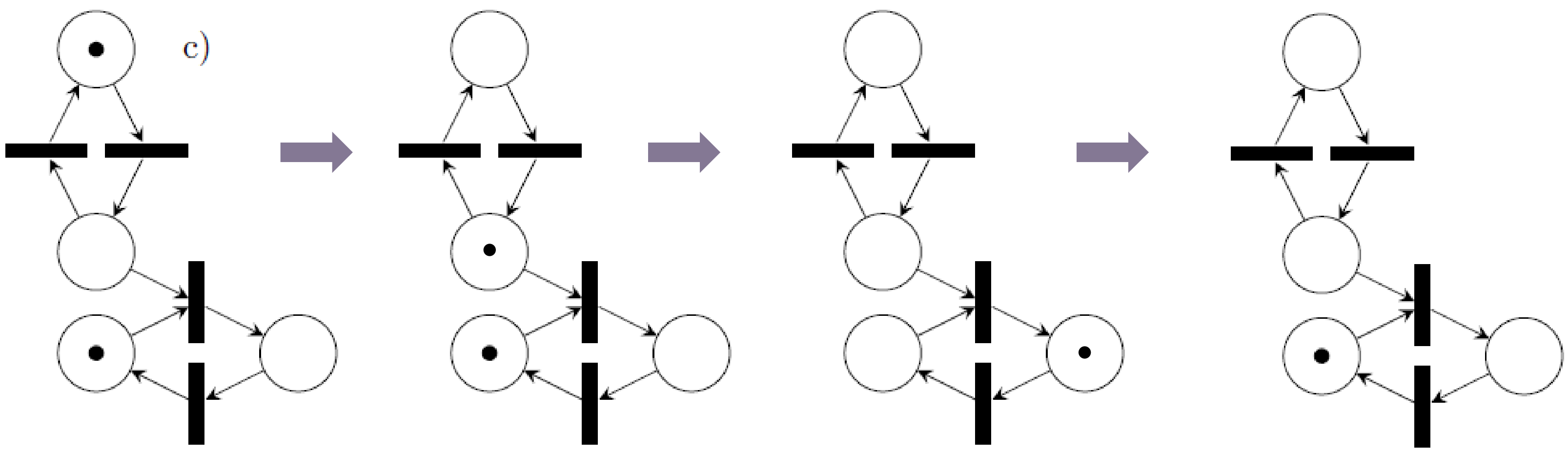
Réseau ni vivant, ni pseudo-vivant



- **Exemples**

Poly 7.2.2

Réseau ni vivant, ni pseudo-vivant

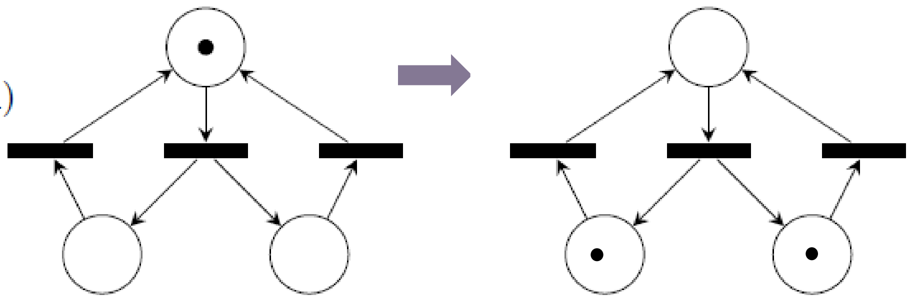


Poly 7.2.2

- **Exemples**

Réseau pas borné

d)

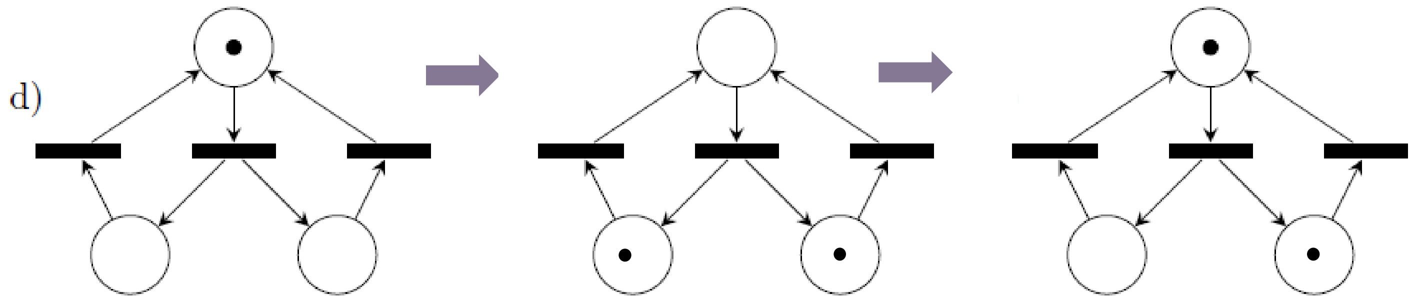


II.1. Propriétés des réseaux de Petri non temporisés

Poly 7.2.2

- **Exemples**

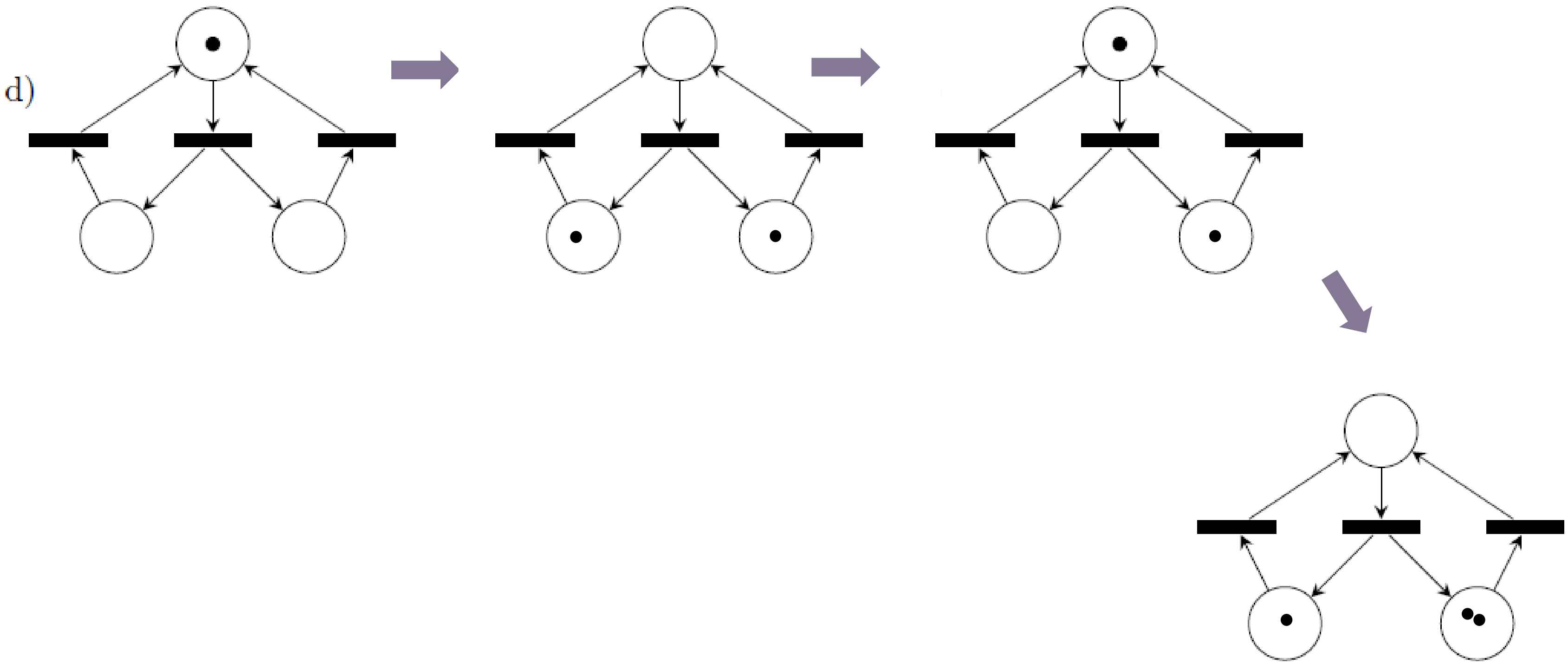
Réseau pas borné



Poly 7.2.2

- **Exemples**

Réseau pas borné

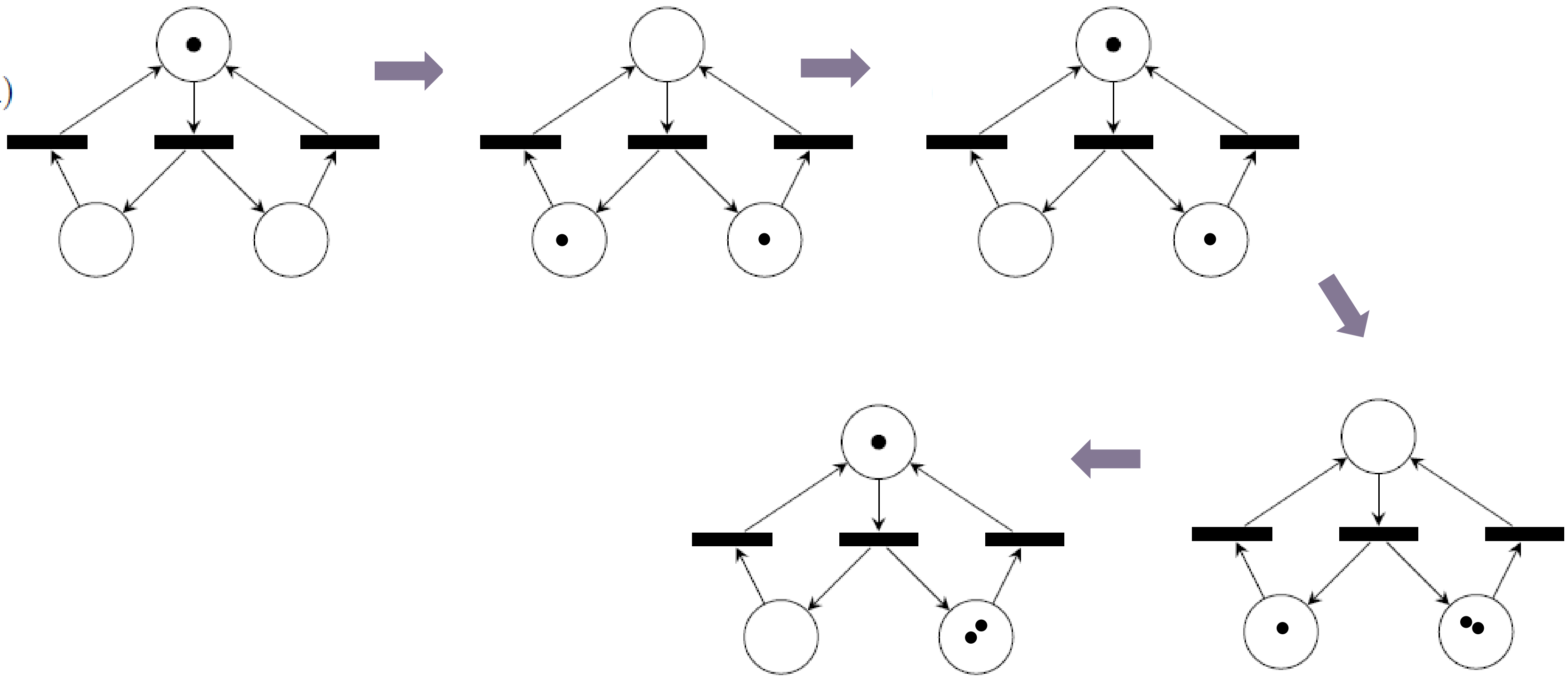


Poly 7.2.2

- **Exemples**

Réseau pas borné

d)

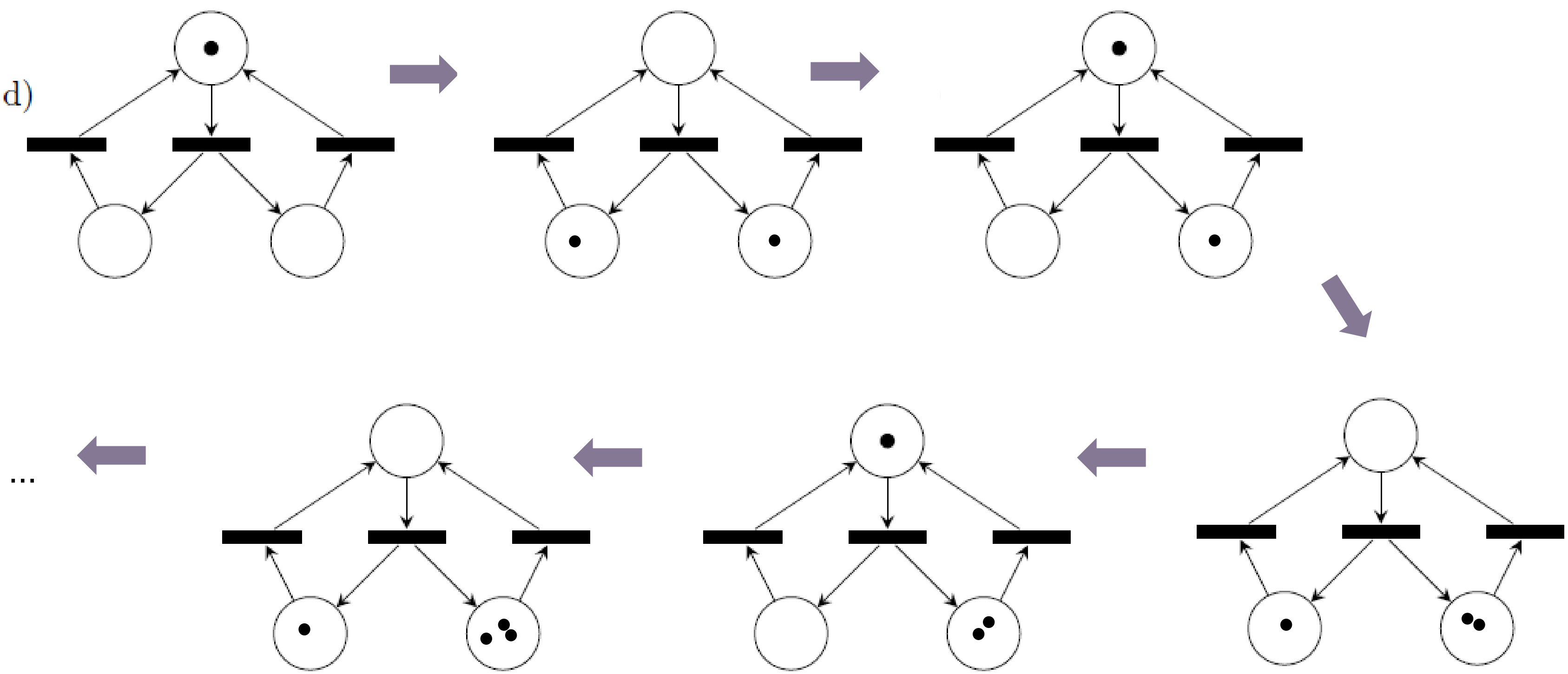


Poly 7.2.2

- **Exemples**

Réseau pas borné

d)



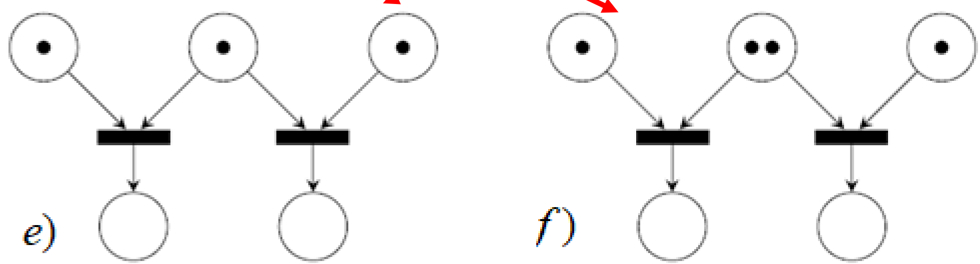
...

II.1. Propriétés des réseaux de Petri non temporisés

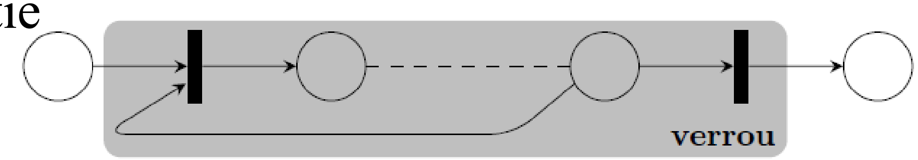
Poly 7.2.2

- **Réseau avec conflit**
  - plusieurs transitions franchissables et le franchissement d'une d'entre elles invalide le franchissement d'une autre

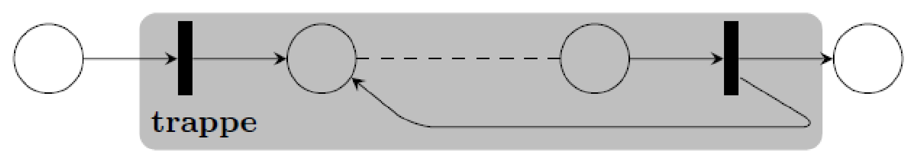
réseaux avec/sans conflits



- **Réseau persistant**
  - pas de situation de conflit
- **Verrou dans un réseau de Petri**
  - ensemble de places t.q. l'ensemble de ses transitions d'entrée est inclus dans l'ensemble de ses transitions de sortie



- **Trappe dans un réseau de Petri**
  - ensemble de places t.q. l'ensemble de ses transitions de sortie est inclus dans l'ensemble de ses transitions d'entrée



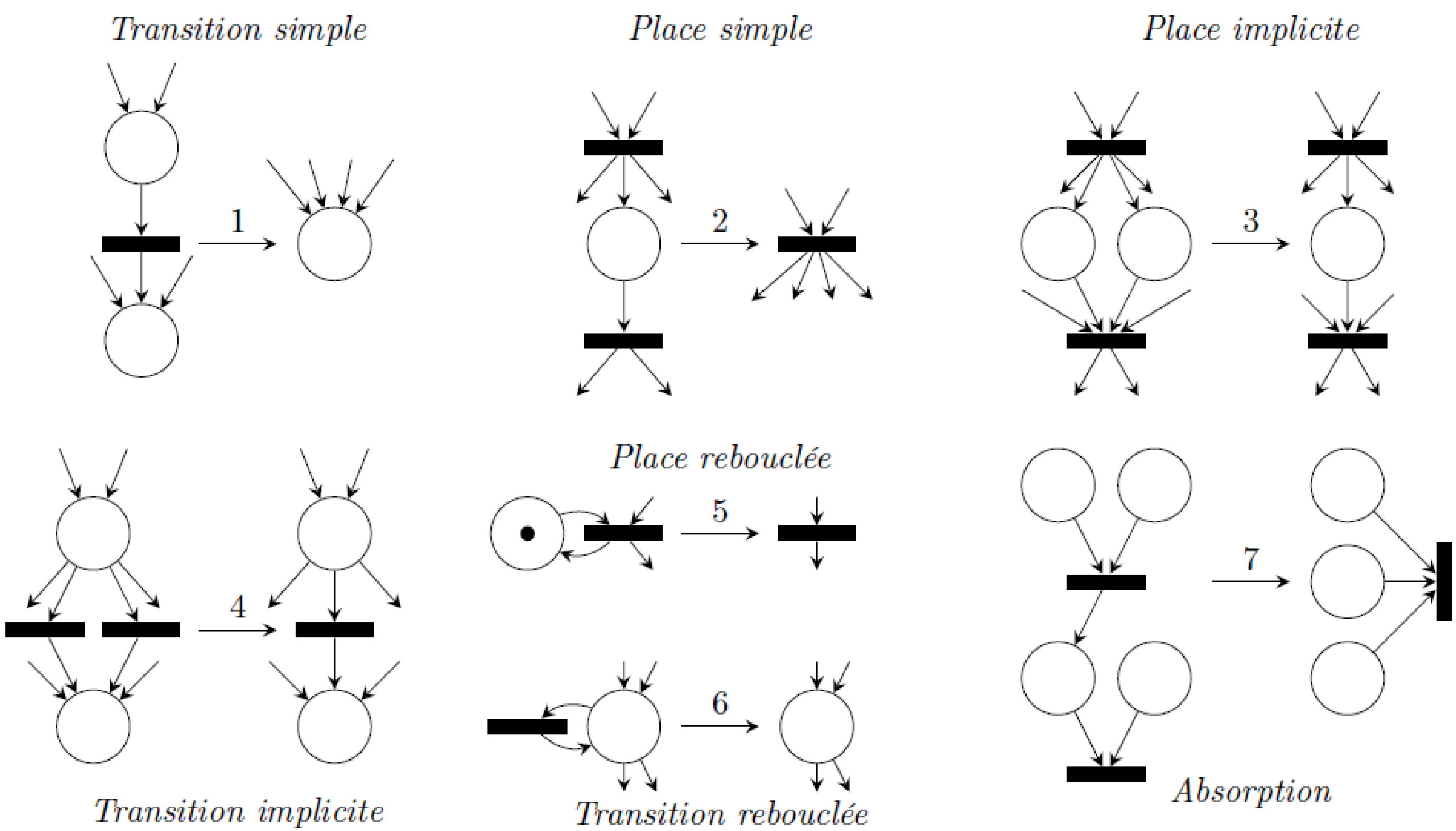
Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation par automates non temporisés
  - Modélisation par réseaux de Petri non temporisés
    - Définitions et propriétés des réseaux de Petri
    - Analyse des réseaux de Petri par réduction et analyse linéaire
    - Réseaux de Petri non autonomes synchronisés
- Chapitre 2 : Modélisation des systèmes à évènements discrets temporisés
- Chapitre 3 : Automates stochastiques

II.1. Analyse des réseaux de Petri par réduction

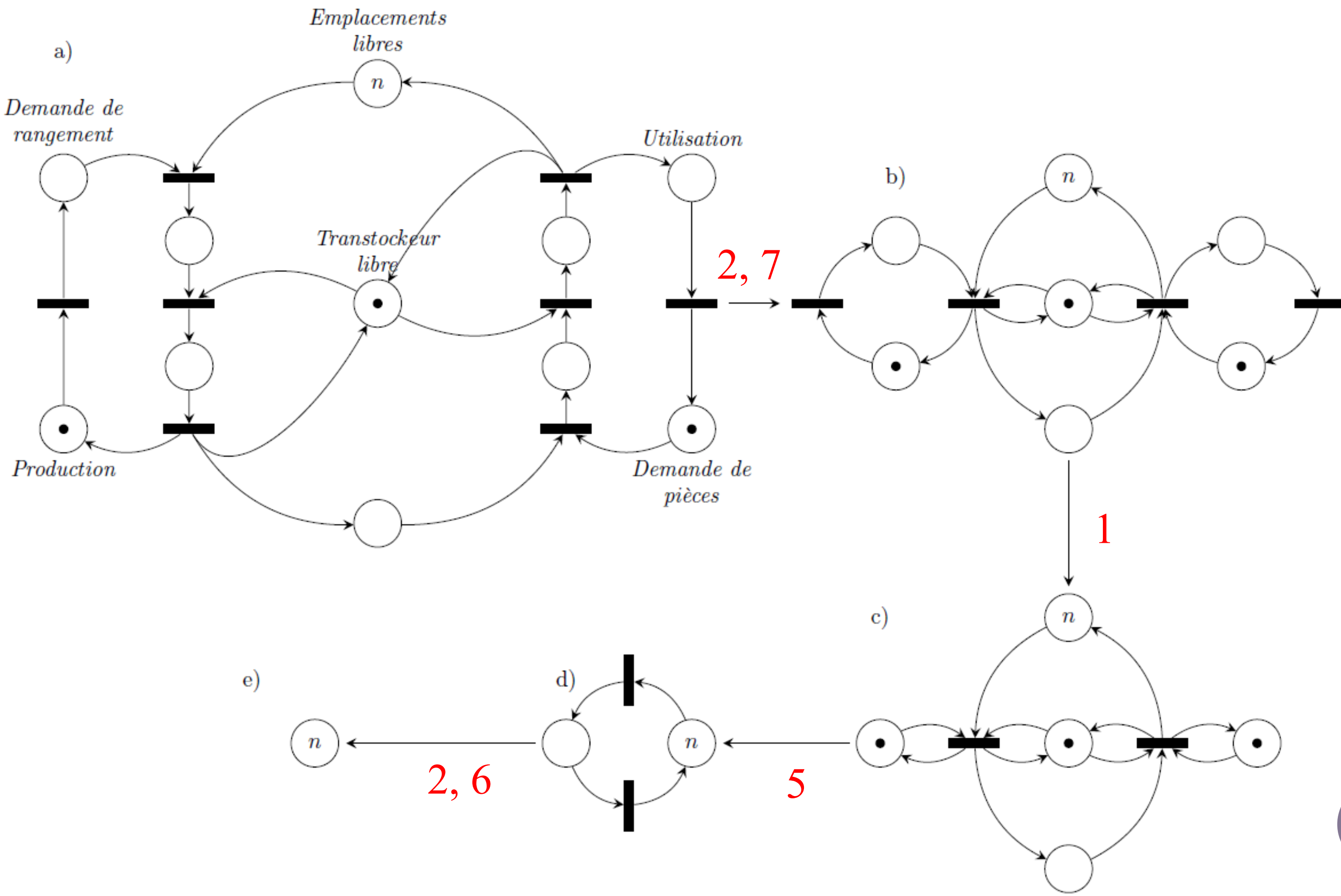
Poly 7.2.3

- Réduction du modèle par des transformations conservant les propriétés "vivants et bornés" des réseaux de Petri



II.1. Analyse des réseaux de Petri par réduction

- Exemple : modélisation de l'accès concurrent à un magasin de pièces



Poly 7.2.3

II.1. Analyse linéaire des réseaux de Petri

Poly 7.2.4

- **Matrice d'incidence** (et principes de l'analyse linéaire)

$$W = \left[w_{ij} \right], \text{ avec } w_{ij} = Post(P_i, tr_j) - Pre(P_i, tr_j), \; i \in \{1, \ldots, n\}, \; j \in \{1, \ldots, m\}$$

ou $Pre(P_i, tr_j) = \begin{cases} 1, & \text{si } P_i \text{ est place d'entrée pour } tr_j \\ 0, & \text{sinon} \end{cases}$ $\quad Post(P_i, tr_j) = \begin{cases} 1, & \text{si } P_i \text{ est place de sortie pour } tr_j \\ 0, & \text{sinon} \end{cases}$

- Validation d'une transition testée par $\boxed{\mathrm{m}(P) \geq -W(P, tr_i)}$

- Franchissement d'une transition tr_i validée :

$$\boxed{\mathrm{m}_2(P) = \mathrm{m}_1(P) + W(P, tr_i)}$$

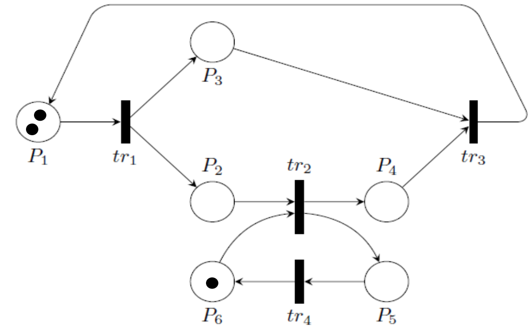
marquage résultant

marquage de départ

colonne correspondant à tr_i

II.1. Analyse linéaire des réseaux de Petri

- **Exemple**
  - Ecrire la matrice d'incidence
  - Choisir une séquence de transition et évaluer le marquage des places

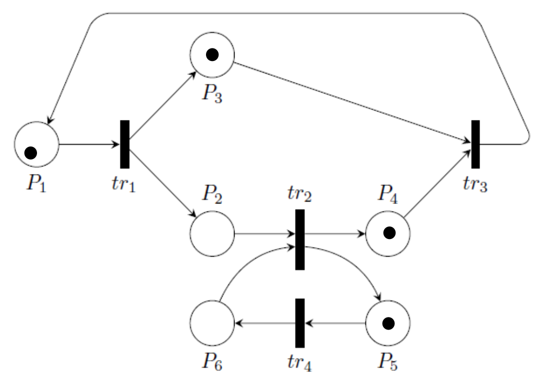


$$Post:\begin{pmatrix}0&0&1&0\\1&0&0&0\\1&0&0&0\\0&1&0&0\\0&1&0&0\\0&0&0&1\end{pmatrix}\quad Pre:\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&0\\0&0&0&1\\0&1&0&0\end{pmatrix}\Rightarrow W=\begin{pmatrix}-1&0&1&0\\1&-1&0&0\\1&0&-1&0\\0&1&-1&0\\0&1&0&-1\\0&-1&0&1\end{pmatrix}$$

$$tr_1 \to tr_2 \to \ldots$$

Toutes les transitions doivent être validées !

$$m_2 = m_0 + W(:,1) + W(:,2) = \begin{pmatrix}2\\0\\0\\0\\0\\1\end{pmatrix} + \begin{pmatrix}-1\\1\\1\\0\\0\\0\end{pmatrix} + \begin{pmatrix}0\\-1\\0\\1\\1\\-1\end{pmatrix} = \begin{pmatrix}1\\0\\1\\1\\1\\0\end{pmatrix}$$



Plan de la Partie II

- Introduction
- Chapitre 1 : Modélisation des systèmes à évènements discrets non temporisés
  - Modélisation par automates non temporisés
  - Modélisation par réseaux de Petri non temporisés
    - Définitions et propriétés des réseaux de Petri
    - Analyse des réseaux de Petri par réduction et analyse linéaire
    - Réseaux de Petri non autonomes synchronisés
- Chapitre 2 : Modélisation des systèmes à évènements discrets temporisés
- Chapitre 3 : Automates stochastiques

II.1. Réseaux de Petri non autonomes synchronisés

Poly 7.2.5

- **Réseau de Petri synchronisé**
  - réseau de Petri non autonome muni d'une application d'un ensemble d'évènements sur l'ensemble de ses transitions
- **Evénement toujours vrai** : e
- **Transition** tr_i **réceptive à l'évènement** e_i
  - si validée (toutes ses places d'entrée sont marquées)

Sur une occurrence de l'évènement e_i*, la transition devient franchissable*

- **Algorithme d'interprétation d'un réseau synchronisé**

répéter

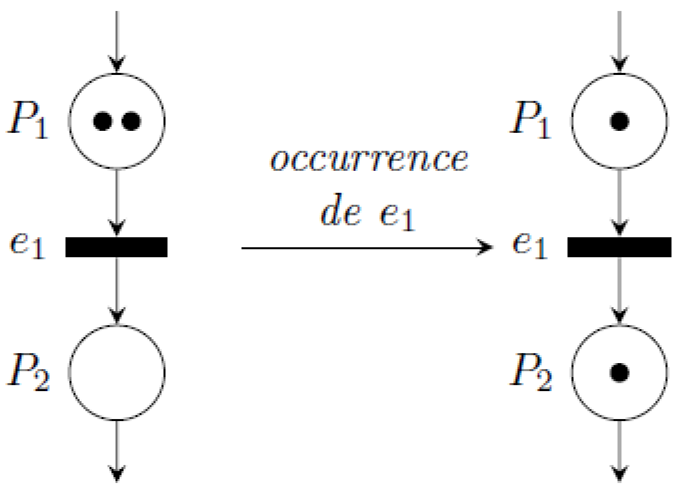
1. déterminer les évènements vrais
2. déterminer les transitions franchissables
3. effectuer une séquence de franchissement avec ces transitions

II.1. Réseaux de Petri non autonomes synchronisés

- **Exemple** : franchissement de transition d'un réseau de Petri synchronisé

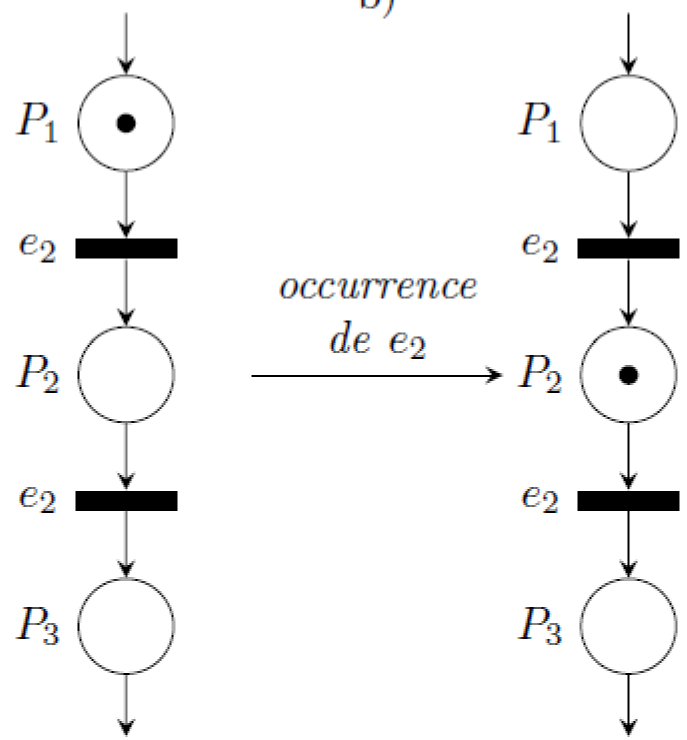
Poly 7.2.5

a)



Si événement vrai et marque en P_1, alors tr_1

b)



On ne peut pas franchir 2 transitions en même temps !