Méthodes Symboliques pour l'Exploration des Systèmes Infinis

Grégoire Sutre

LaBRI

Ecole Jeunes Chercheurs en Algorithmique et Calcul Formel

Plan

Plan

Méthodes formelles : pourquoi ?

Systèmes critiques

Lorsqu'une défaillance peut entraîner des conséquences humaines/économiques importantes

Exemples

- systèmes de supervision (contrôle de procédé industriels, pilotage automatique, . . .)
- systèmes bancaires, protocoles de communication
- . . .

Objectif

Garantir formellement et si possible par des méthodes automatiques la correction de systèmes

Méthodes formelles : comment ?

Modélisation : $\mathcal{M}$

Description du système considéré dans un formalisme rigoureux

- en phase de conception, ou
- en phase de validation

Spécification : φ

Formalisation logique des propriétés attendues du système (cahier des charges)

Vérification

$$\mathcal{M} \overset{?}{\models} \varphi$$

Systèmes de Transitions

Définition

Un *système de transitions étiqueté* est un tuple $(S, \Sigma, \rightarrow)$ où :

- S est un ensemble de *configurations*
- Σ est un alphabet d'*actions*
- $\rightarrow \subseteq S \times \Sigma \times S$ est une relation de *transition*

Notations

- Chemin : $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \cdots s_k \xrightarrow{a_k} s_{k+1}$
- $s \xrightarrow{*} s'$ si s' est atteignable depuis s

Systèmes de Transitions

Définition

Un *système de transitions étiqueté* est un tuple $(S, \Sigma, \rightarrow)$ où :

- S est un ensemble de *configurations*
- Σ est un alphabet d'*actions*
- $\rightarrow \subseteq S \times \Sigma \times S$ est une relation de *transition*

Notations

- Chemin : $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \cdots s_k \xrightarrow{a_k} s_{k+1}$
- $s \xrightarrow{*} s'$ si s' est atteignable depuis s
- Successeurs et prédécesseurs directs :

$$\begin{cases} post(X, a) &= \{s' \in S \mid \exists s \in X, s \xrightarrow{a} s'\} \\ pre(X, a) &= \{s \in S \mid \exists s' \in X, s \xrightarrow{a} s'\} \end{cases}$$

Systèmes de Transitions

Définition

Un *système de transitions étiqueté* est un tuple $(S, \Sigma, \rightarrow)$ où :

- S est un ensemble de *configurations*
- Σ est un alphabet d'*actions*
- $\rightarrow \subseteq S \times \Sigma \times S$ est une relation de *transition*

Notations

- Chemin : $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \cdots s_k \xrightarrow{a_k} s_{k+1}$
- $s \xrightarrow{*} s'$ si s' est atteignable depuis s
- Successeurs et prédécesseurs indirects :

$$\begin{cases} post^*(X) &= \{s' \in S \mid \exists s \in X, s \xrightarrow{*} s'\} \\ pre^*(X) &= \{s \in S \mid \exists s' \in X, s \xrightarrow{*} s'\} \end{cases}$$

Spécification de propriétés de sûreté

Sureté

Le système ne doit admettre aucun mauvais comportement

$$\varphi = AG\ \neg Error$$

Spécification par ensemble de configurations

- $Error \subseteq S$
- Aucune configuration de $Error$ ne doit être atteignable

Spécification par ensemble d'actions

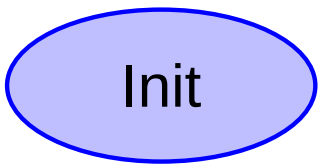
- $Error \subseteq \Sigma$
- Aucun chemin franchissable ne doit contenir d'action de $Error$

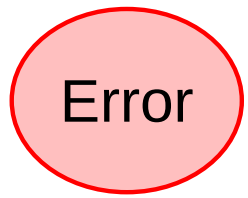
Model-checking classique (systèmes finis)

Vérification d'une propriété de sûreté

$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

Model-checking classique (systèmes finis)





Vérification d'une propriété de sûreté

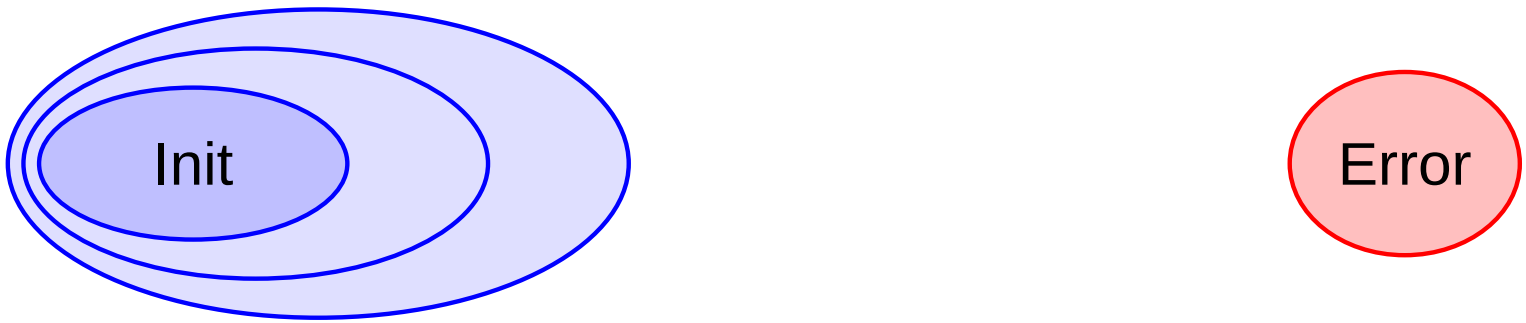
$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

Model-checking classique (systèmes finis)

Vérification d'une propriété de sûreté

$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

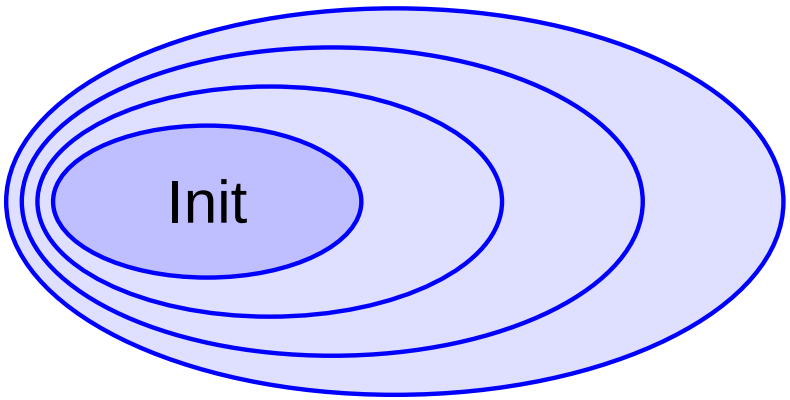
Model-checking classique (systèmes finis)

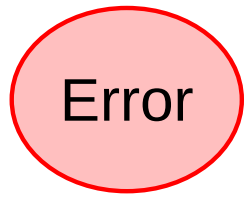


Vérification d'une propriété de sûreté

$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

Model-checking classique (systèmes finis)

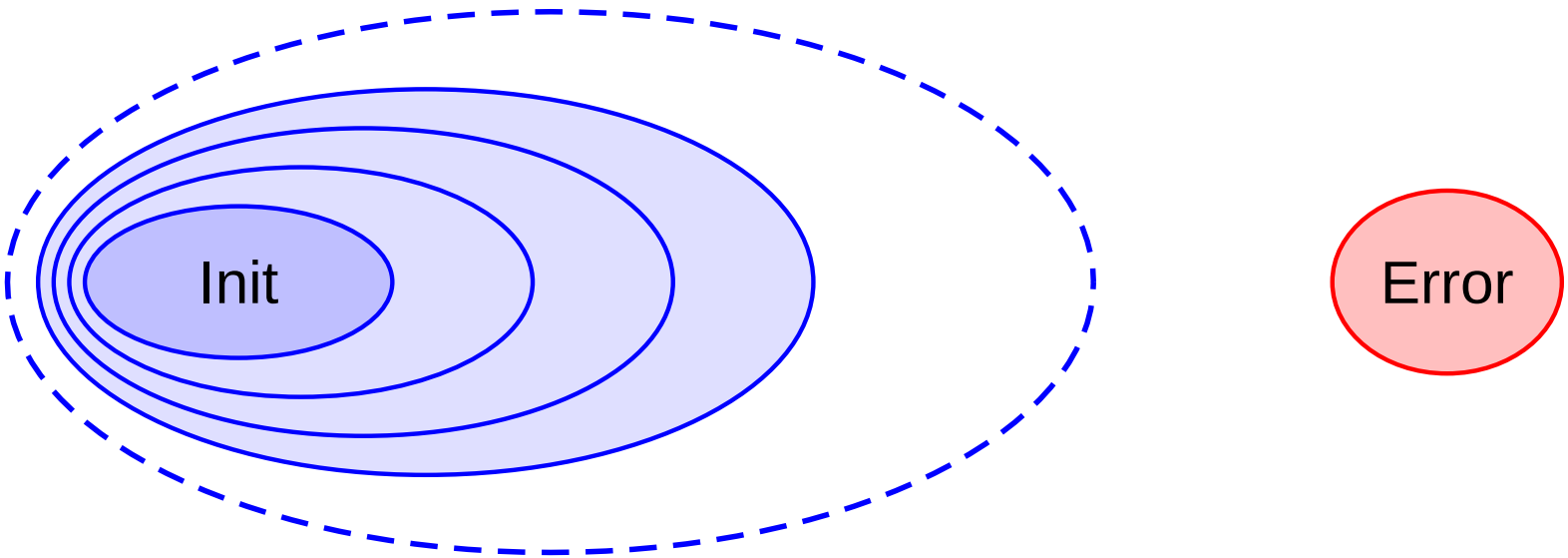




Vérification d'une propriété de sûreté

$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

Model-checking classique (systèmes finis)



Vérification d'une propriété de sûreté

$$\mathcal{M} \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

Systèmes infinis

Qu'est-ce que c'est ?

Modèle $\mathcal{M}$ dont la sémantique $[\![\mathcal{M}]\!]$ est un S.T. infini

D'où ça vient ?

Variables de $\mathcal{M}$ à valeurs dans des domaines infinis

Exemples

- $\mathbb{D} = \mathbb{Z}$: réseaux de Petri (processus dynamiques concurrents), systèmes à compteurs (variables entières)
- $\mathbb{D} = \Sigma^*$: automates à file (systèmes communicant par canaux FIFO), automates à pile (programmes récursifs)
- $\mathbb{D} = \mathbb{R}$: automates temporisés (horloges), systèmes hybrides (chronomètres, contrôle de processus physiques)

Vérification automatique des systèmes infinis

Besoin de nouvelles techniques

Le calcul explicite de $[\![\mathcal{M}]\!]$ est impossible

Indécidabilité

Le problème de l'atteignabilité est en général indécidable pour les systèmes infinis

Approches développées

- Recherche de classes décidables et d'algorithmes dédiés
- Exploration explicite à la volée avec troncature
- Calcul symbolique de $[\![\mathcal{M}]\!]$
- Vérification par abstraction

Vérification automatique des systèmes infinis

Besoin de nouvelles techniques

Le calcul explicite de $[\![\mathcal{M}]\!]$ est impossible

Indécidabilité

Le problème de l'atteignabilité est en général indécidable pour les systèmes infinis

Approches développées

- Recherche de classes décidables et d'algorithmes dédiés
- Exploration explicite à la volée avec troncature
- Calcul symbolique de $[\![\mathcal{M}]\!]$
- Vérification par abstraction

Vérification automatique des systèmes infinis

Besoin de nouvelles techniques

Le calcul explicite de $[\![\mathcal{M}]\!]$ est impossible

Indécidabilité

Le problème de l'atteignabilité est en général indécidable pour les systèmes infinis

Approches développées

- Recherche de classes décidables et d'algorithmes dédiés
- Exploration explicite à la volée avec troncature
- Calcul symbolique de $[\![\mathcal{M}]\!]$
- Vérification par abstraction

Vérification automatique des systèmes infinis

Besoin de nouvelles techniques

Le calcul explicite de $[\![\mathcal{M}]\!]$ est impossible

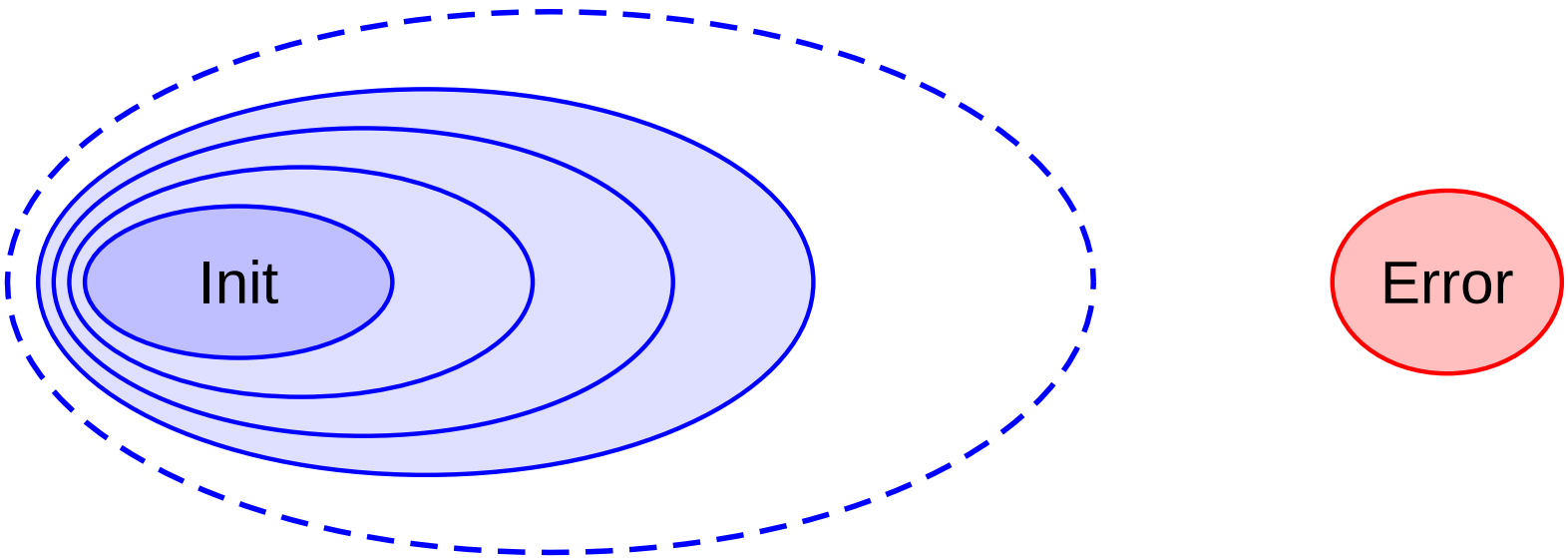
Indécidabilité

Le problème de l'atteignabilité est en général indécidable pour les systèmes infinis

Approches développées

- Recherche de classes décidables et d'algorithmes dédiés
- Exploration explicite à la volée avec troncature
- Calcul symbolique de $[\![\mathcal{M}]\!]$
- Vérification par abstraction

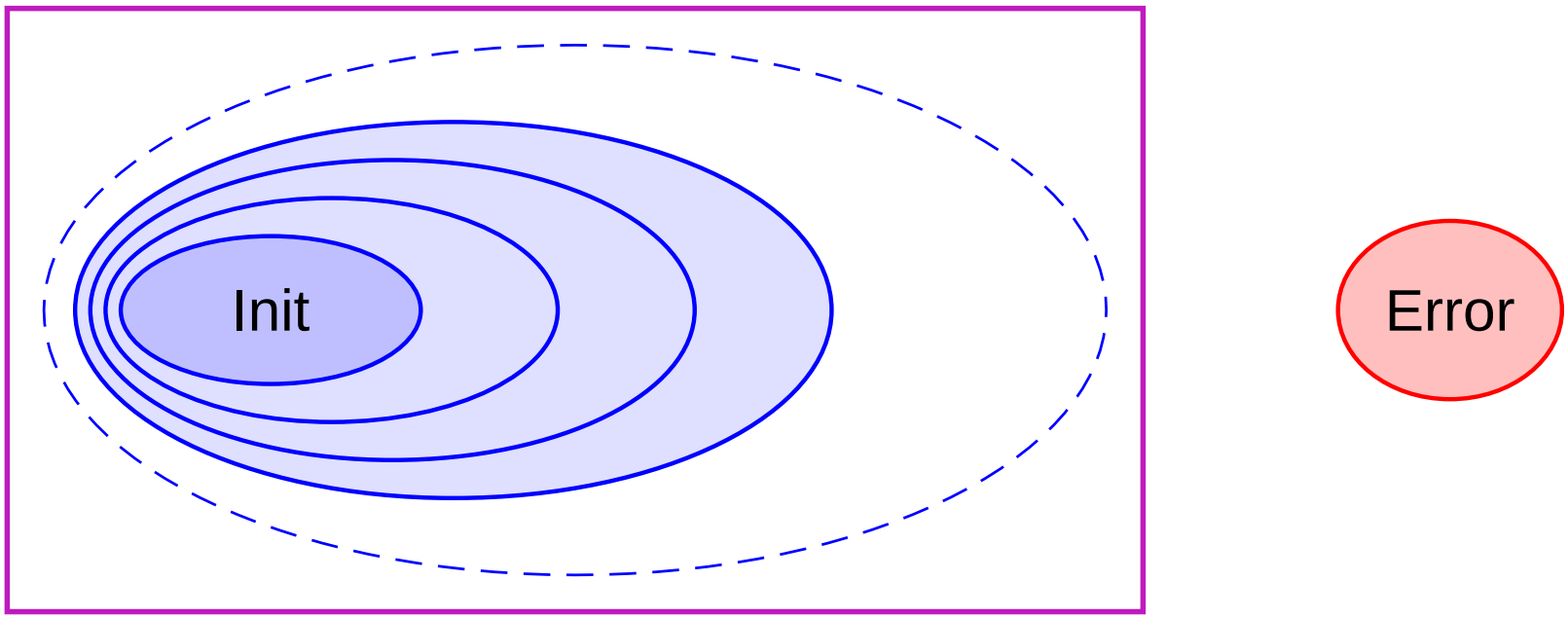
Model-checking classique (rappel)



Vérification d'une propriété de sûreté

$$[\![\mathcal{M}]\!] \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

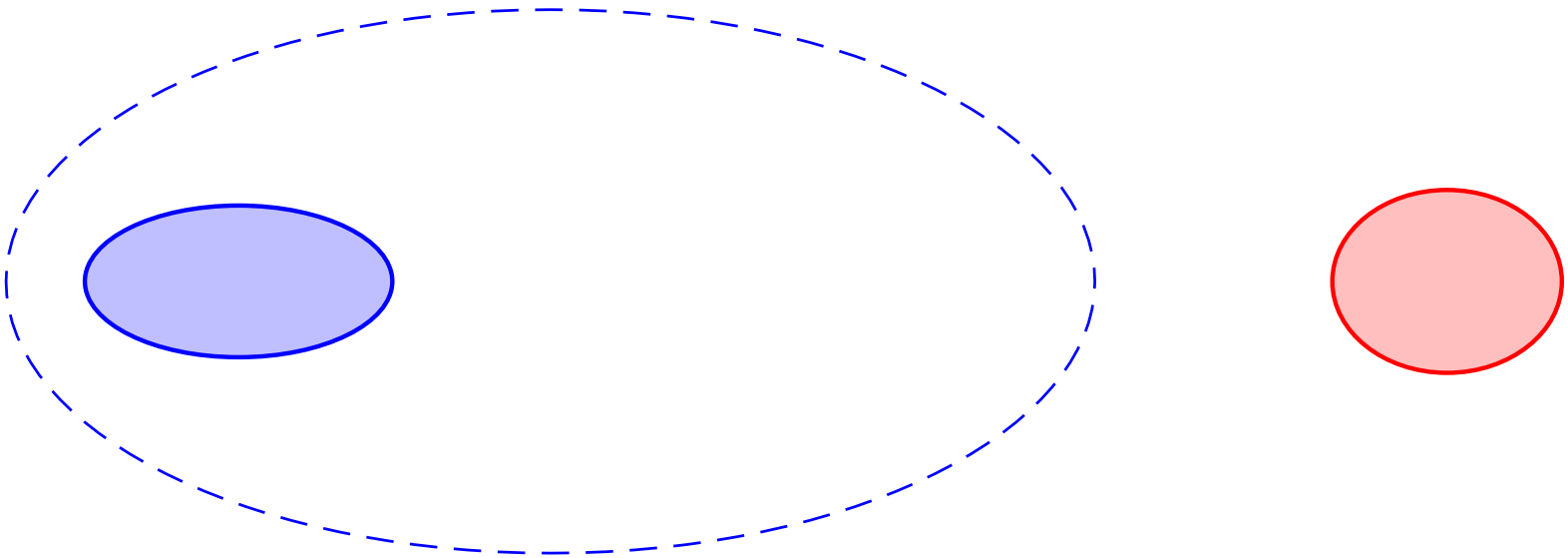
Model-checking classique (rappel)



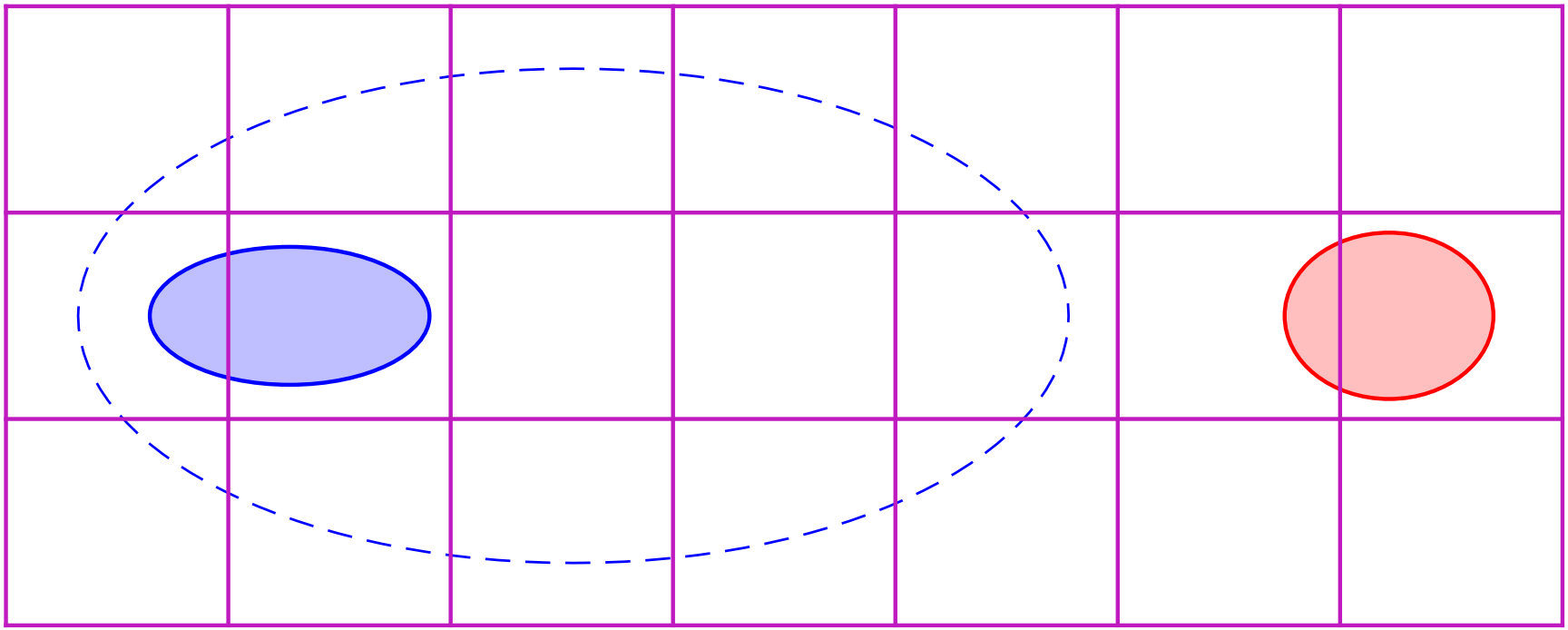
Vérification d'une propriété de sûreté

$$[\![\mathcal{M}]\!] \models AG\ \neg Error \quad \text{ssi} \quad post^*(Init) \cap [\![Error]\!] = \emptyset$$

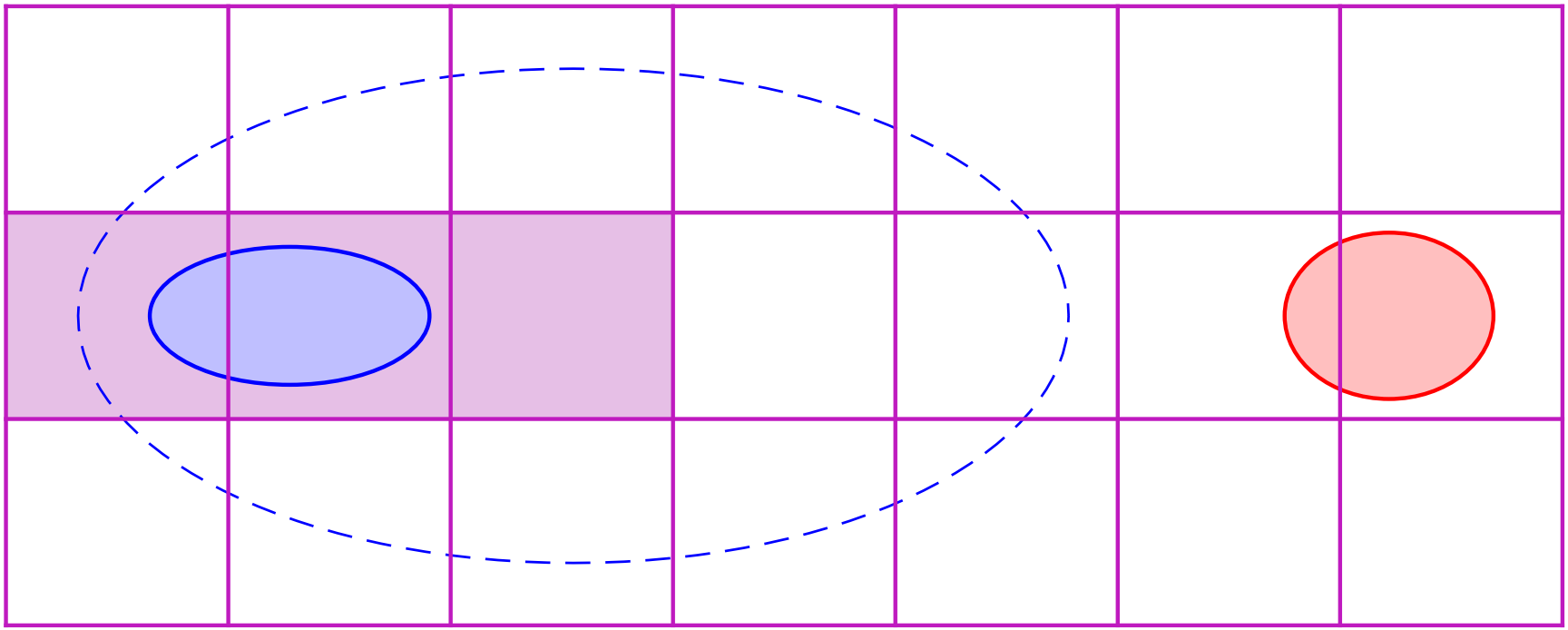
Abstraction par prédicats [Graf-Saïdi 97]



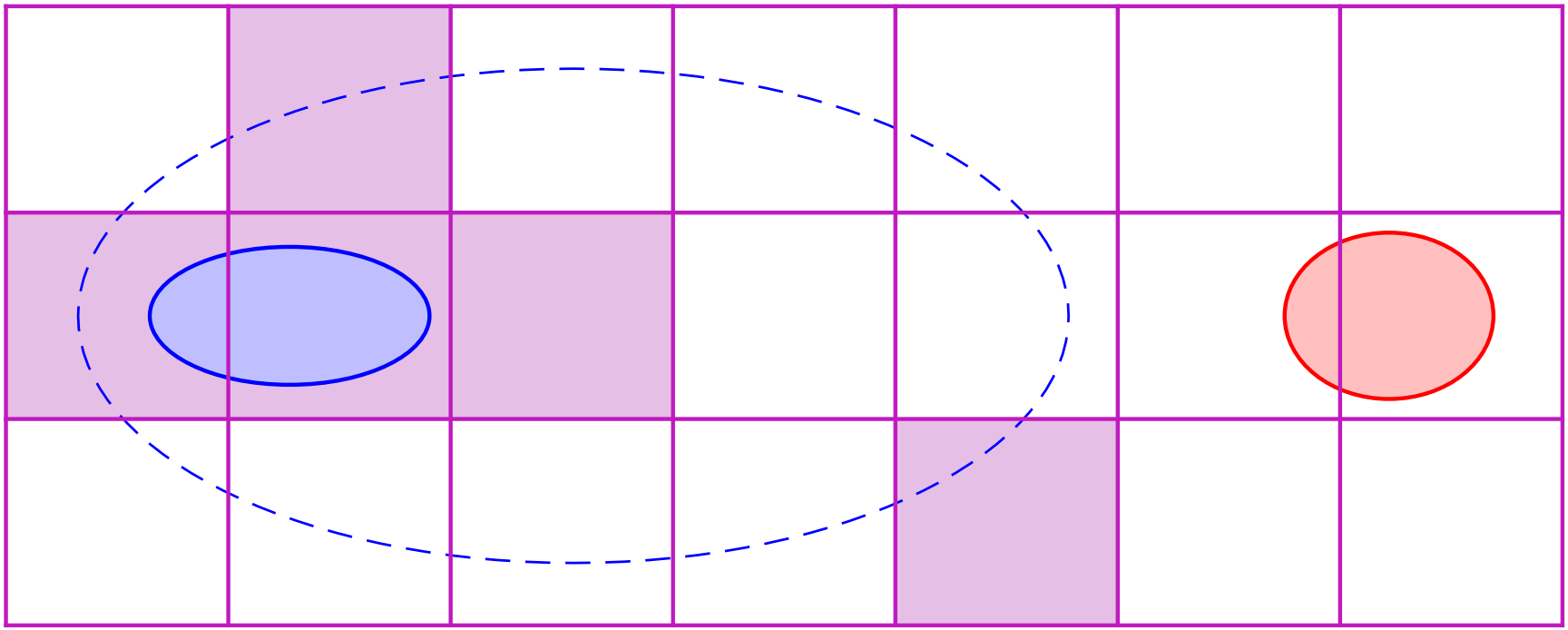
Abstraction par prédicats [Graf-Saïdi 97]



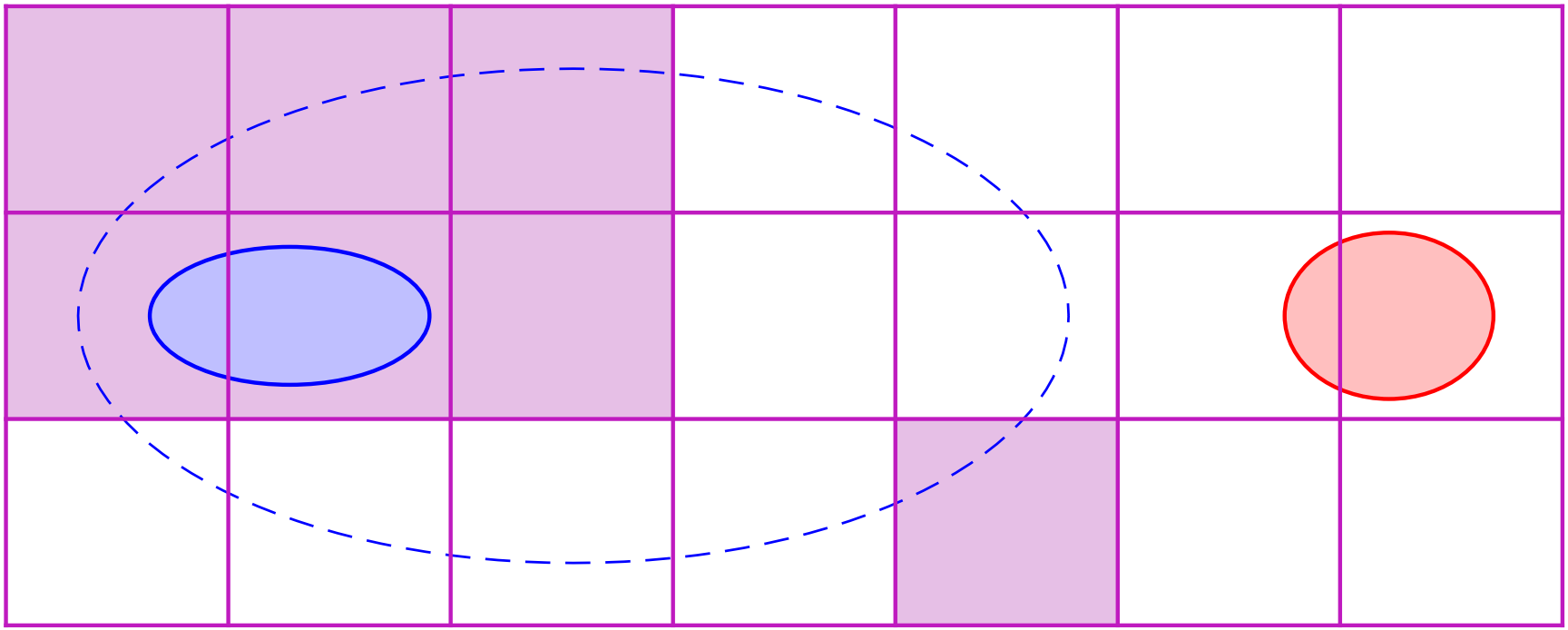
Abstraction par prédicats [Graf-Saïdi 97]



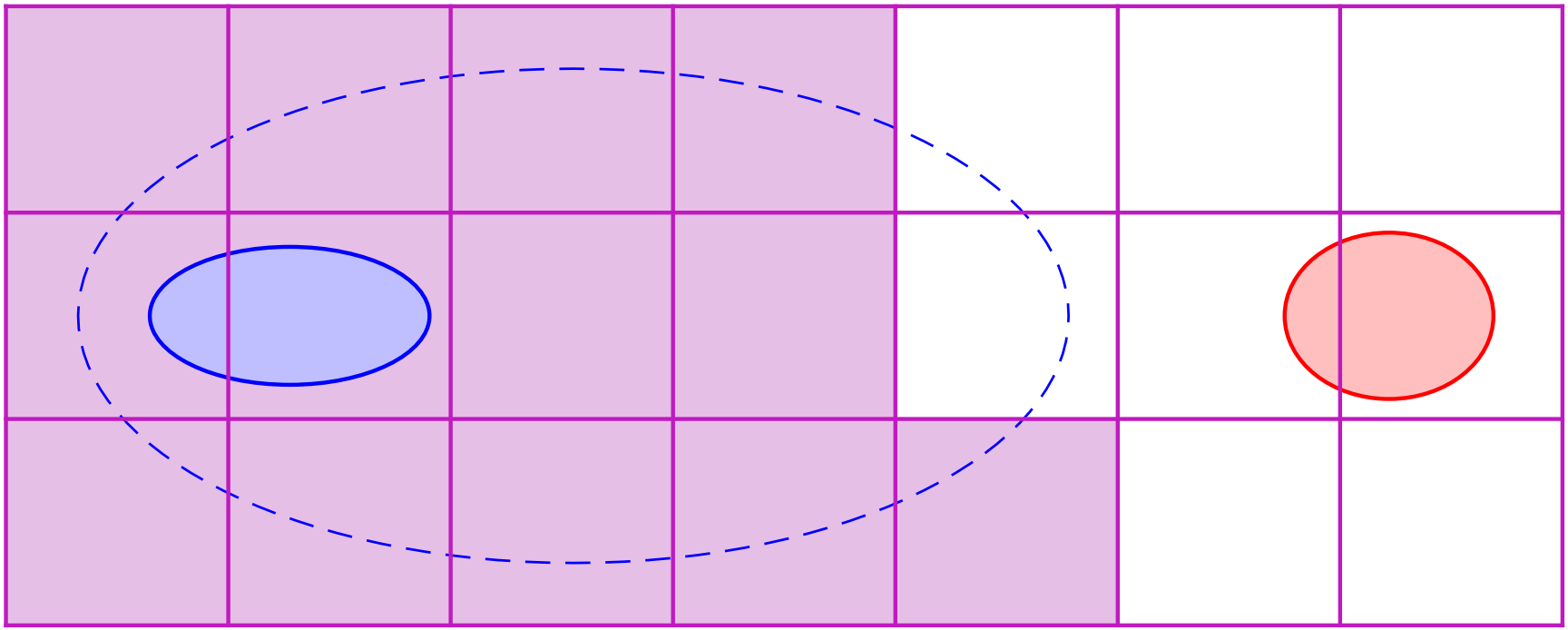
Abstraction par prédicats [Graf-Saïdi 97]



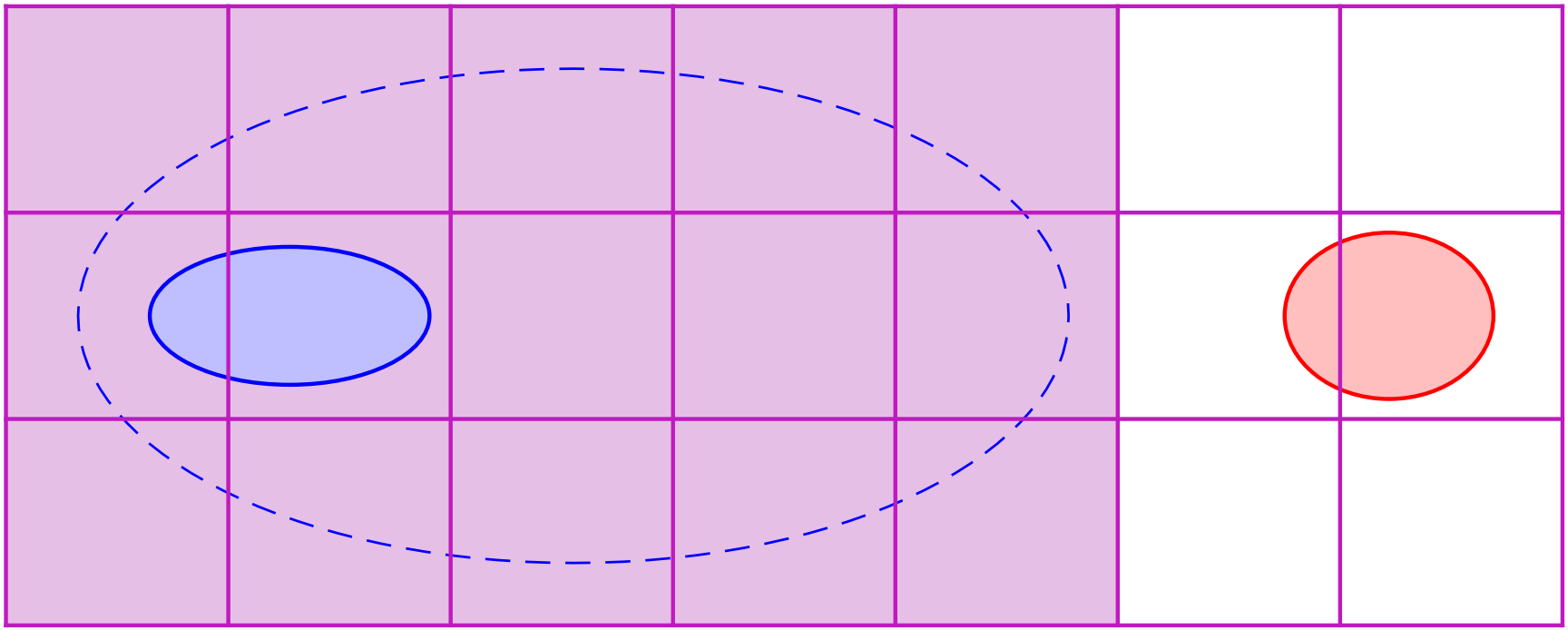
Abstraction par prédicats [Graf-Saïdi 97]



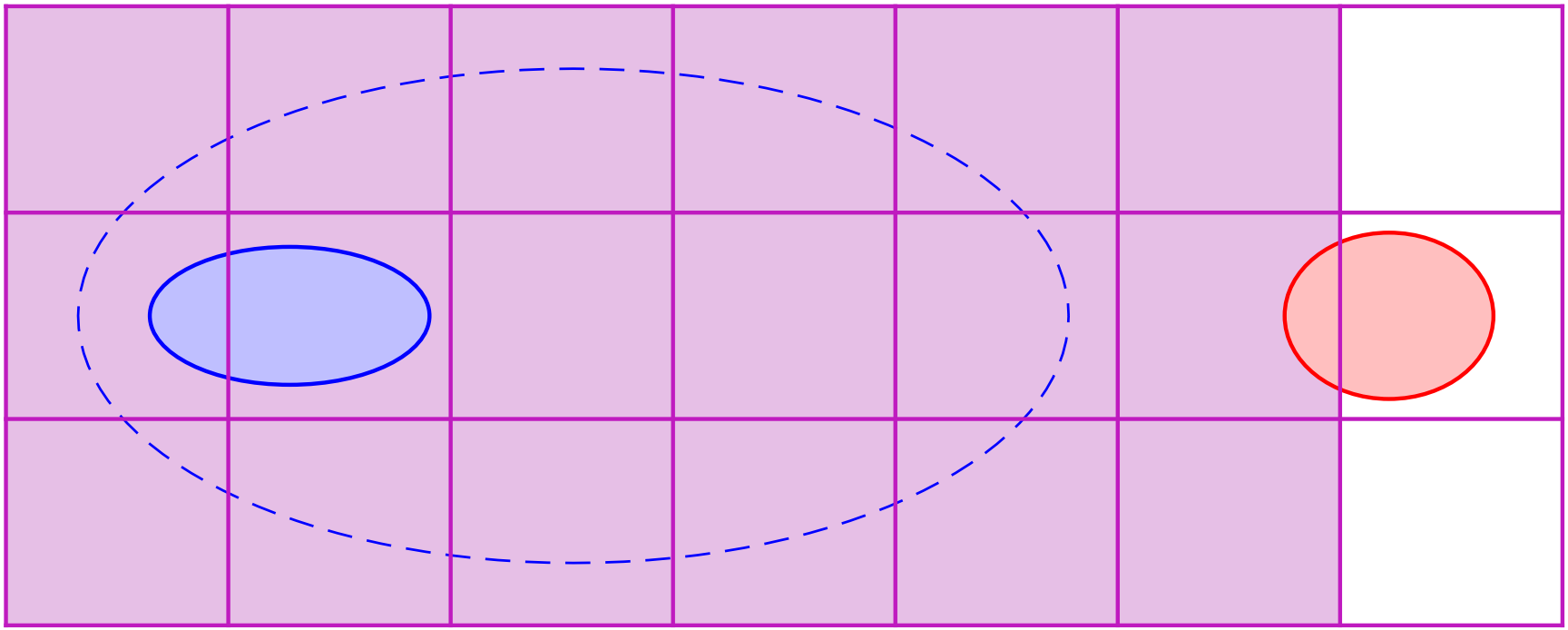
Abstraction par prédicats [Graf-Saïdi 97]



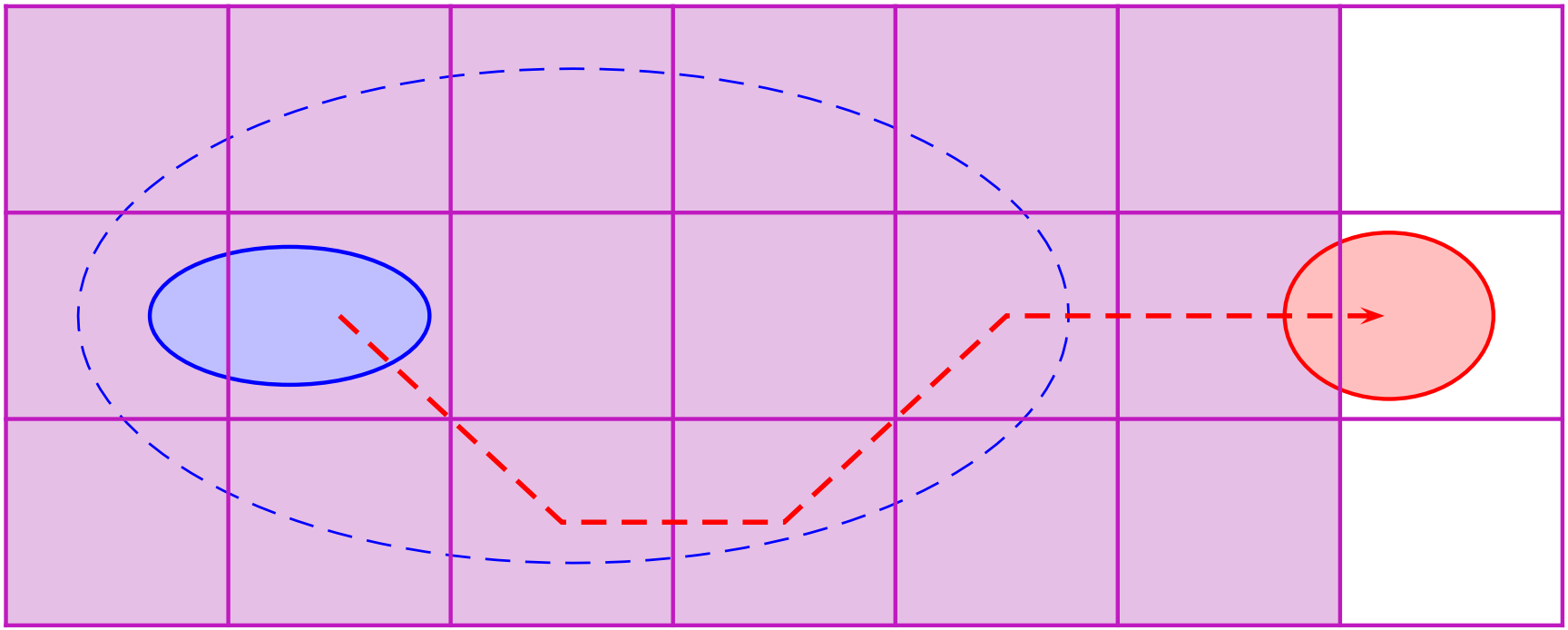
Abstraction par prédicats [Graf-Saïdi 97]



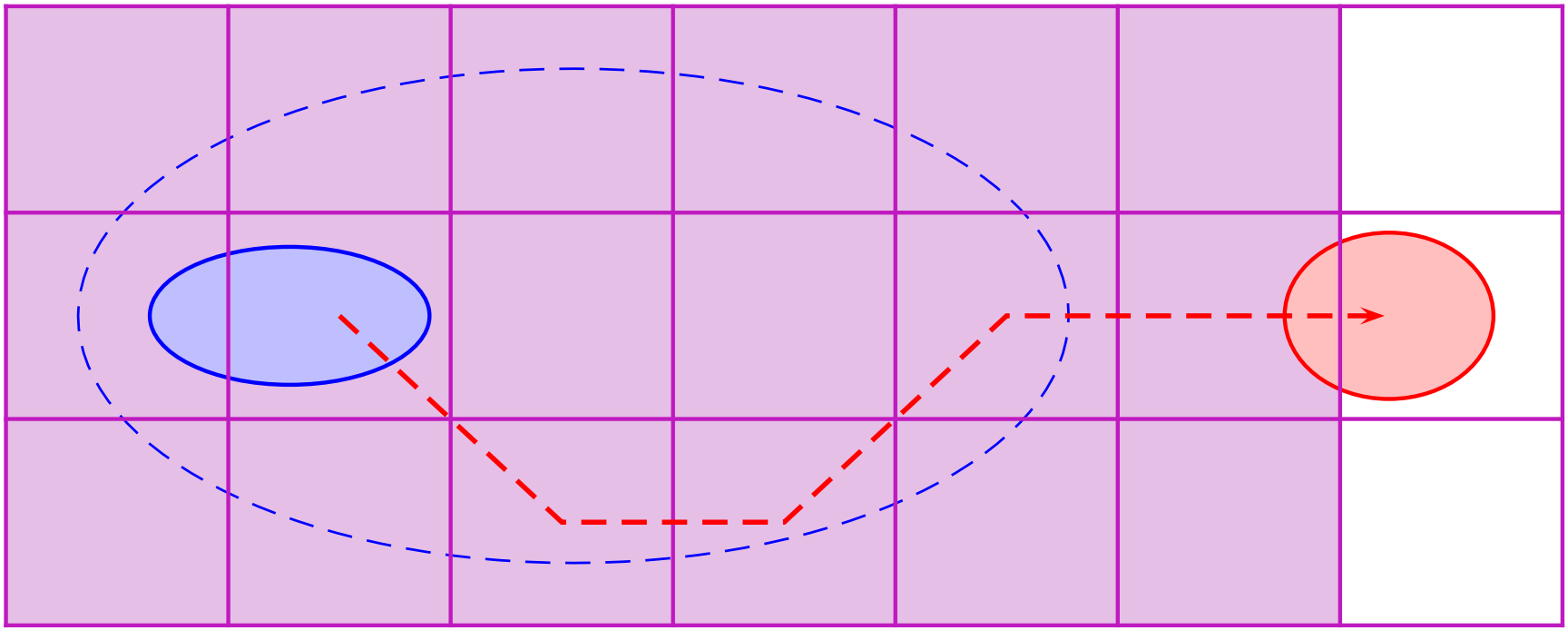
Abstraction par prédicats [Graf-Saïdi 97]



Abstraction par prédicats [Graf-Saïdi 97]



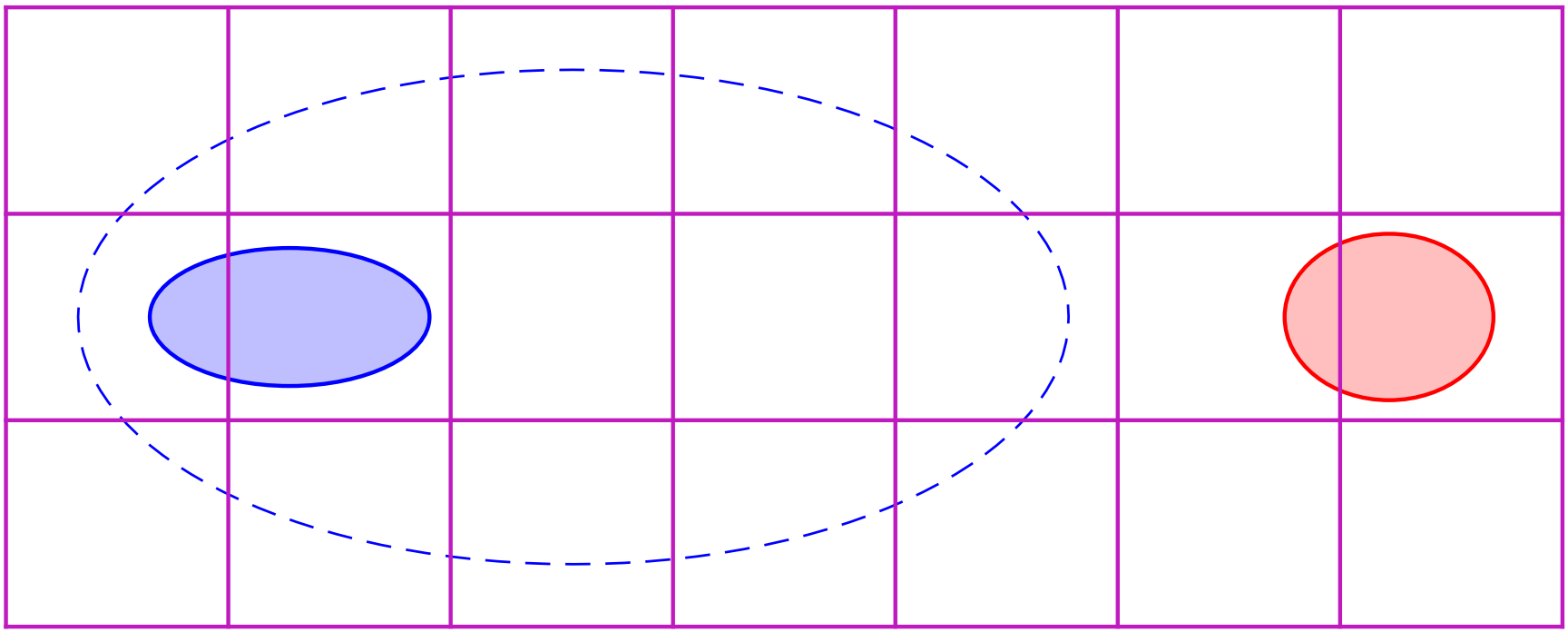
Abstraction par prédicats [Graf-Saïdi 97]



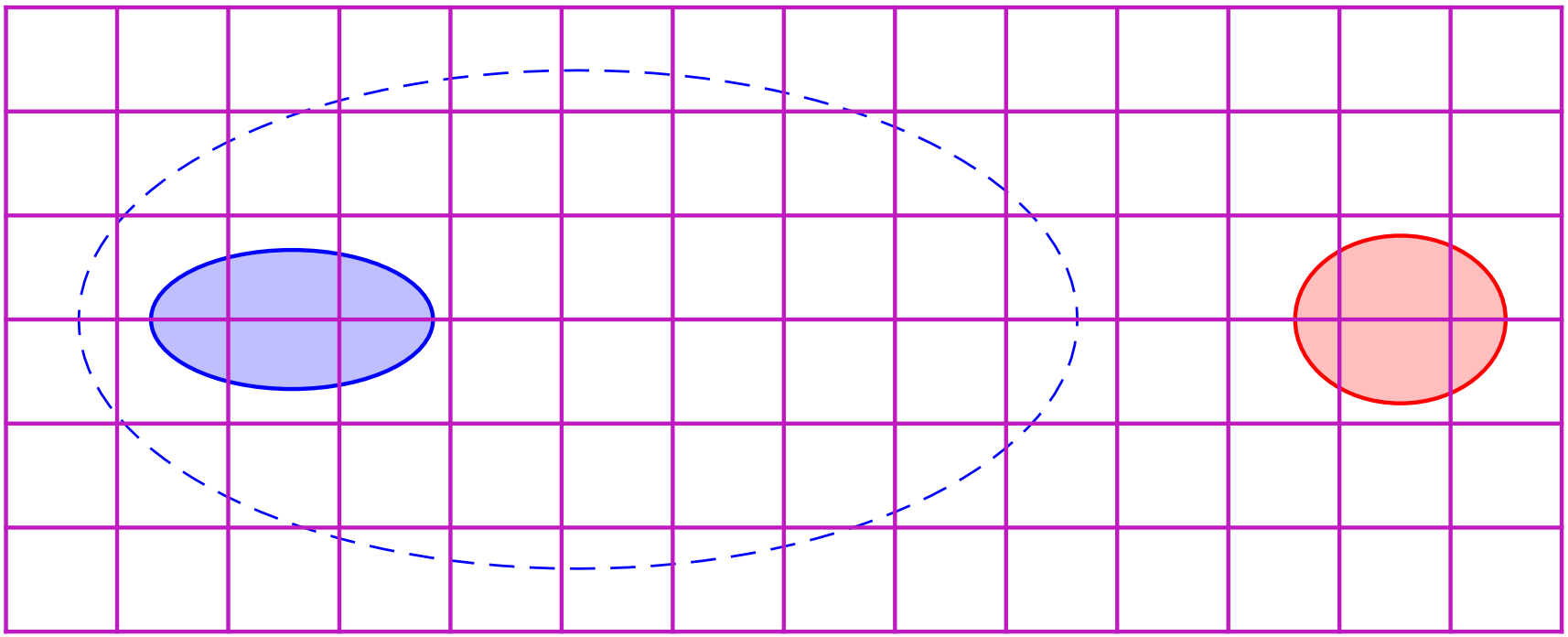
Analyse du contre-exemple abstrait

- Abstraction trop grossière
- Raffinement pour éviter ce contre-exemple erroné

Raffinement de l'abstraction par prédicats



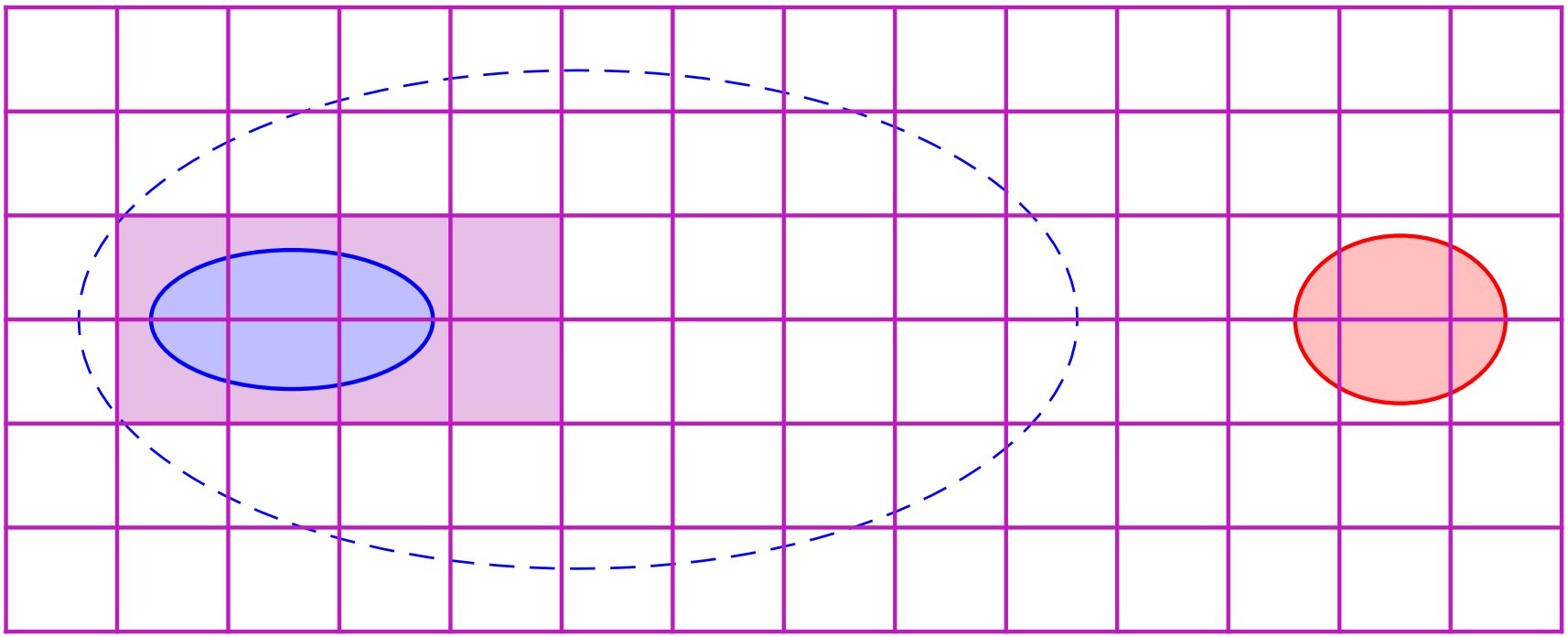
Raffinement de l'abstraction par prédicats



Raffinement

- Ajout de nouveaux prédicats : découpage des boîtes
- Le système est correct !

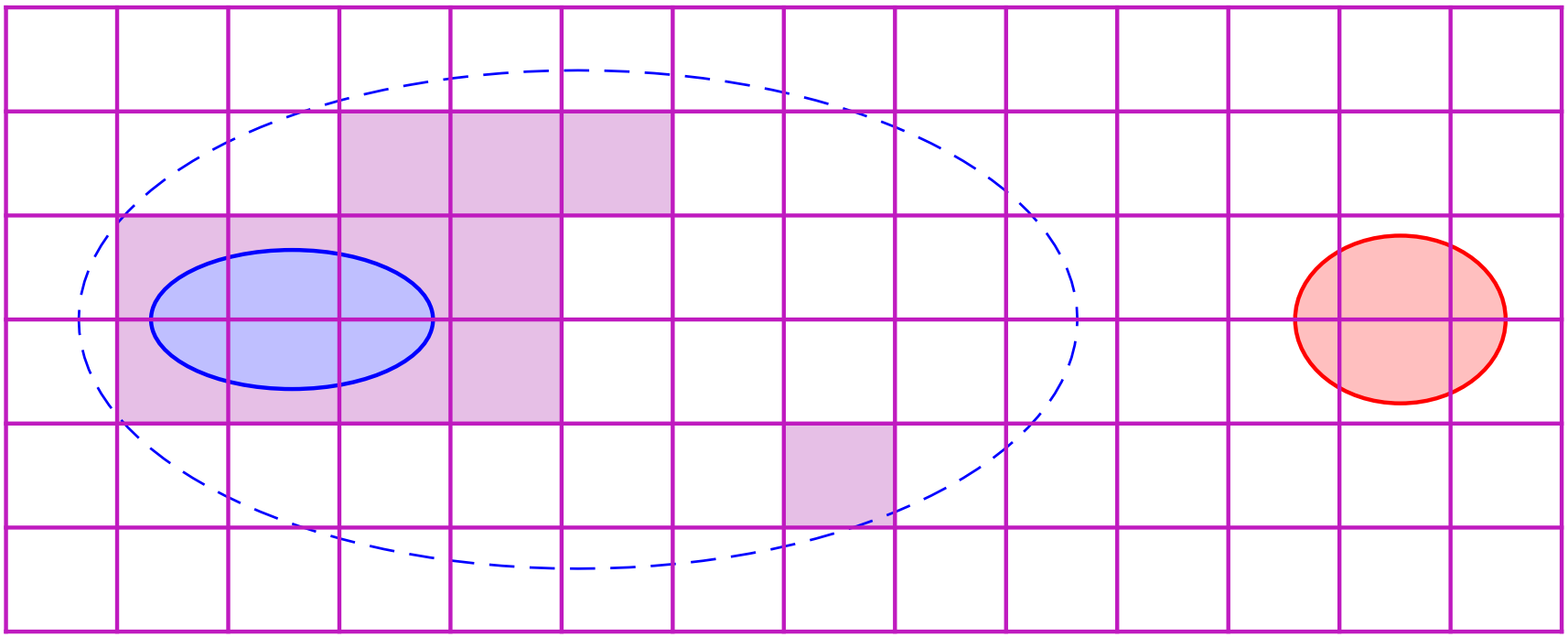
Raffinement de l'abstraction par prédicats



Raffinement

- Ajout de nouveaux prédicats : découpage des boîtes
- Le système est correct !

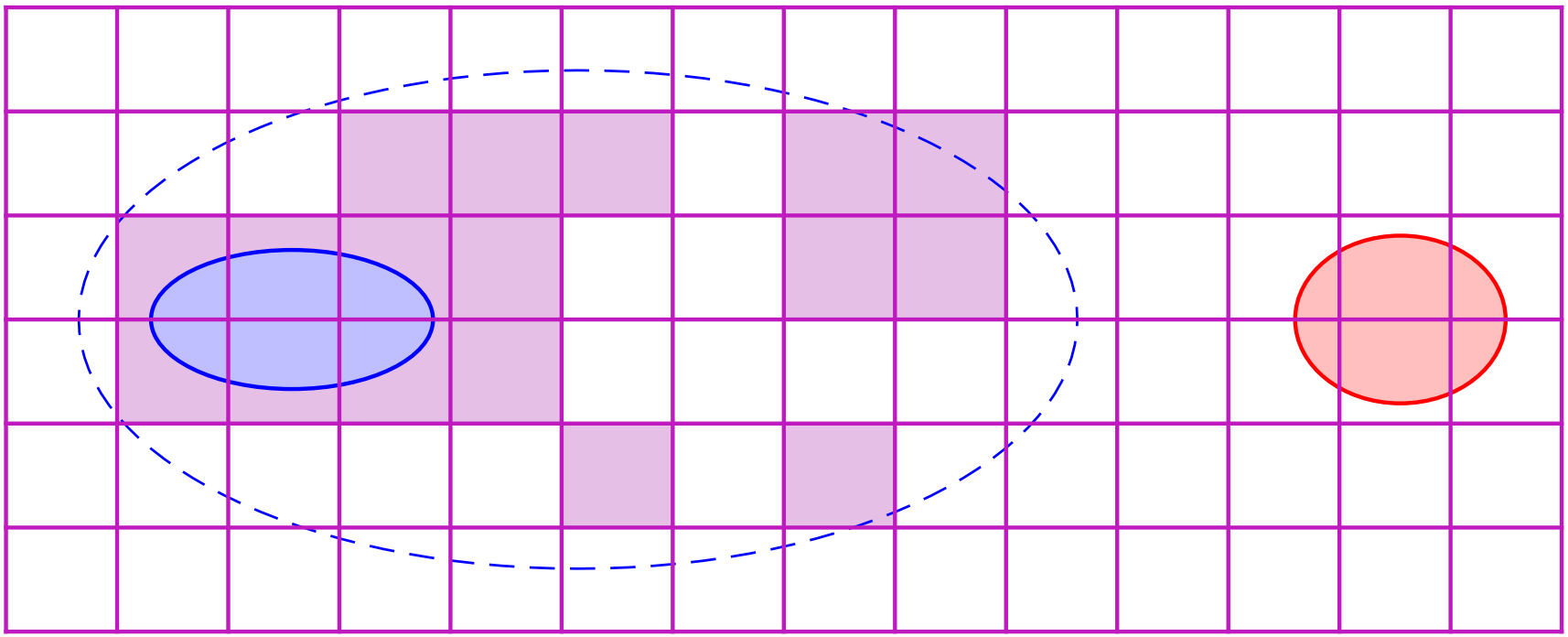
Raffinement de l'abstraction par prédicats



Raffinement

- Ajout de nouveaux prédicats : découpage des boîtes
- Le système est correct !

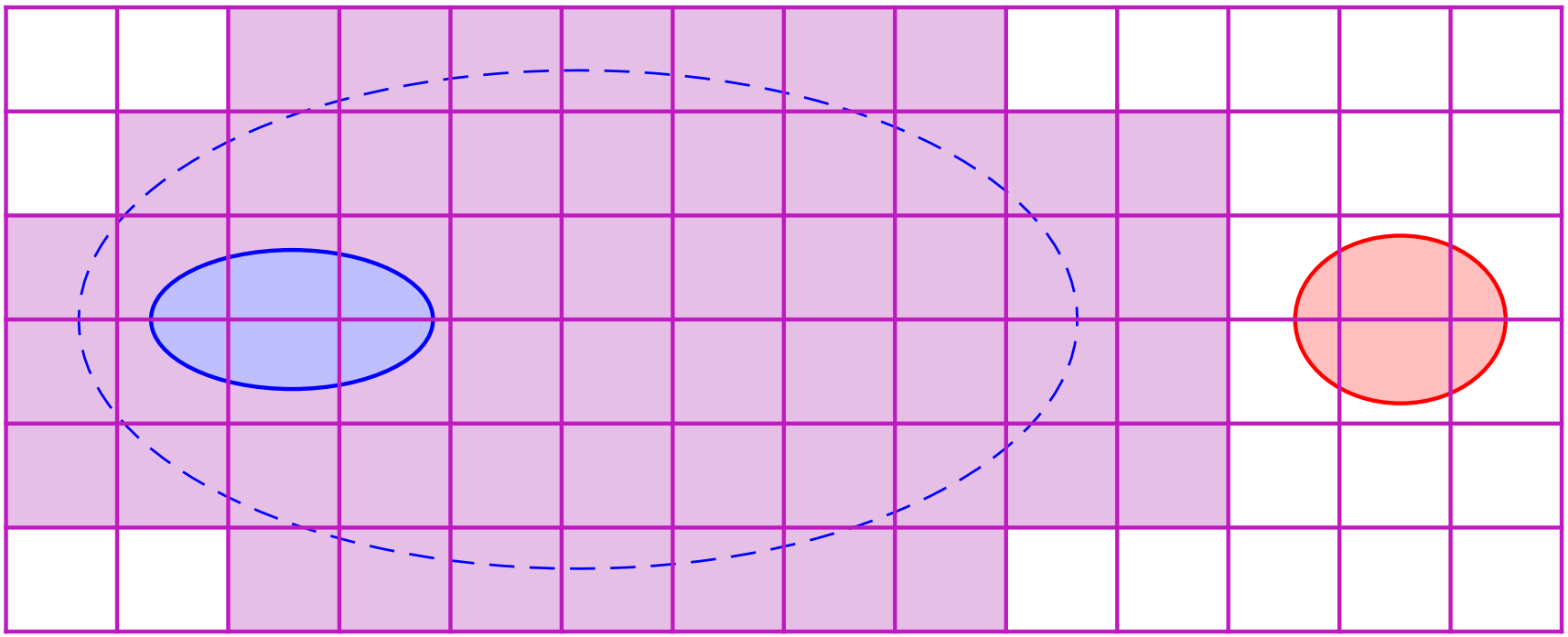
Raffinement de l'abstraction par prédicats



Raffinement

- Ajout de nouveaux prédicats : découpage des boîtes
- Le système est correct !

Raffinement de l'abstraction par prédicats

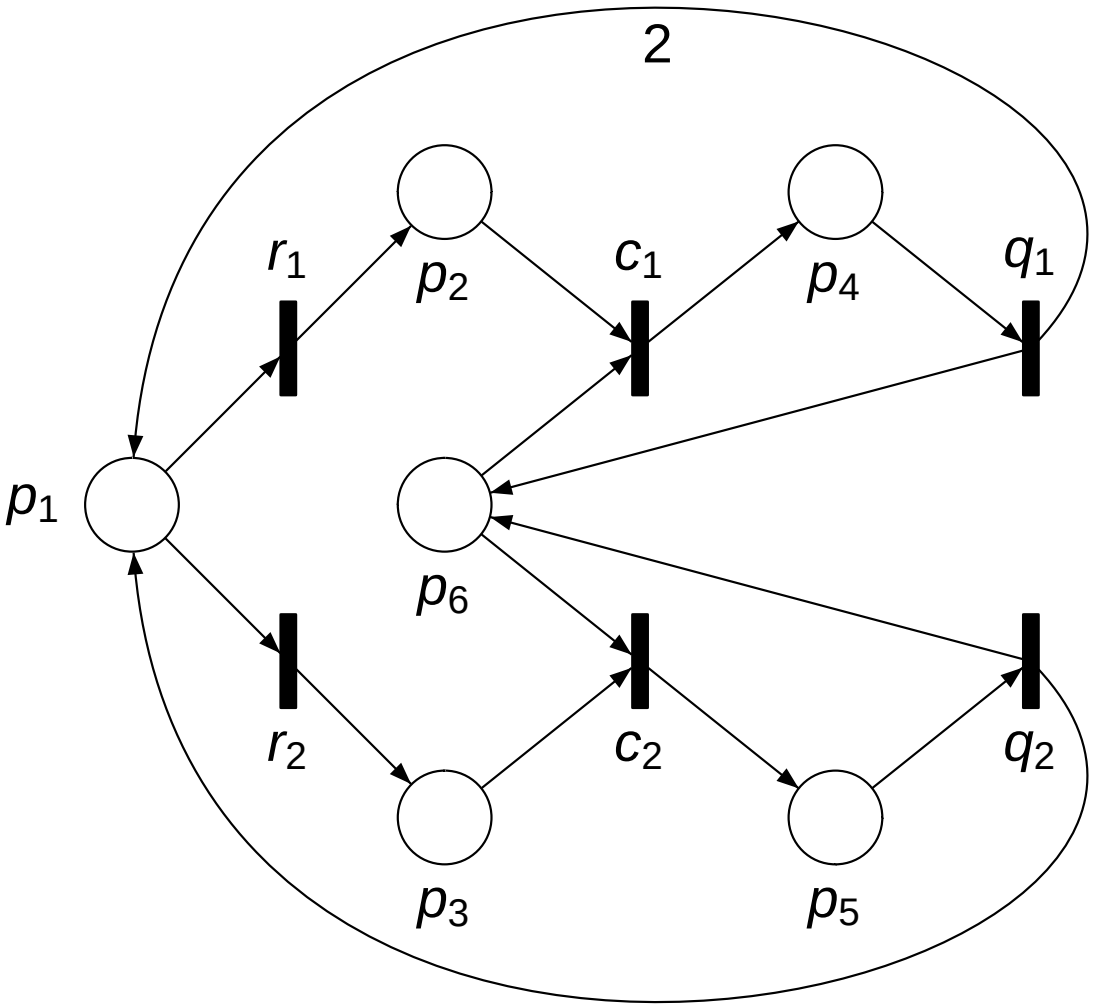


Raffinement

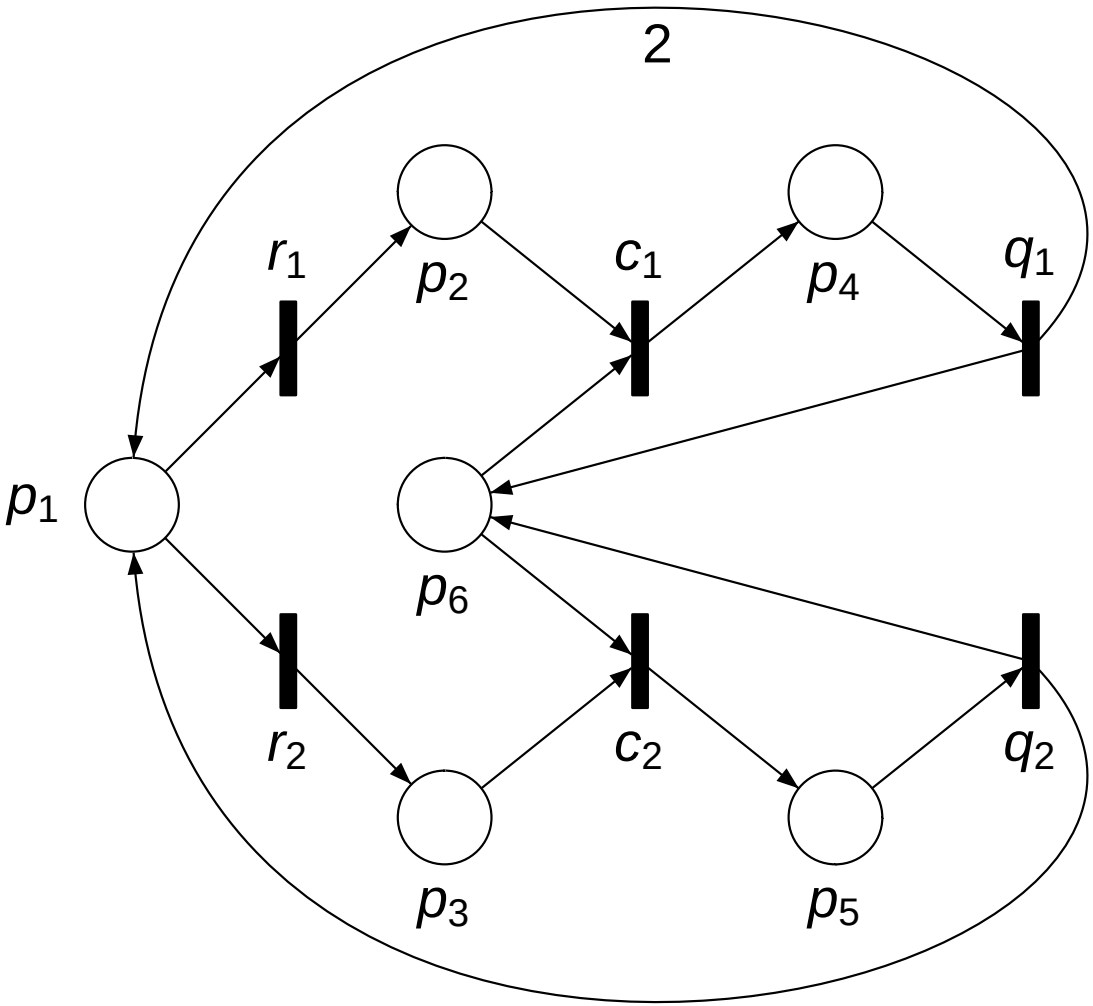
- Ajout de nouveaux prédicats : découpage des boîtes
- Le système est correct !

Plan

Exemple de réseau de Petri : exclusion mutuelle



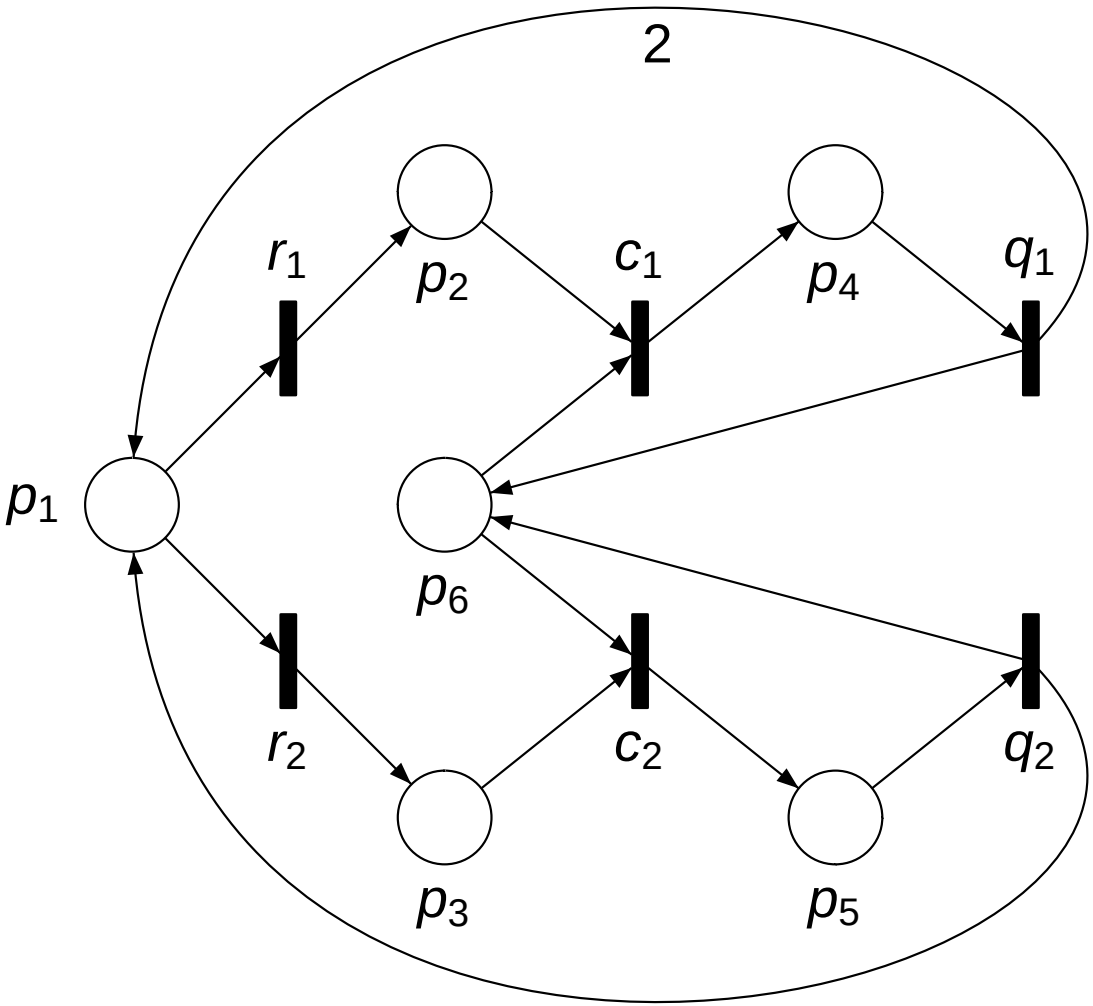
Exemple de réseau de Petri : exclusion mutuelle



Places

- p_1 : inactif
- p_2, p_3 : en attente
- p_4, p_5 : en section critique
- p_6 : verrou

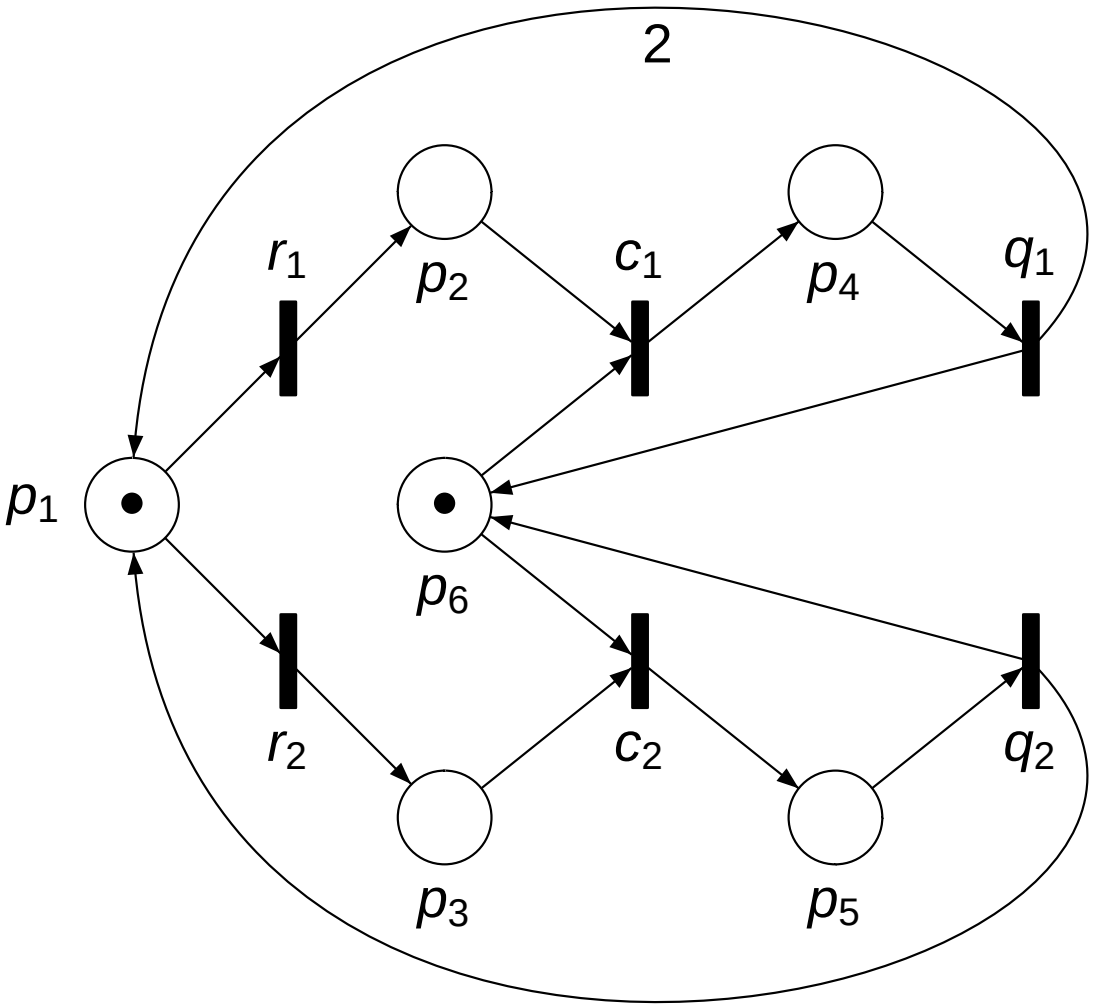
Exemple de réseau de Petri : exclusion mutuelle



Transitions

- r_1, r_2 : requête d'accès
- c_1, c_2 : entrée en section critique
- q_1, q_2 : sortie de section critique

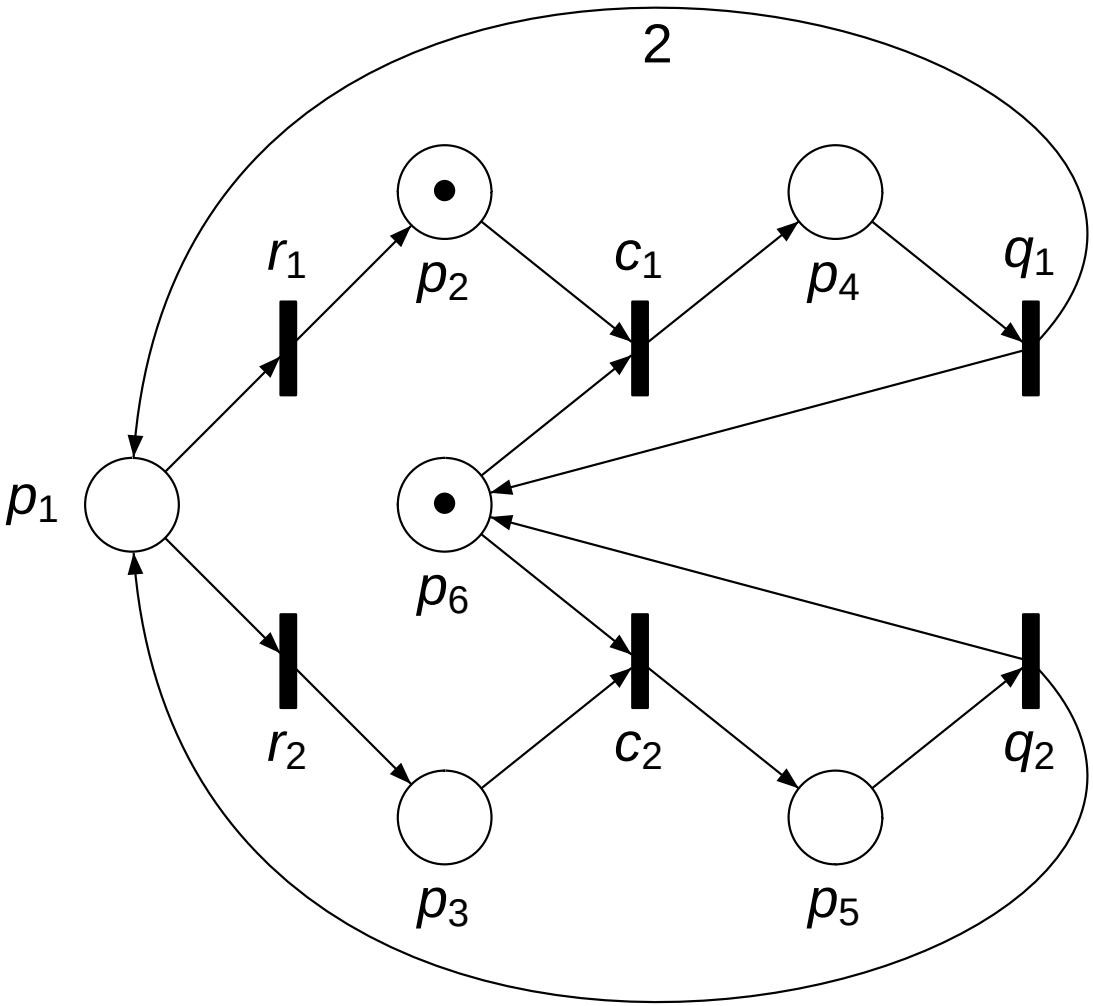
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

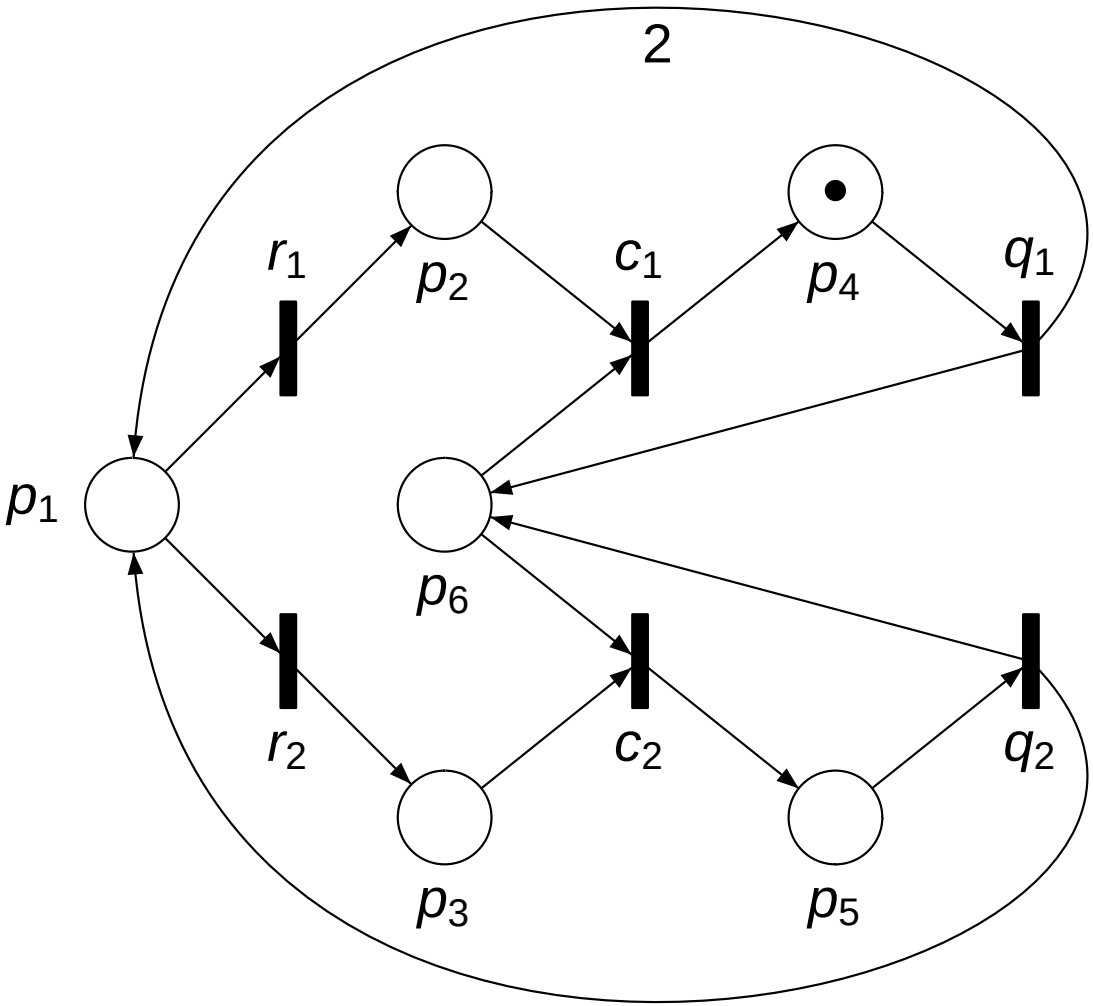
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

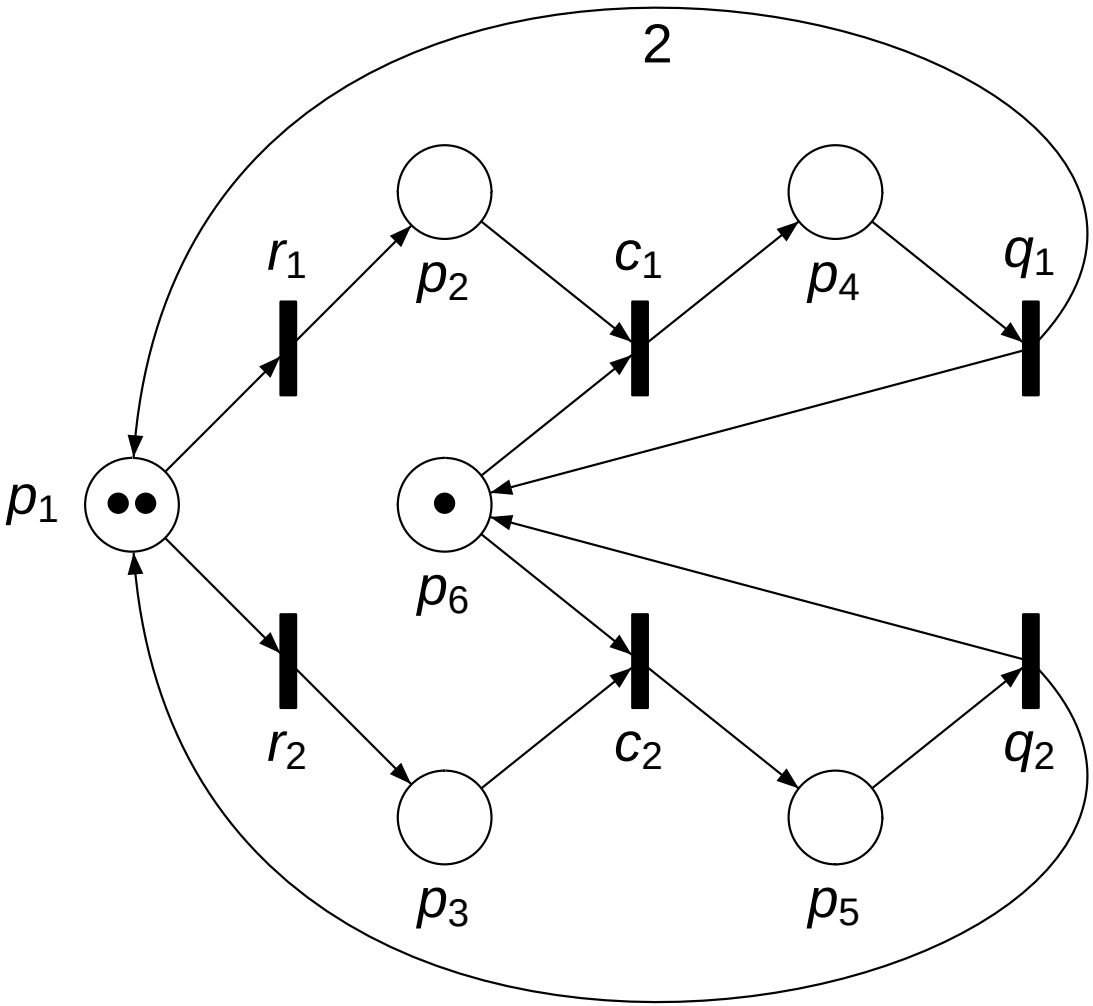
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

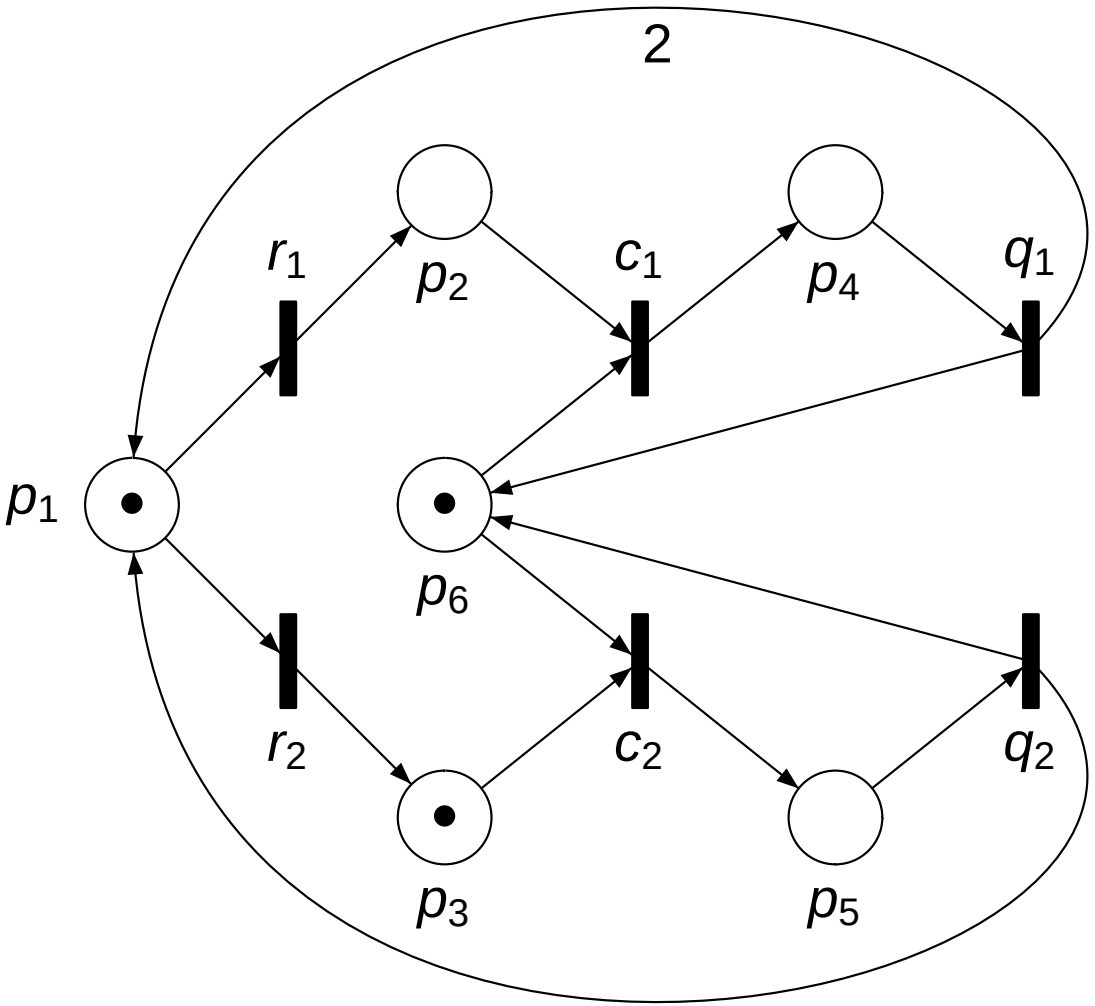
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

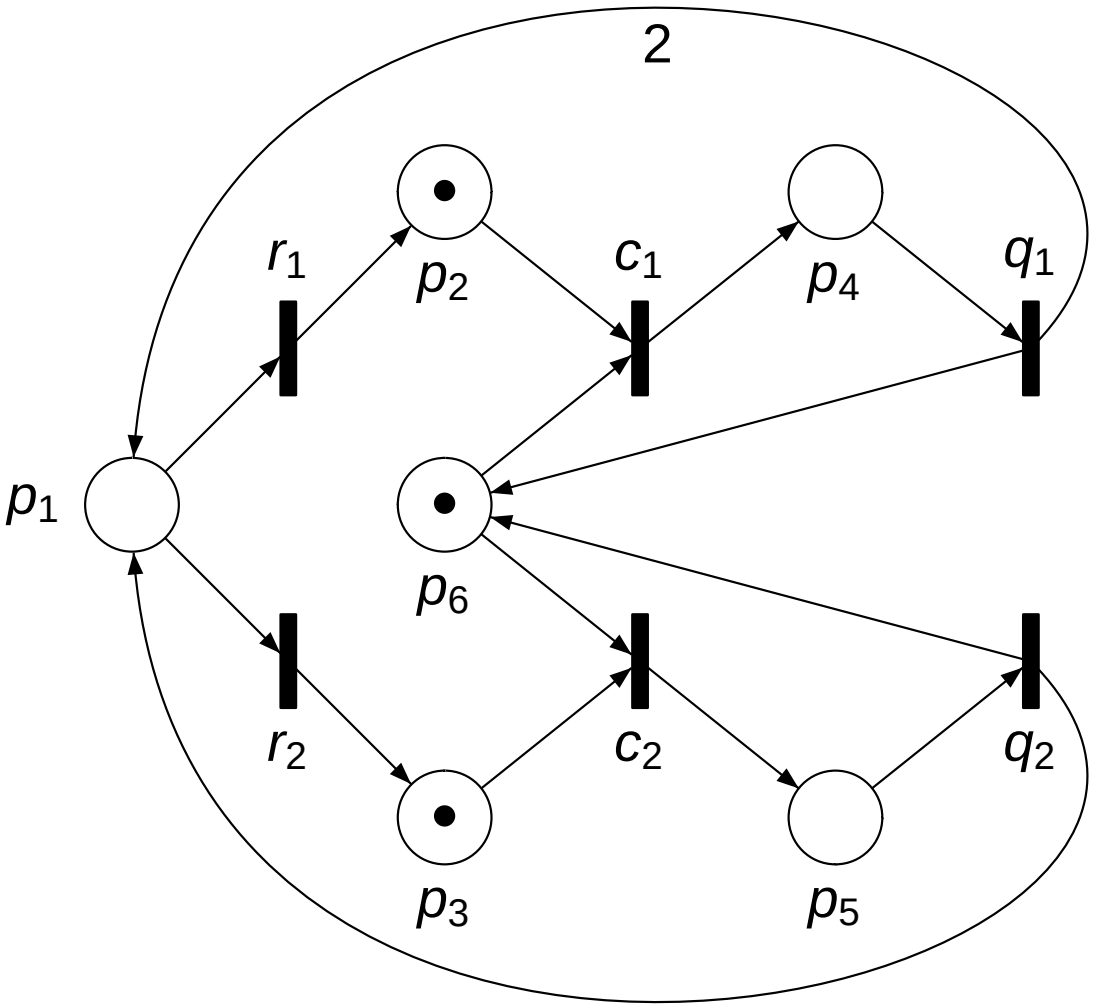
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

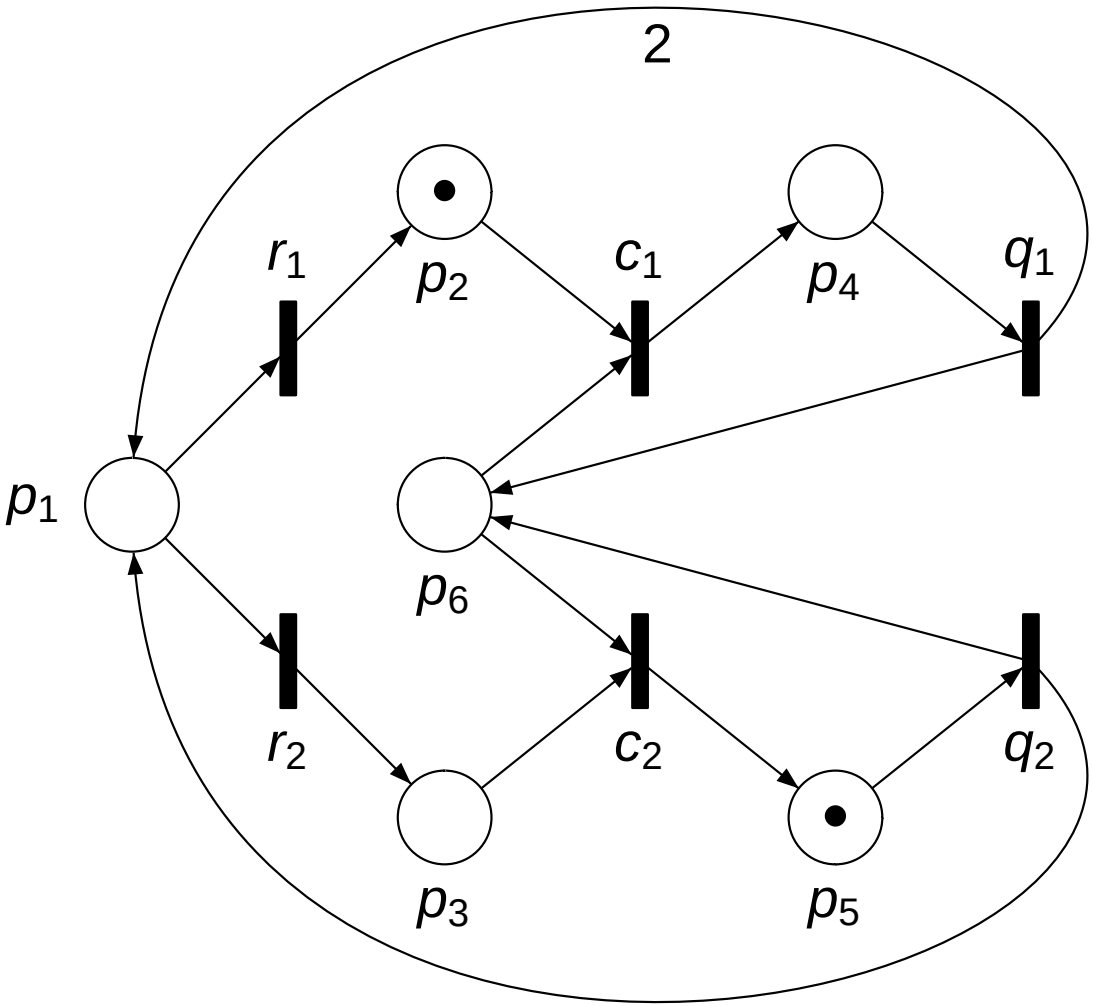
Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

Exemple de réseau de Petri : exclusion mutuelle



Dynamique

- jetons dans les places
- activation de transitions
- franchissement de transitions

Réseaux de Petri : syntaxe et sémantique

Syntaxe

Un *réseau de Petri* est un triplet $\mathcal{N} = (P, T, F)$ où :

- P : ensemble de *places*
- T : ensemble de *transitions*
- $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ est la fonction de flot

Sémantique

Système de transitions $(S, \Sigma, \to)$ où $S = \mathbb{N}^P$, $\Sigma = T$ et :

$$s \xrightarrow{t} s' \text{ ssi pour tout } p \in P, \begin{cases} s(p) \geq F(p,t) \\ s'(p) = s(p) - F(p,t) + F(t,p) \end{cases}$$

Réseaux de Petri : syntaxe et sémantique

Syntaxe

Un *réseau de Petri* est un triplet $\mathcal{N} = (P, T, F)$ où :

- P : ensemble de *places*
- T : ensemble de *transitions*
- $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ est la fonction de flot

Sémantique

Système de transitions $(S, \Sigma, \to)$ où $S = \mathbb{N}^P$, $\Sigma = T$ et :

$$s \xrightarrow{t} s' \text{ ssi pour tout } p \in P, \begin{cases} s(p) \geq F(p,t) \\ s'(p) = s(p) - F(p,t) + F(t,p) \end{cases}$$

Sûreté : couverture et quasi-vivacité

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

Problème de la Quasi-vivacité

Etant donné un réseau $\mathcal{N}$, $s_0 \in \mathbb{N}^P$, et $t \in T$ existe-t-il un chemin franchissable depuis s_0 contenant t ?

Remarque

Une transition t est quasi-vivante ssi s est couvert, où $s(p) = F(p, t)$.

Sûreté : couverture et quasi-vivacité

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

Problème de la Quasi-vivacité

Etant donné un réseau $\mathcal{N}$, $s_0 \in \mathbb{N}^P$, et $t \in T$ existe-t-il un chemin franchissable depuis s_0 contenant t ?

Remarque

Une transition t est quasi-vivante ssi s est couvert, où $s(p) = F(p, t)$.

Sûreté : couverture et quasi-vivacité

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

Problème de la Quasi-vivacité

Etant donné un réseau $\mathcal{N}$, $s_0 \in \mathbb{N}^P$, et $t \in T$ existe-t-il un chemin franchissable depuis s_0 contenant t ?

Remarque

Une transition t est quasi-vivante ssi s est couvert, où $s(p) = F(p,t)$.

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

Idée

- on cherche les s_m minimaux tels que $s_m \xrightarrow{*} s' \geq s$
- on exécute le réseau en arrière
- s' est couvert ssi $s_0 \geq s_m \xrightarrow{*} s' \geq s$ pour un des s_m

Terminaison garantie par le lemme de Dickson

Tout sous-ensemble de $\mathbb{N}^P$ admet un nombre fini d'éléments minimaux

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

Idée

- on cherche les s_m minimaux tels que $s_m \xrightarrow{*} s' \geq s$
- on exécute le réseau en arrière
- s' est couvert ssi $s_0 \geq s_m \xrightarrow{*} s' \geq s$ pour un des s_m

Terminaison garantie par le lemme de Dickson

Tout sous-ensemble de $\mathbb{N}^P$ admet un nombre fini d'éléments minimaux

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un réseau $\mathcal{N}$ et $s_0, s \in \mathbb{N}^P$, existe-t-il $s' \geq s$ atteignable depuis s_0 ?

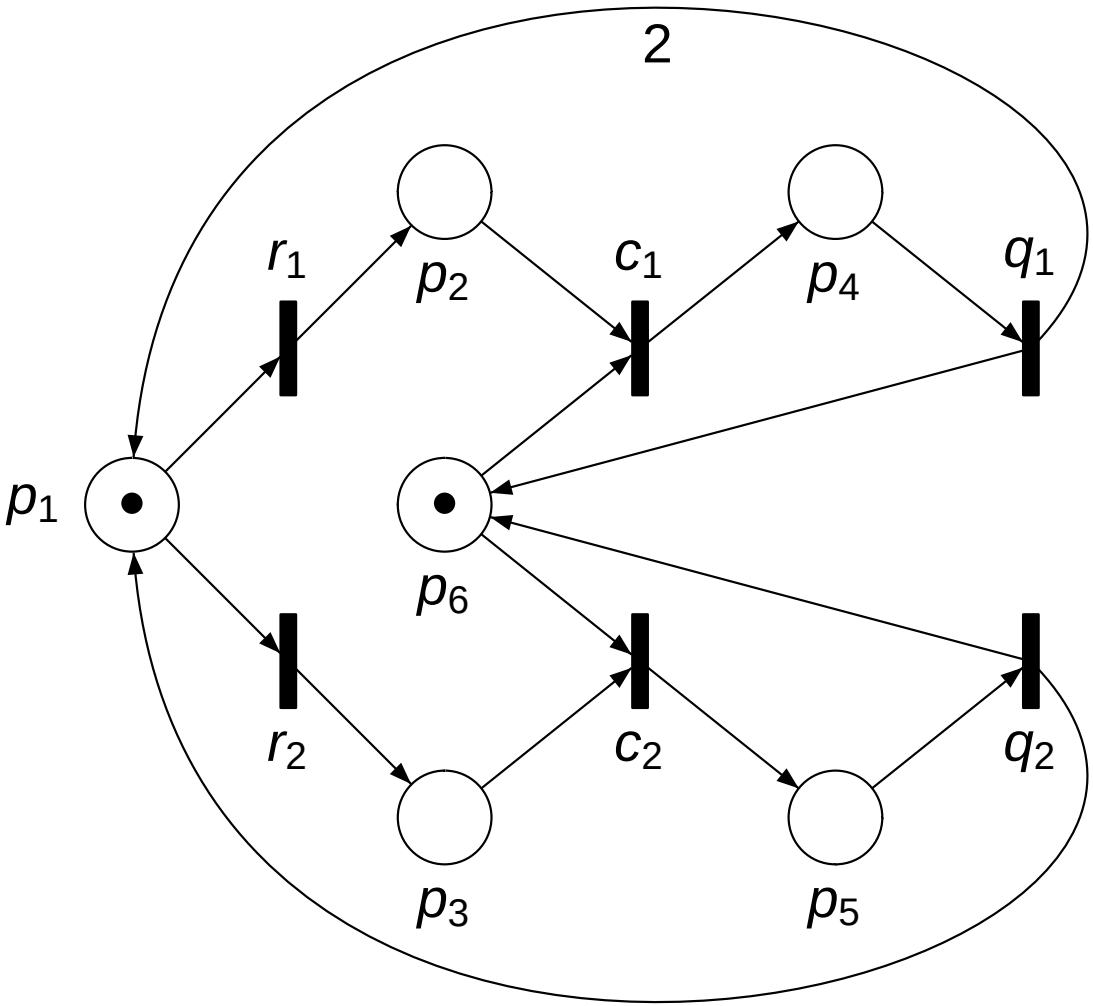
Idée

- on cherche les s_m minimaux tels que $s_m \xrightarrow{*} s' \geq s$
- on exécute le réseau en arrière
- s' est couvert ssi $s_0 \geq s_m \xrightarrow{*} s' \geq s$ pour un des s_m

Terminaison garantie par le lemme de Dickson

Tout sous-ensemble de $\mathbb{N}^P$ admet un nombre fini d'éléments minimaux

Analyse en arrière sur l'exemple



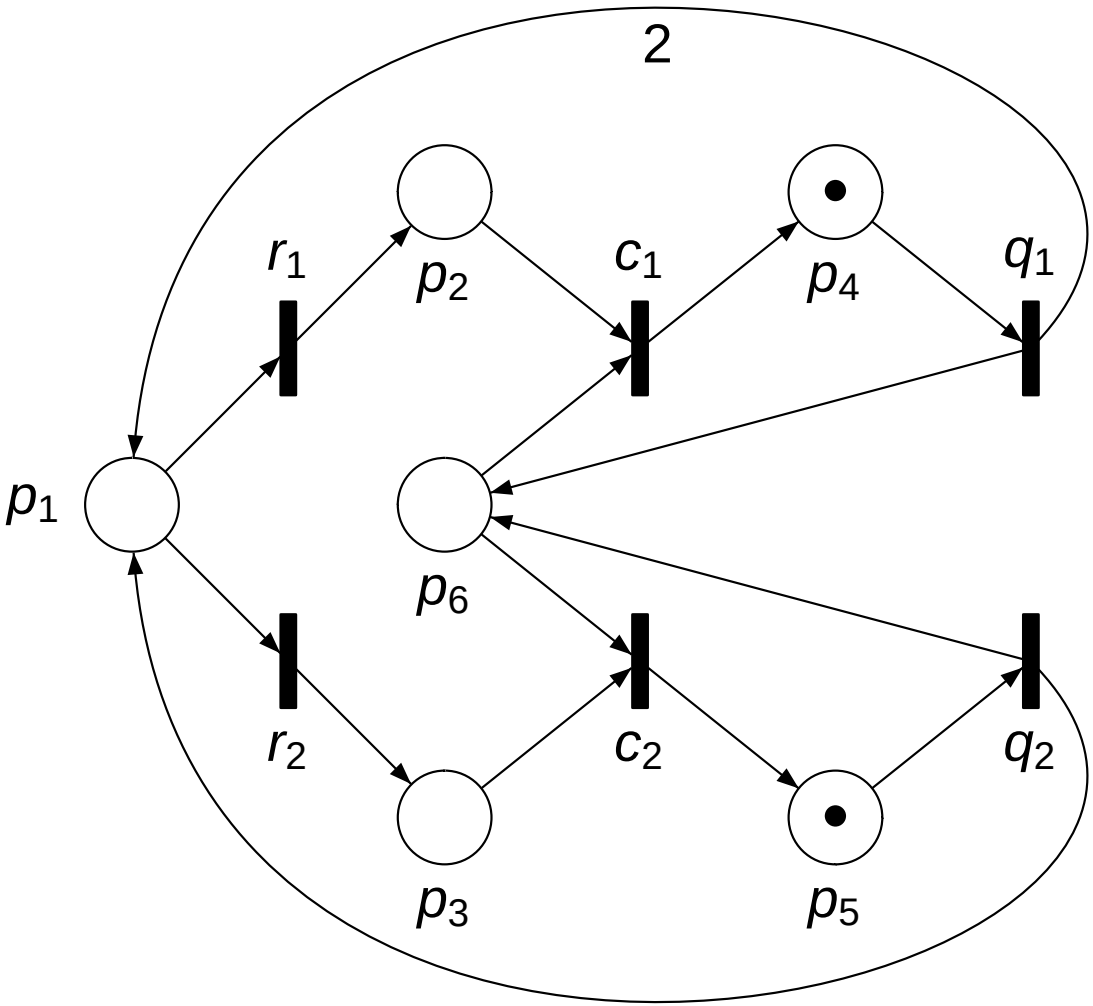
Config. initiale

$(1, 0, 0, 0, 0, 1)$

Config. erreur

$(0, 0, 0, 1, 1, 0)$

Analyse en arrière sur l'exemple



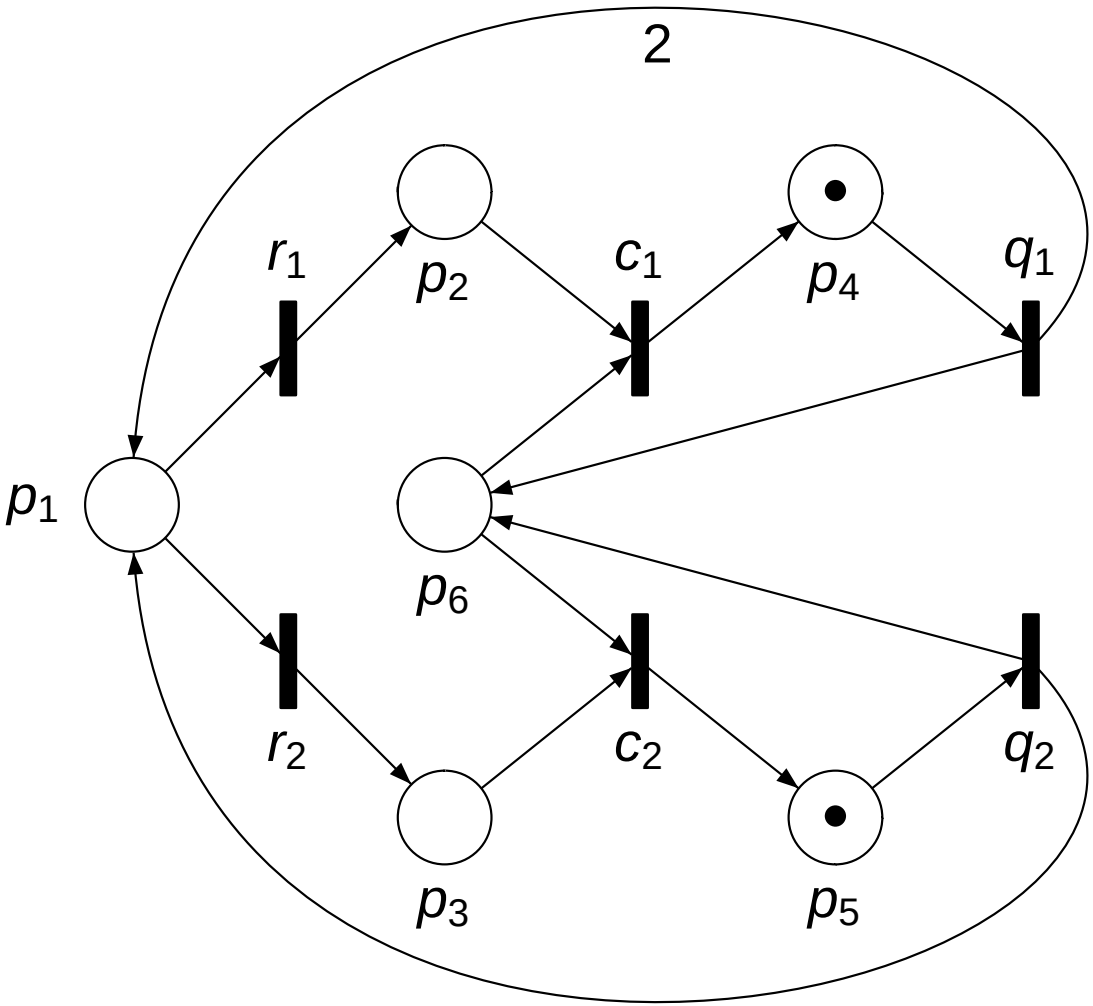
Config. initiale

$(1, 0, 0, 0, 0, 1)$

Config. erreur

$(0, 0, 0, 1, 1, 0)$

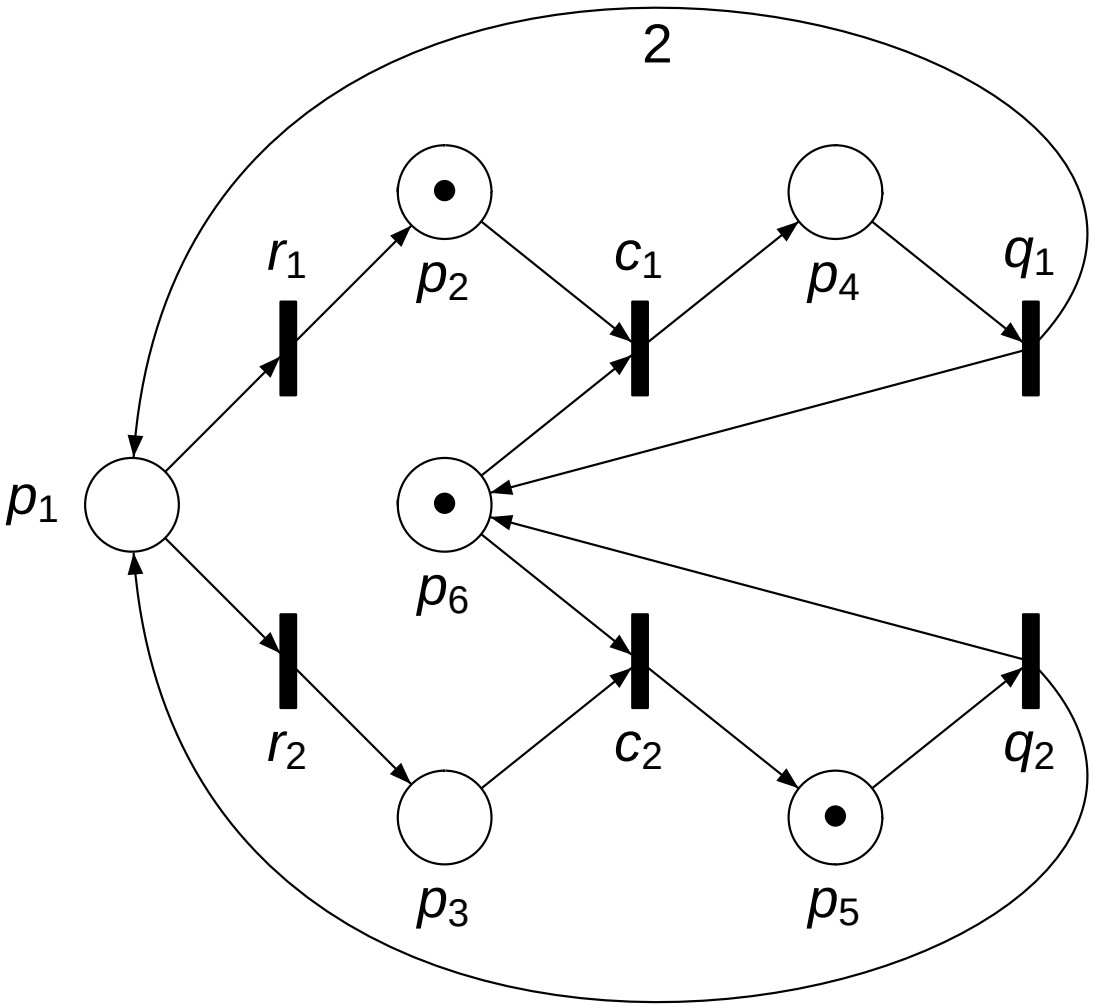
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

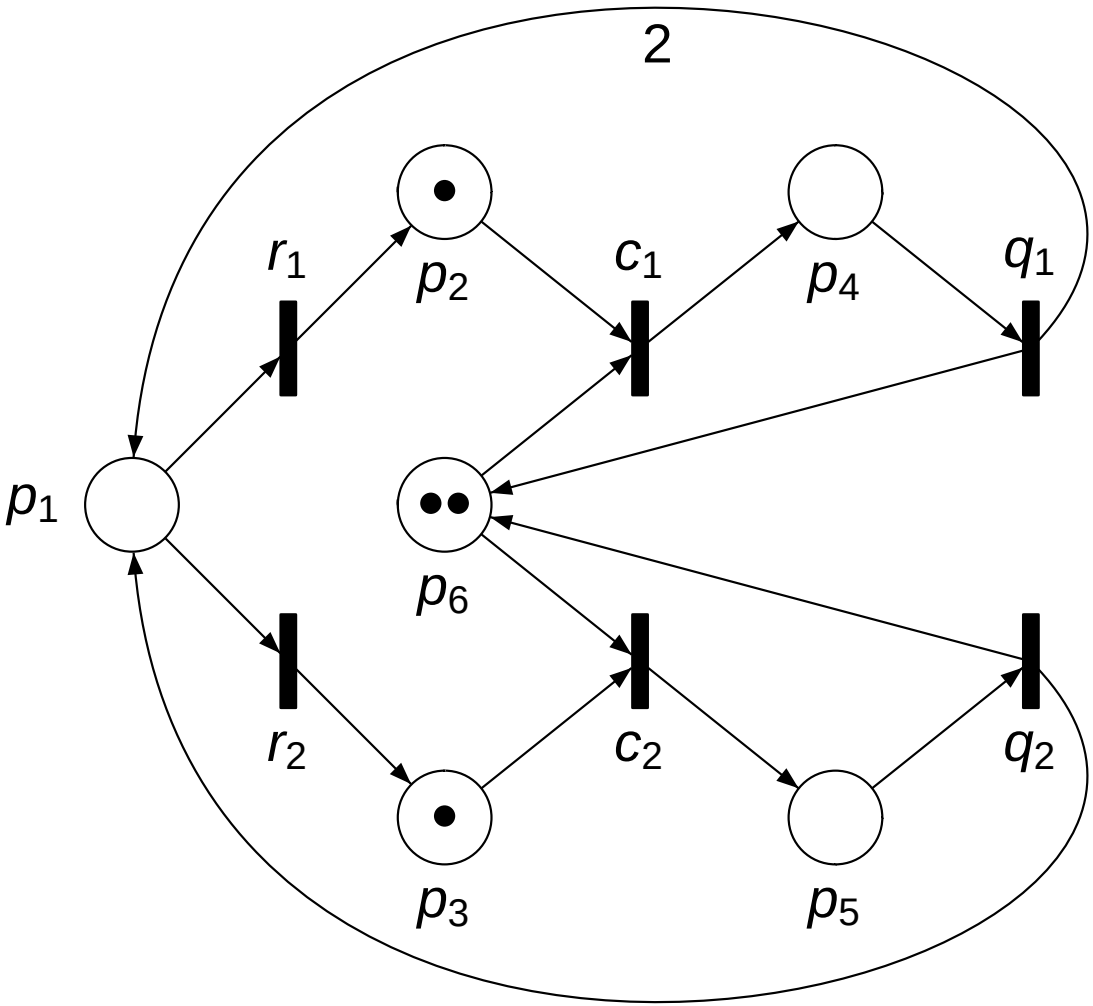
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

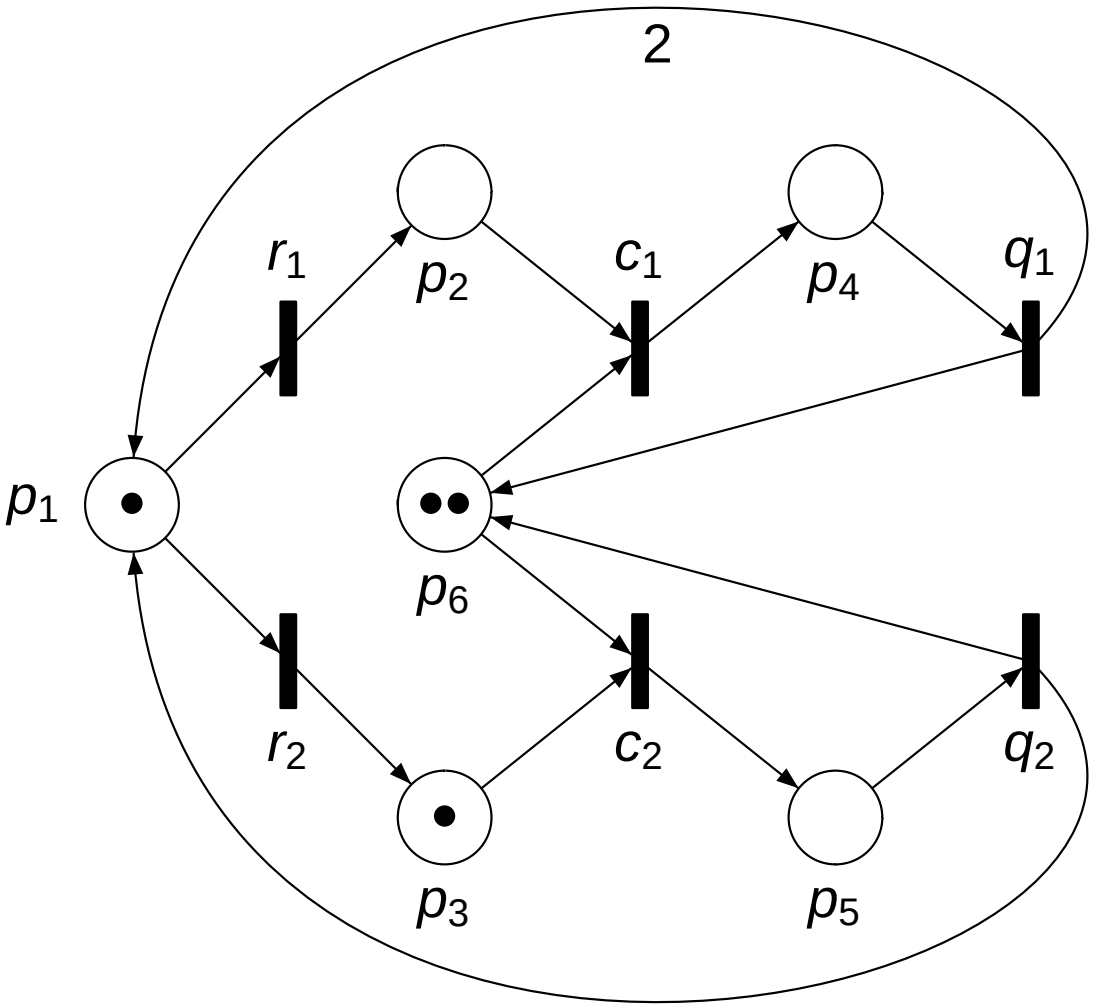
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

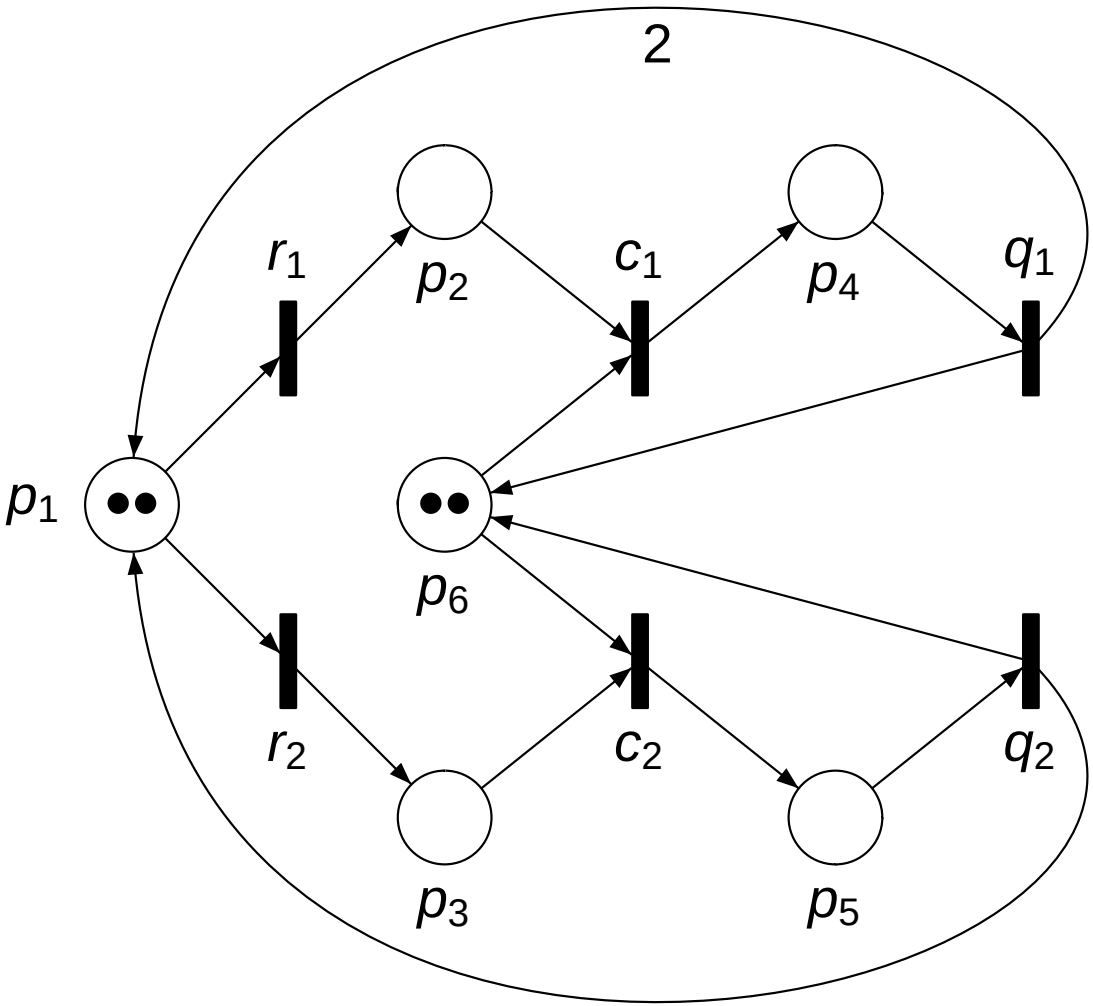
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

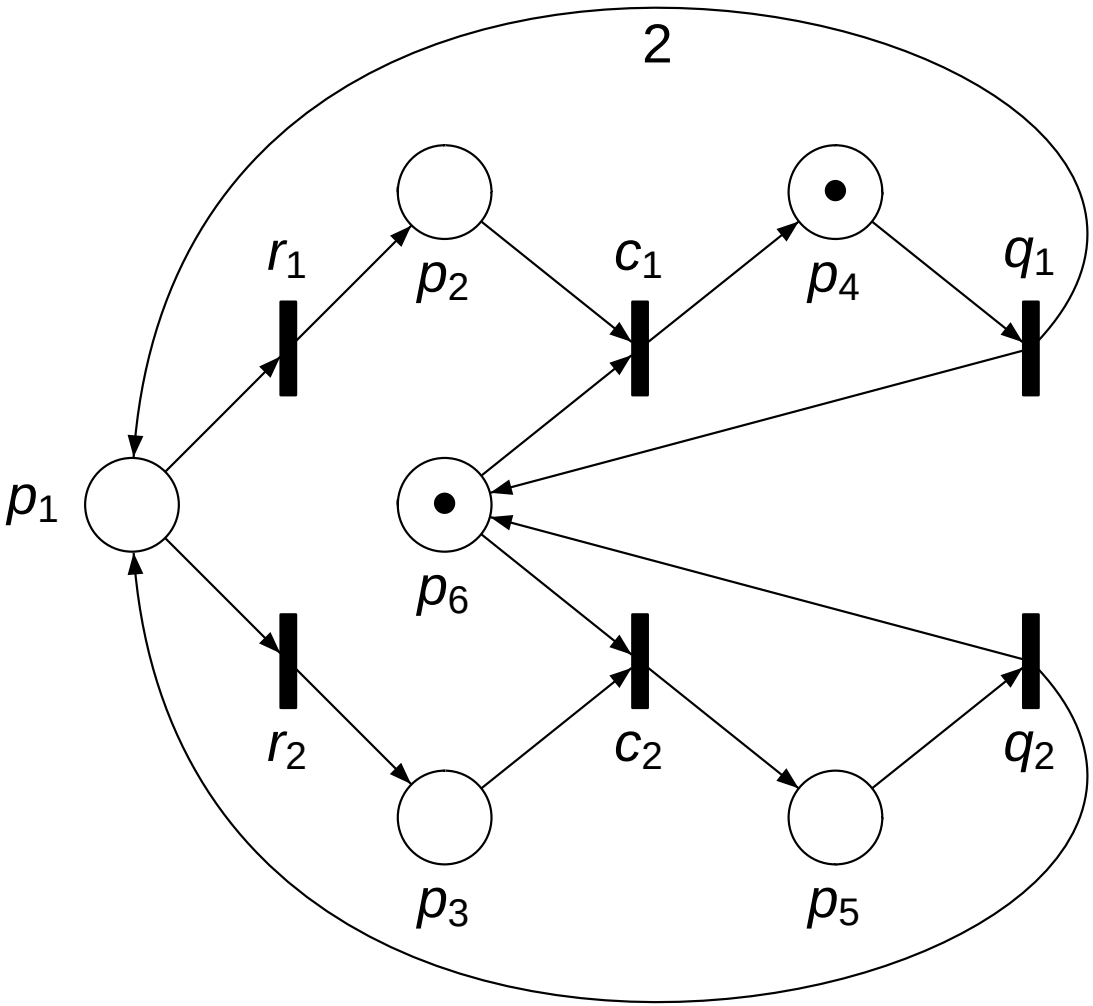
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

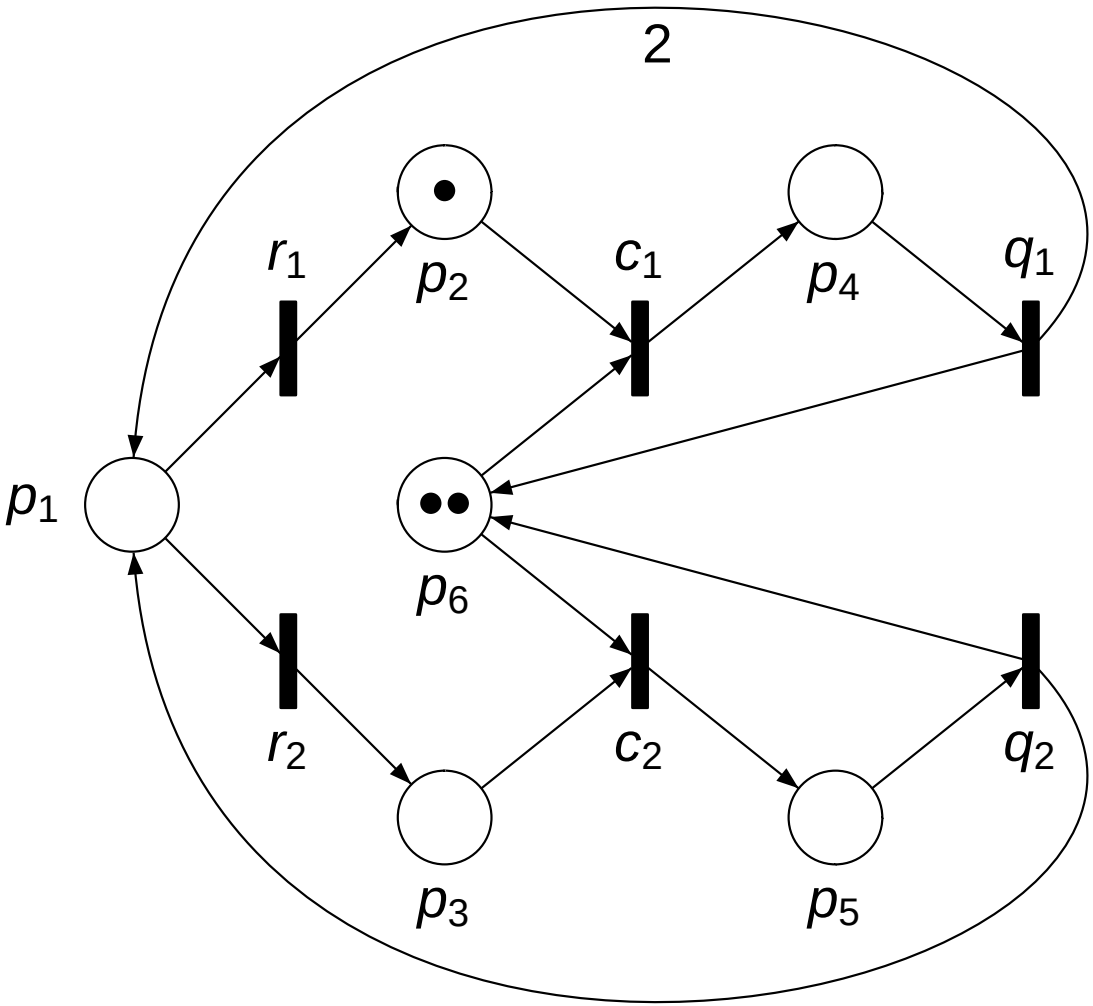
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

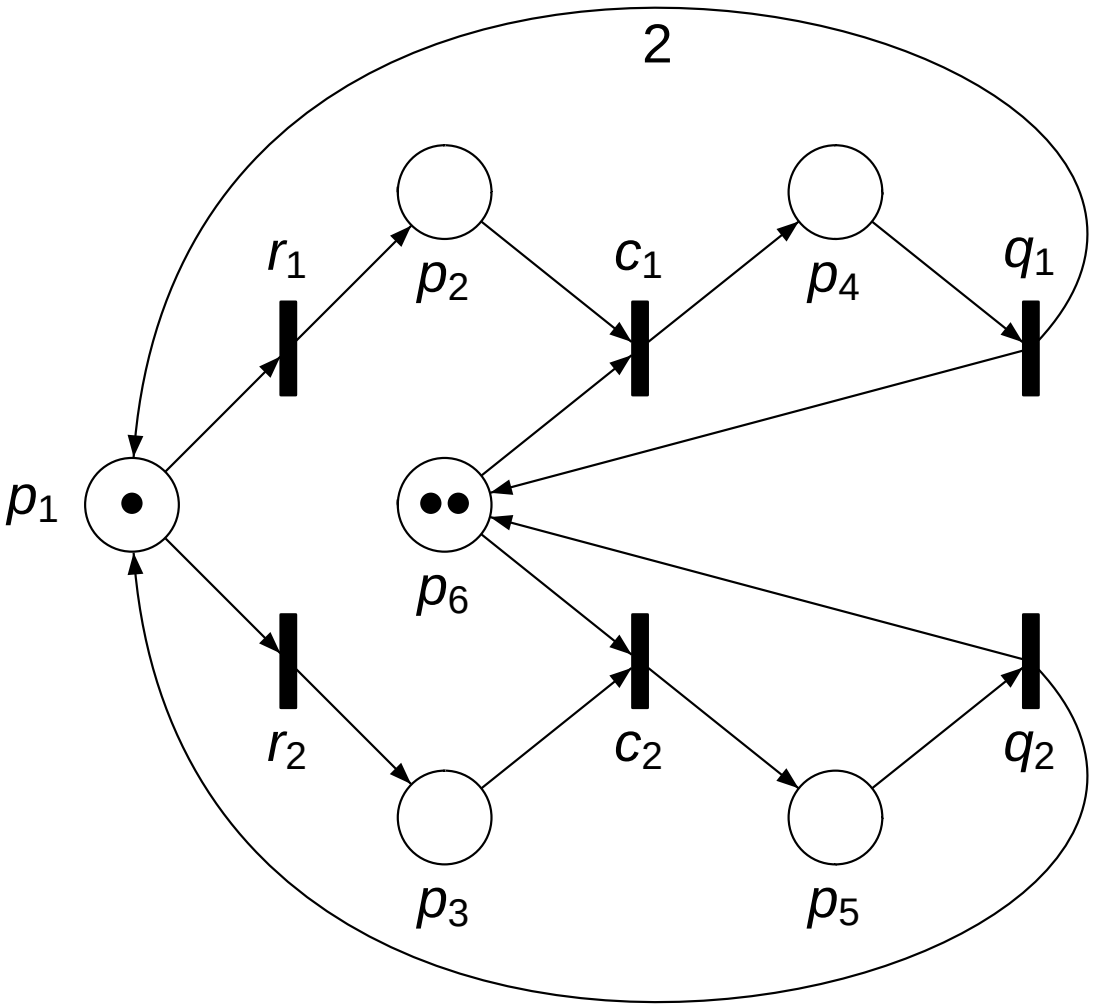
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

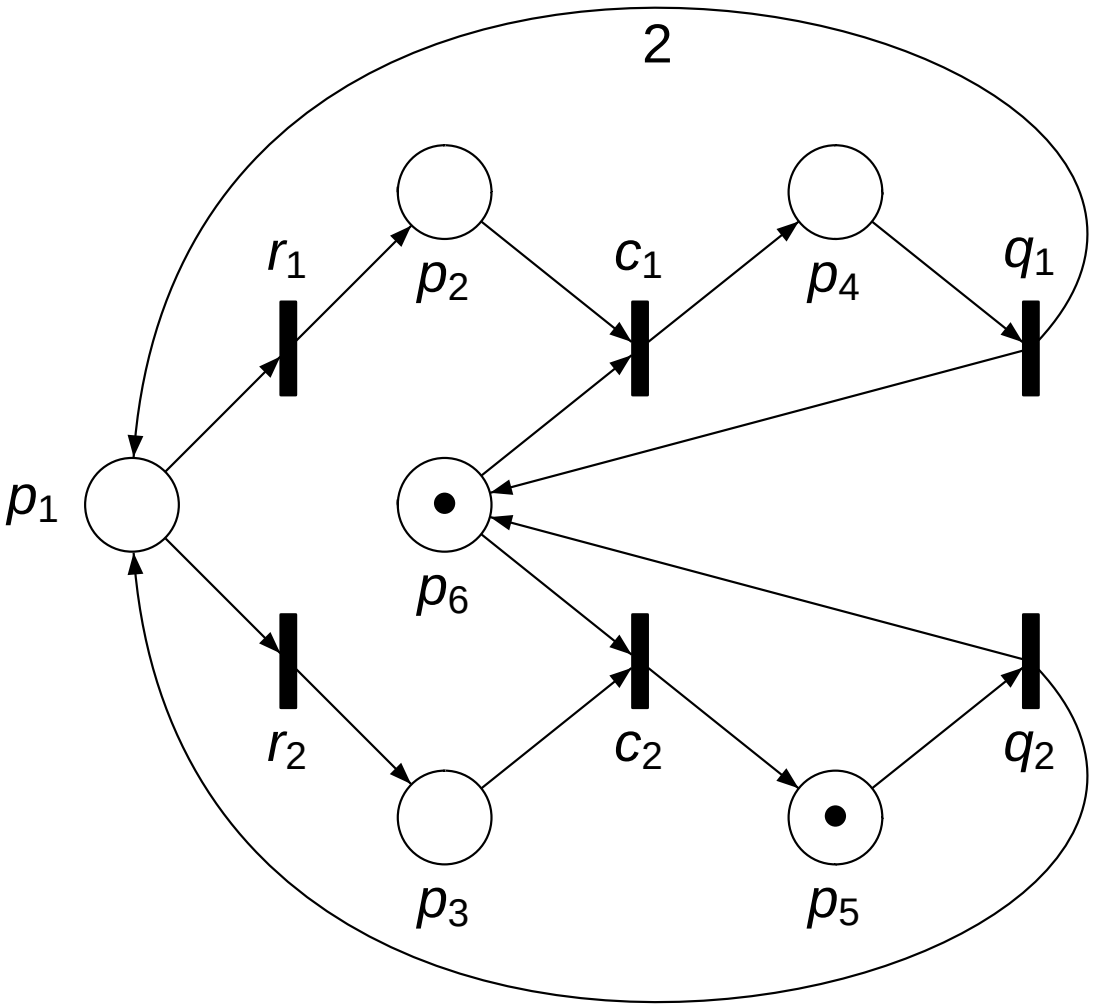
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

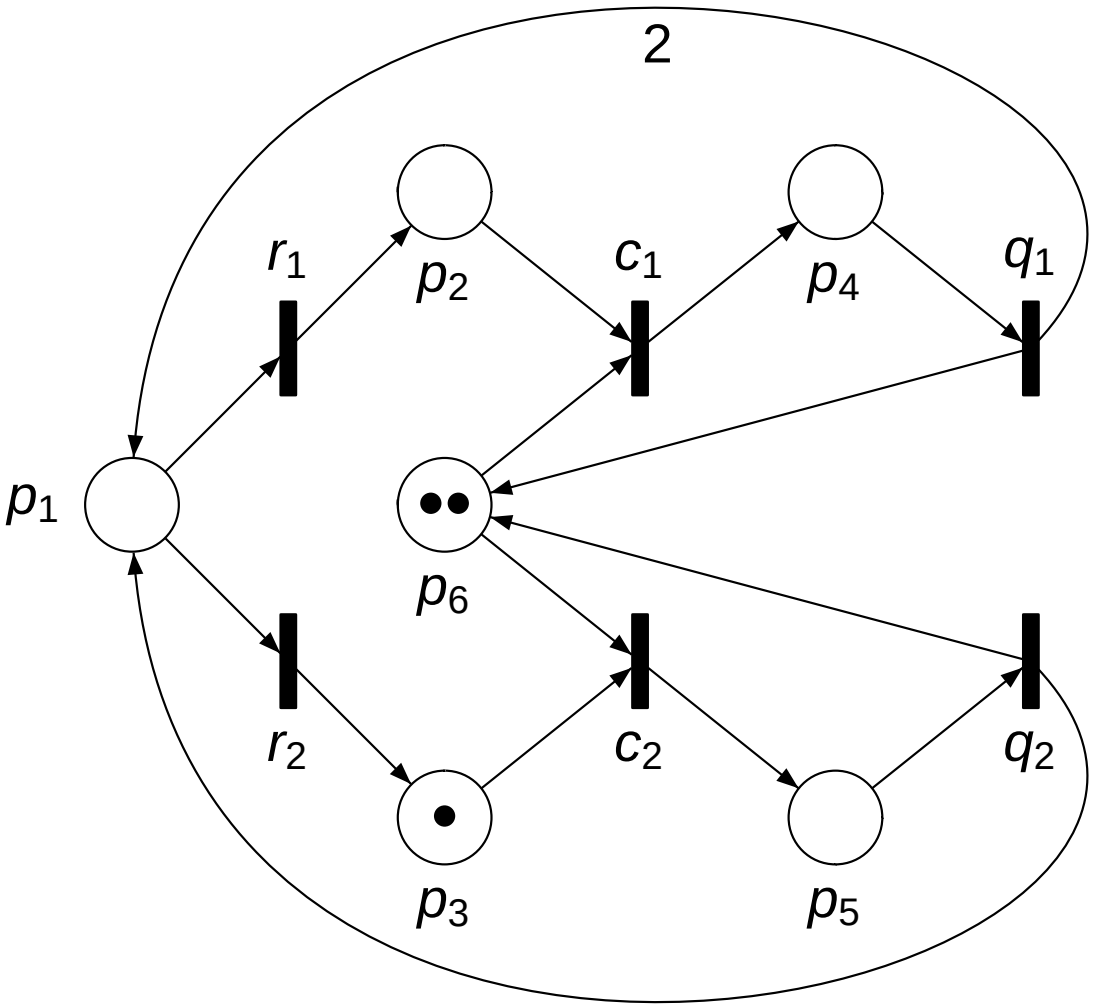
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

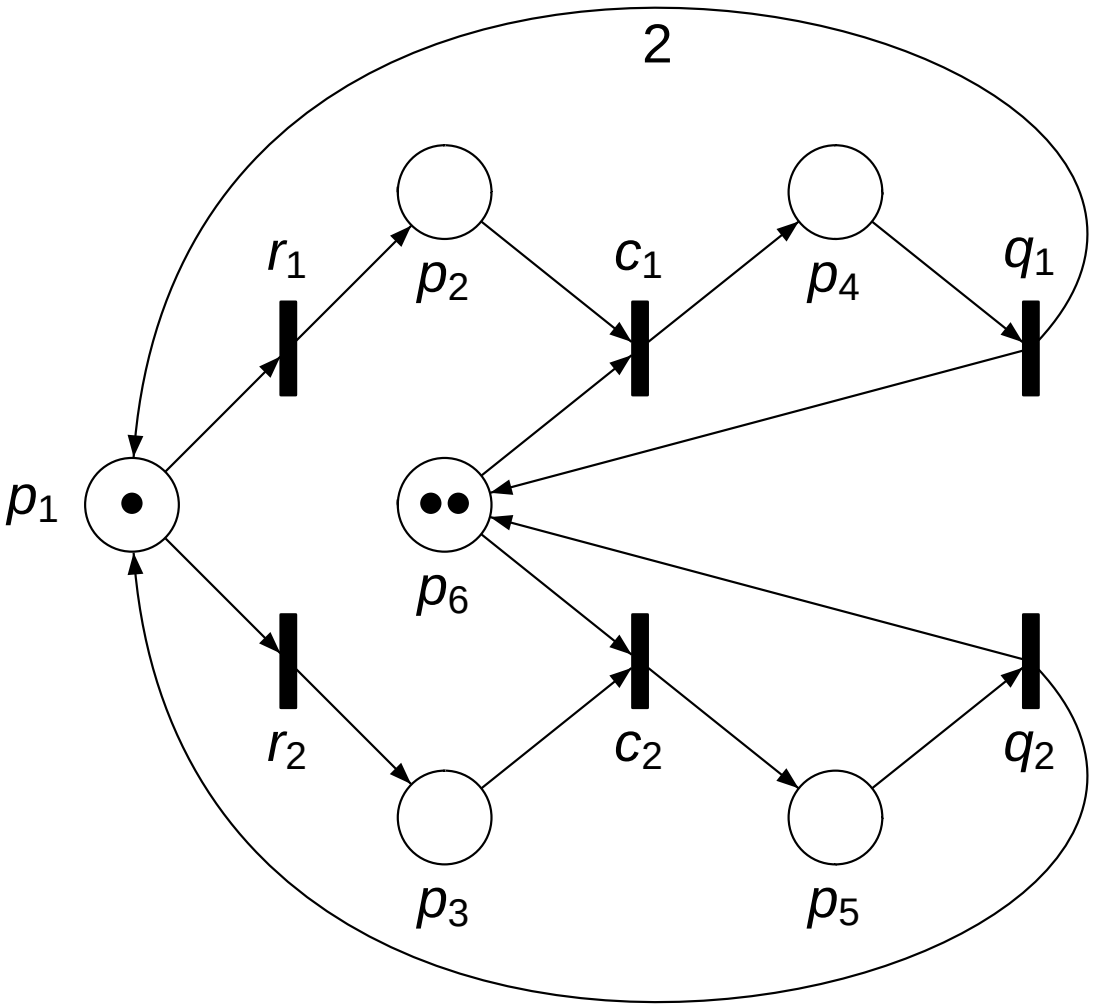
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

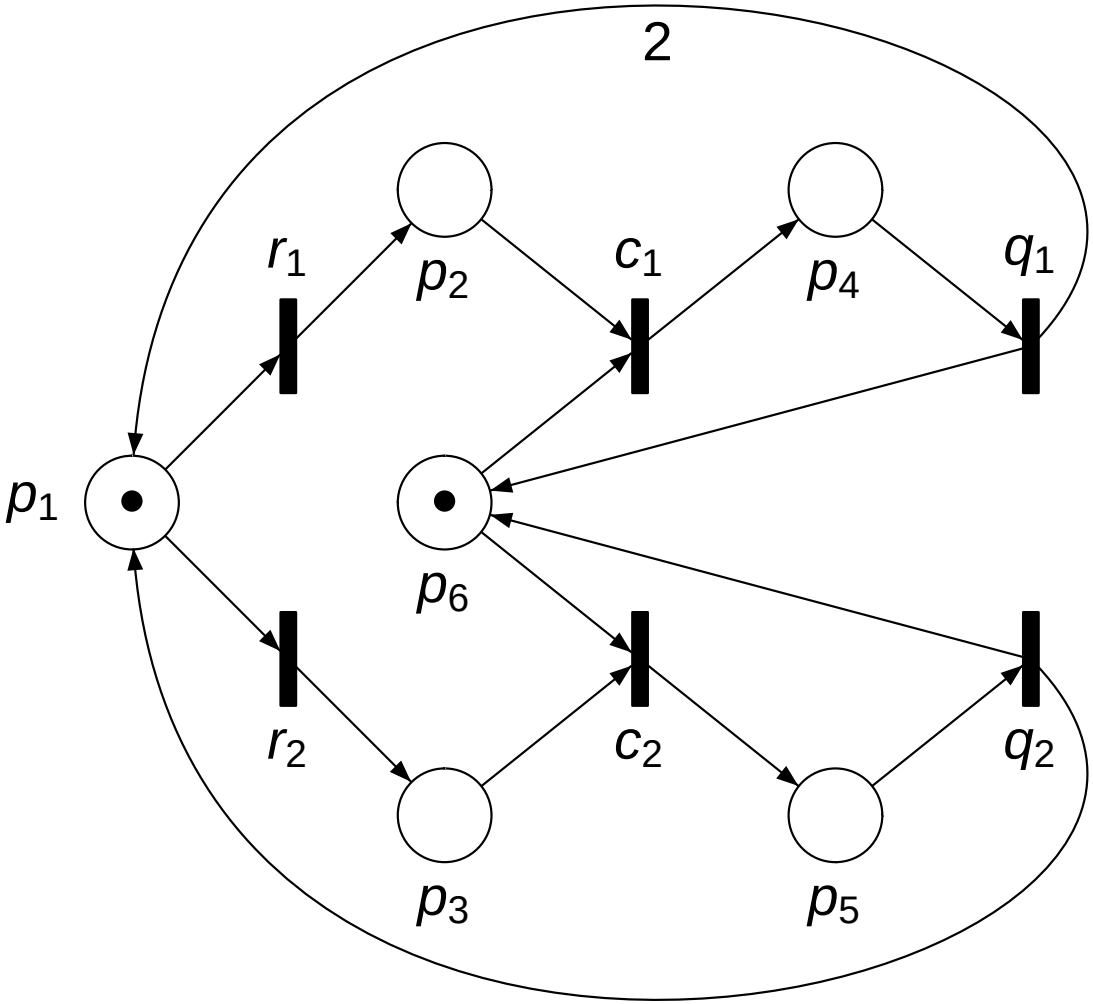
Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 1, 0, 0, 1, 1)$
- $(0, 1, 1, 0, 0, 2)$
- $(1, 0, 1, 0, 0, 2)$
- $(2, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

Analyse en arrière sur l'exemple



Config. minimales

- $(0, 0, 0, 1, 1, 0)$
- $(0, 0, 0, 1, 0, 1)$
- $(0, 1, 0, 0, 0, 2)$
- $(1, 0, 0, 0, 0, 2)$
- $(0, 0, 0, 0, 1, 1)$
- $(0, 0, 1, 0, 0, 2)$

Résultat

$(0, 0, 0, 1, 1, 0)$ n'est pas couvert

Analyse en avant pour le caractère borné

Problème du Caractère borné

Etant donné un réseau $\mathcal{N}$ et $s_0 \in \mathbb{N}^P$, $post^*(s_0)$ est-il fini ?

Idée (Karp-Miller)

- on construit un arbre d'accessibilité classique
- on tronque les noeuds $n' : s'$ ayant un ancêtre $n : s$ tels que s et s' sont comparables
- l'arbre obtenu est fini par le lemme de Dickson
- le réseau $\mathcal{N}$ est non borné ssi il existe $n' : s'$ ayant un ancêtre $n : s$ avec $s < s'$

Analyse en avant pour le caractère borné

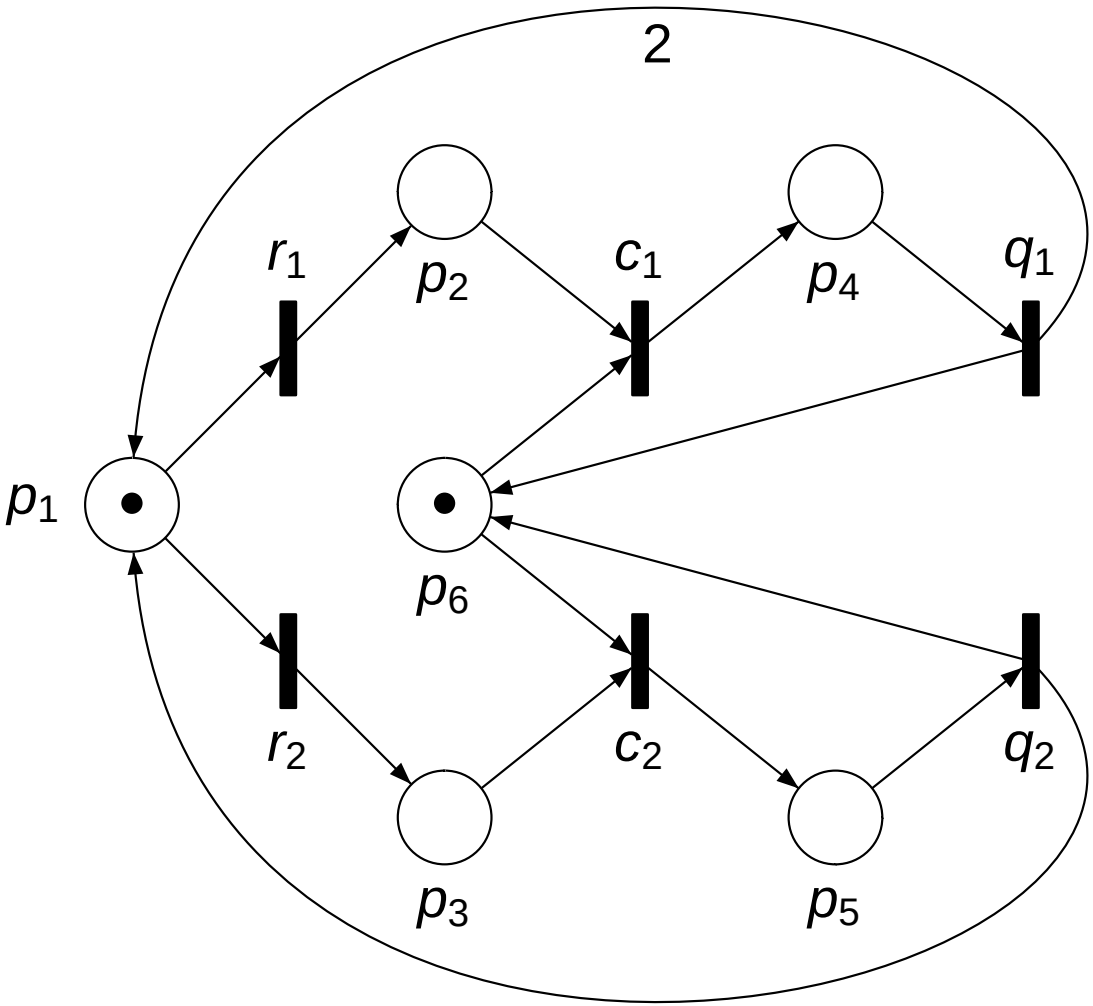
Problème du Caractère borné

Etant donné un réseau $\mathcal{N}$ et $s_0 \in \mathbb{N}^P$, $post^*(s_0)$ est-il fini ?

Idée (Karp-Miller)

- on construit un arbre d'accessibilité classique
- on tronque les noeuds $n' : s'$ ayant un ancêtre $n : s$ tels que s et s' sont comparables
- l'arbre obtenu est fini par le lemme de Dickson
- le réseau $\mathcal{N}$ est non borné ssi il existe $n' : s'$ ayant un ancêtre $n : s$ avec $s < s'$

Analyse en avant sur l'exemple



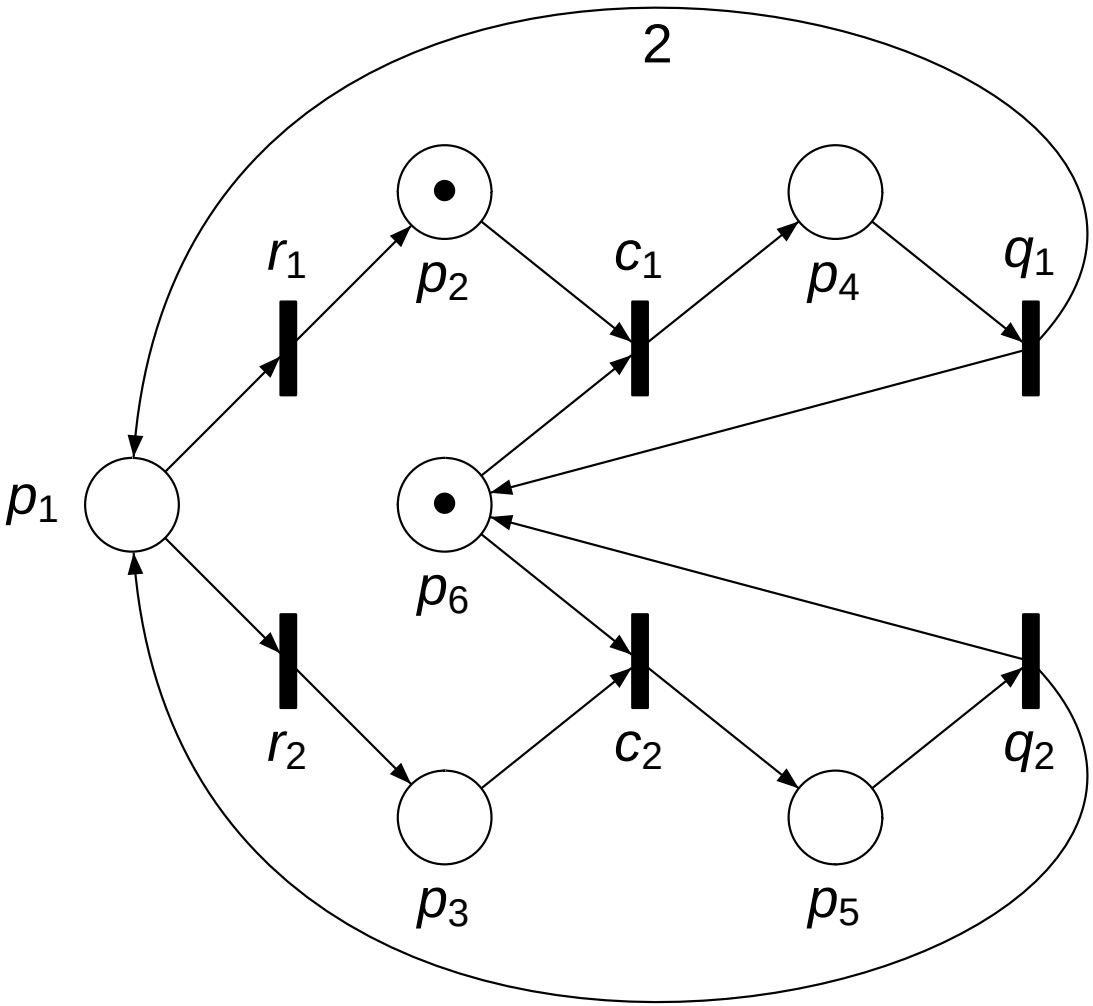
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



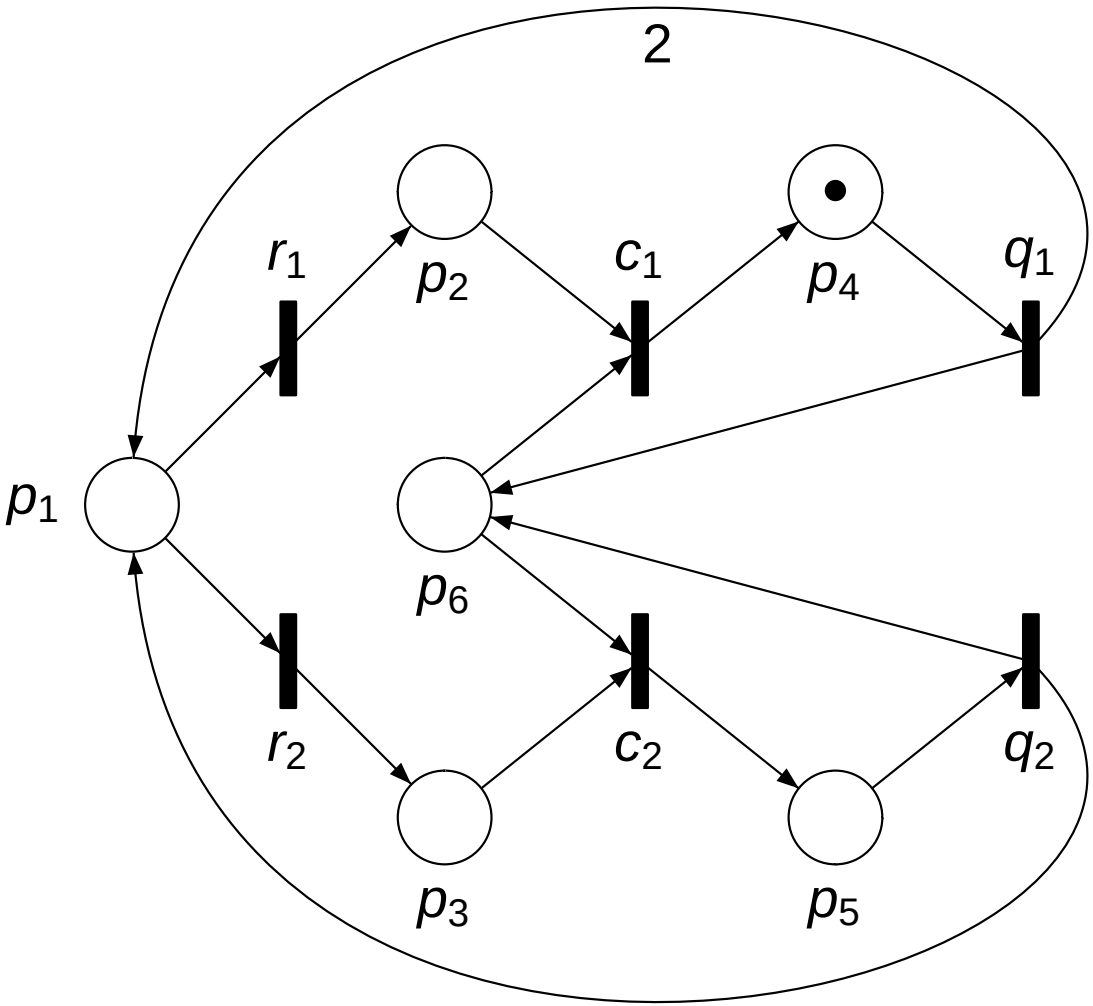
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



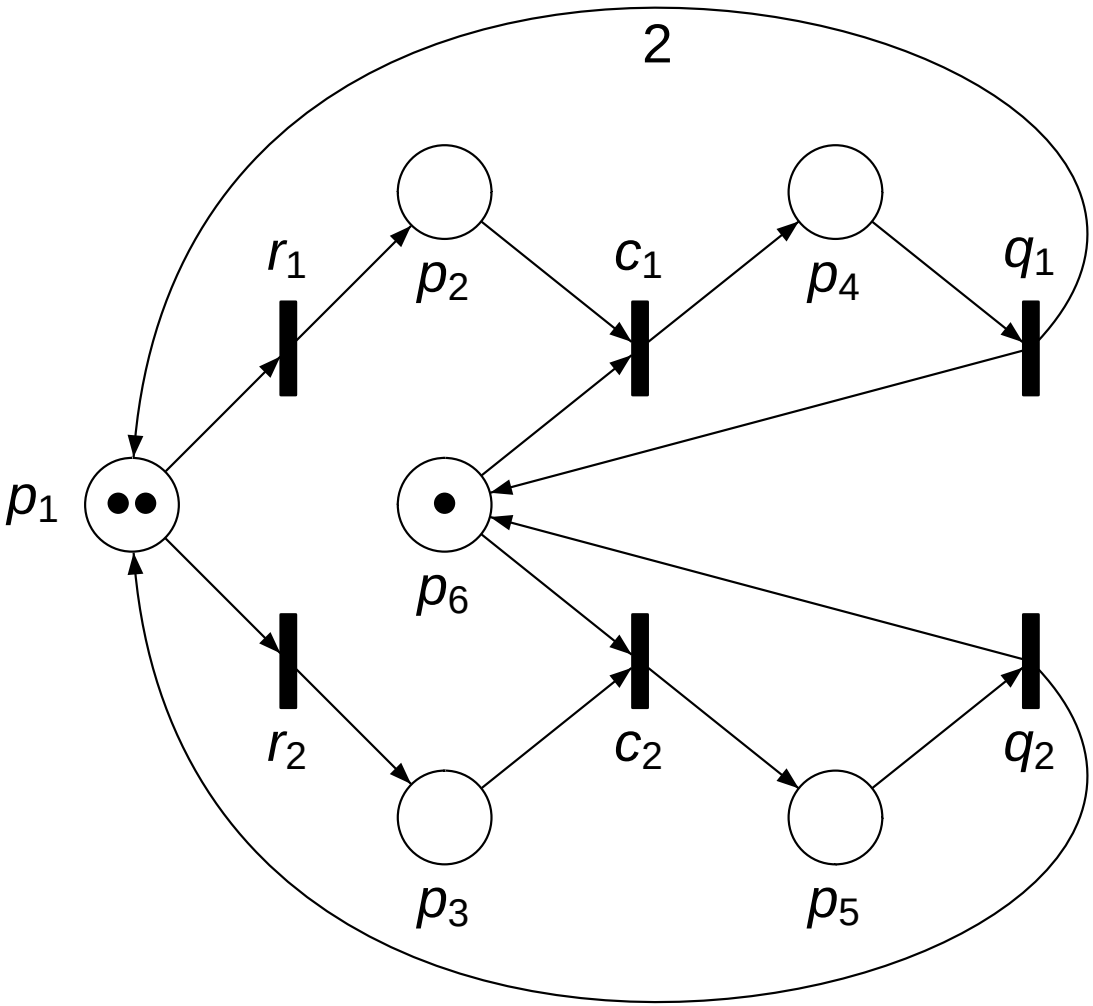
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



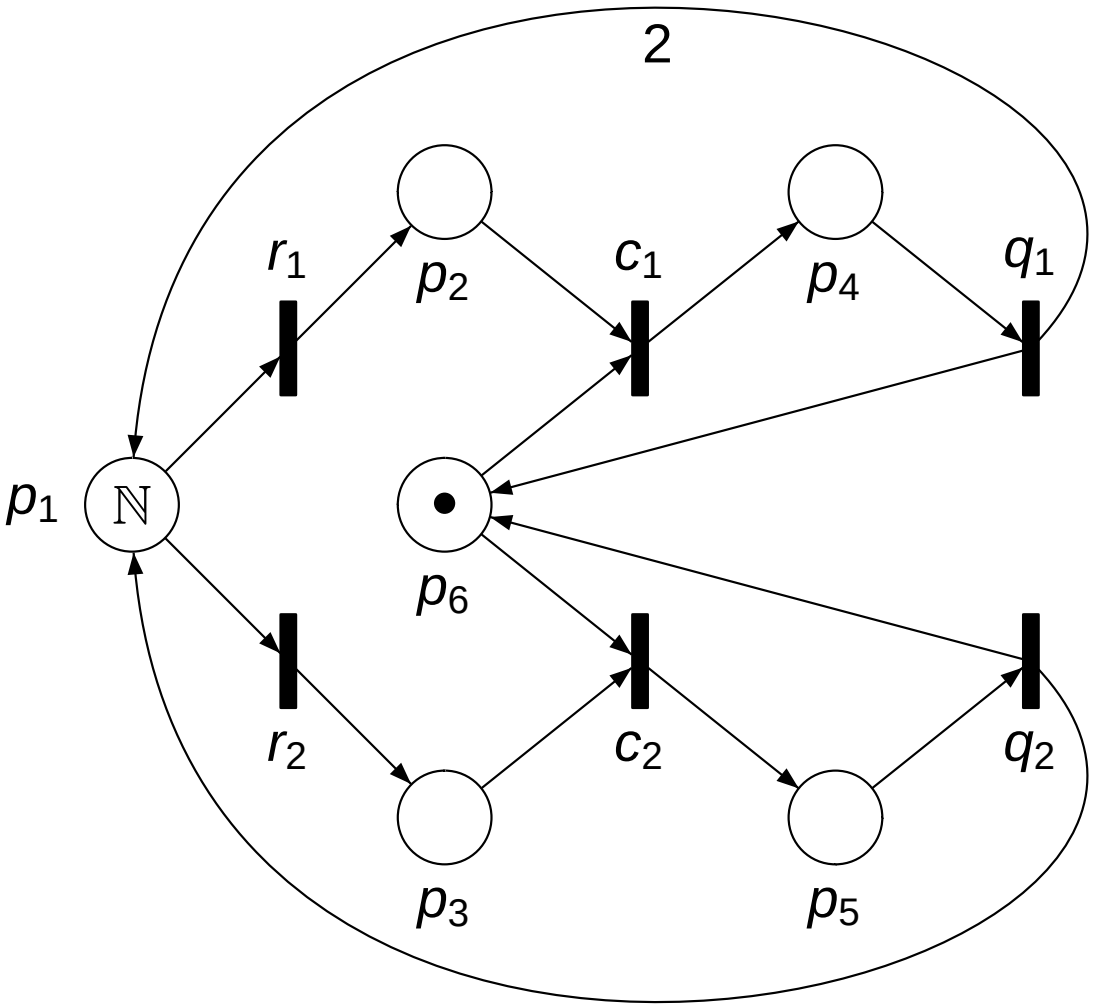
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



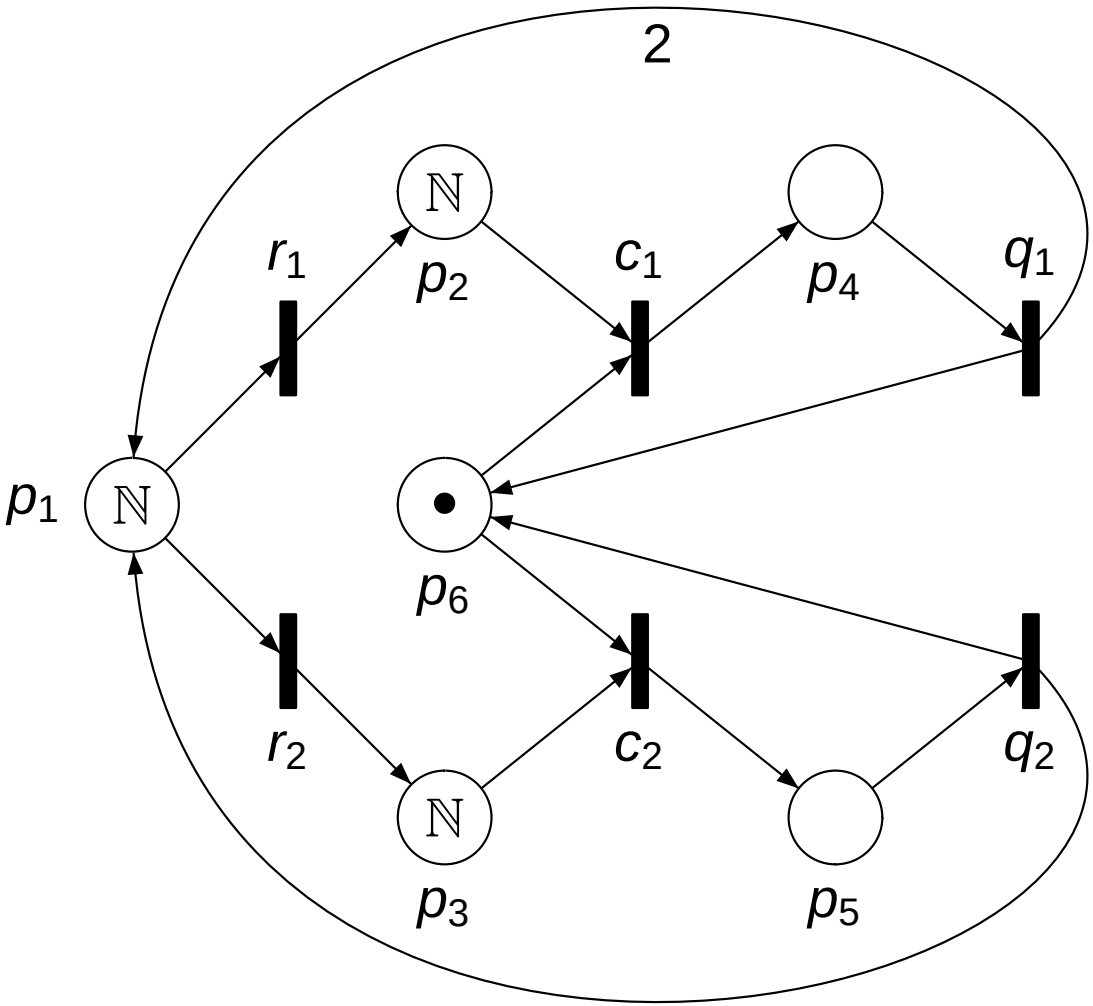
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



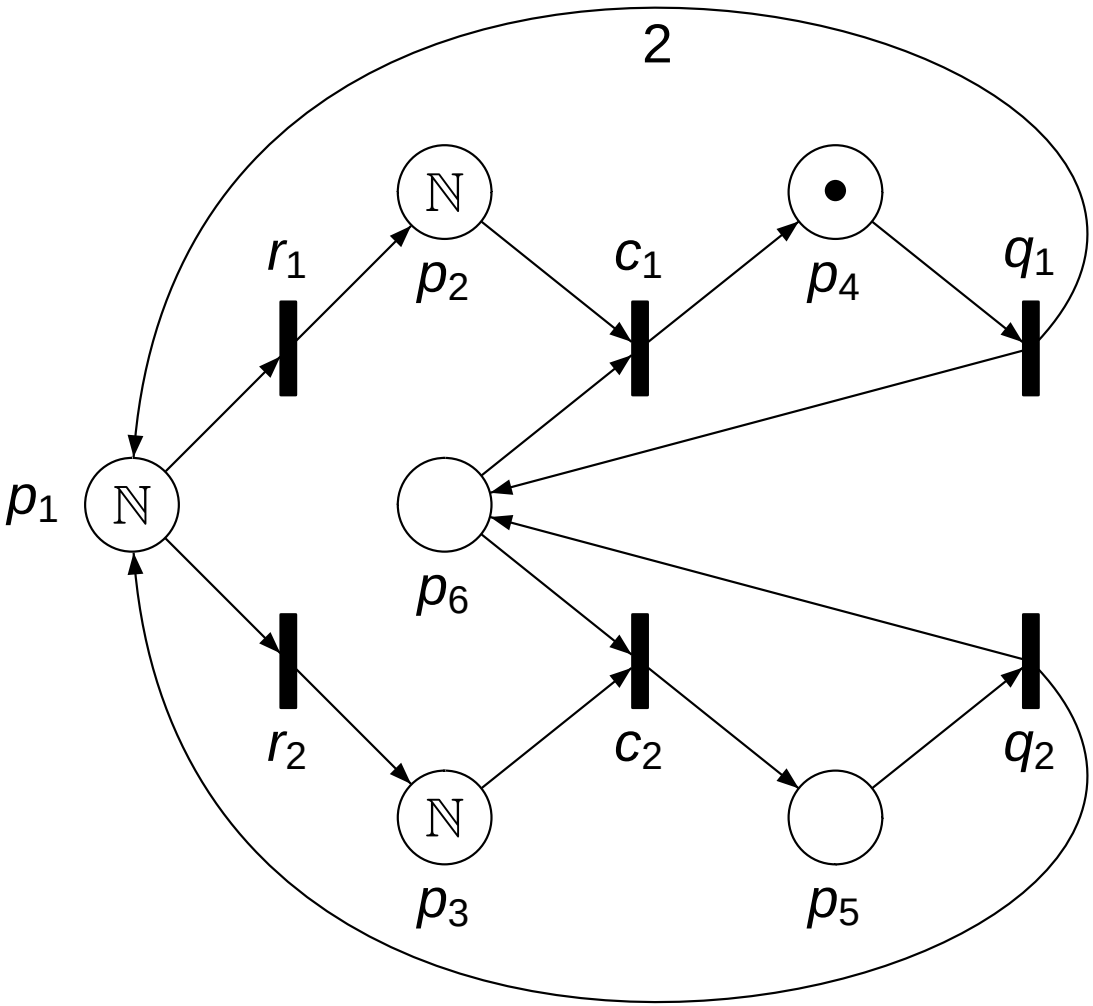
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



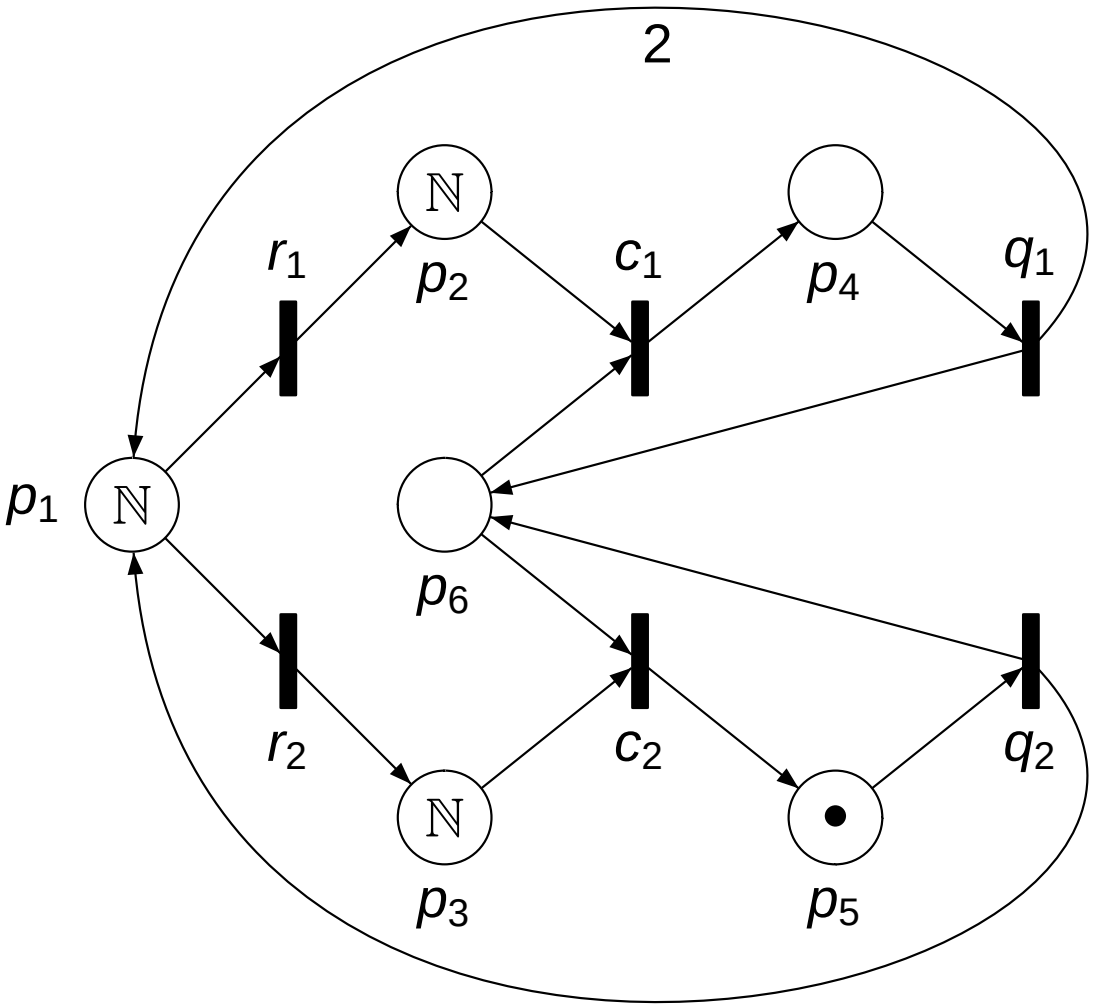
Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Analyse en avant sur l'exemple



Branche

- $(1, 0, 0, 0, 0, 1)$
- $(0, 1, 0, 0, 0, 1)$
- $(0, 0, 0, 1, 0, 0)$
- $(2, 0, 0, 0, 0, 1)$

Résultat

Réseau non borné

Plan

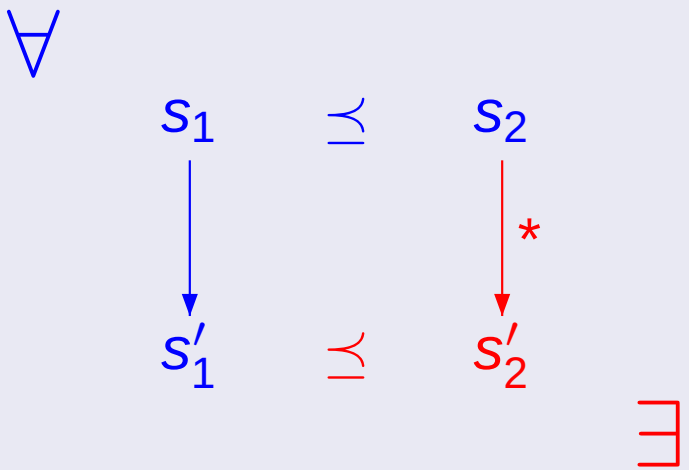
Définition des systèmes de transitions bien structurés

Beau pré-ordre

Un pré-ordre $\preceq$ sur S est un *beau pré-ordre* si de toute suite infinie $(s_i)_{i\in\mathbb{N}}$ on peut extraire une sous-suite croissante.

Compatibilité

Un pré-ordre $\preceq$ est compatible pour un S.T. $(S, \Sigma, \rightarrow)$ si :



Définition des systèmes de transitions bien structurés

Beau pré-ordre

Un pré-ordre $\preceq$ sur S est un *beau pré-ordre* si de toute suite infinie $(s_i)_{i\in\mathbb{N}}$ on peut extraire une sous-suite croissante.

Compatibilité

Un pré-ordre $\preceq$ est compatible pour un S.T. $(S, \Sigma, \rightarrow)$ si : pour tout $s_2 \succeq s_1 \rightarrow s_1'$, il existe s_2' tel que $s_2 \stackrel{*}{\rightarrow} s_2' \succeq s_1$.

Définition (Finkel 90, Abdulla *et al.* 96)

Un *système de transitions bien structuré* est un S.T. $(S, \Sigma, \rightarrow, \preceq)$ muni d'un beau pré-ordre compatible $\preceq$ sur S.

Exemples de systèmes de transitions bien structurés

- Réseaux de Petri étendu (reset, transfert)
- Automates communicant par canaux FIFO avec perte de messages
- Automates temporisés
- Et beaucoup d'autres classes de systèmes infinis

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un S.T. bien structuré $\mathcal{S}$ et $s_0, s \in S$, existe-t-il $s' \succeq s$ atteignable depuis s_0 ?

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un S.T. bien structuré $\mathcal{S}$ et $s_0, s \in S$, existe-t-il $s' \succeq s$ atteignable depuis s_0 ?

Calcul de $pre^*(\uparrow s)$

$$\begin{cases} X_0 &= \uparrow \{s\} \\ X_{k+1} &= X_k \bigcup \uparrow pre(X_k) \end{cases}$$

- (X_k) est une suite croissante (pour $\subseteq$) d'ensembles clos par le haut
- (X_k) est donc stationnaire (car $\preceq$ b.p.o.)
- $pre^*(\uparrow s) = \bigcup_{k \in \mathbb{N}} X_k$ est donc calculable

Analyse en arrière pour la couverture

Problème de la Couverture

Etant donné un S.T. bien structuré $\mathcal{S}$ et $s_0, s \in S$, existe-t-il $s' \succeq s$ atteignable depuis s_0 ?

Théorème (Abdulla *et al.* 96)

Une base finie de $pre^*(\uparrow s)$ *est calculable pour les S.T. bien structurés.*

Corollaire

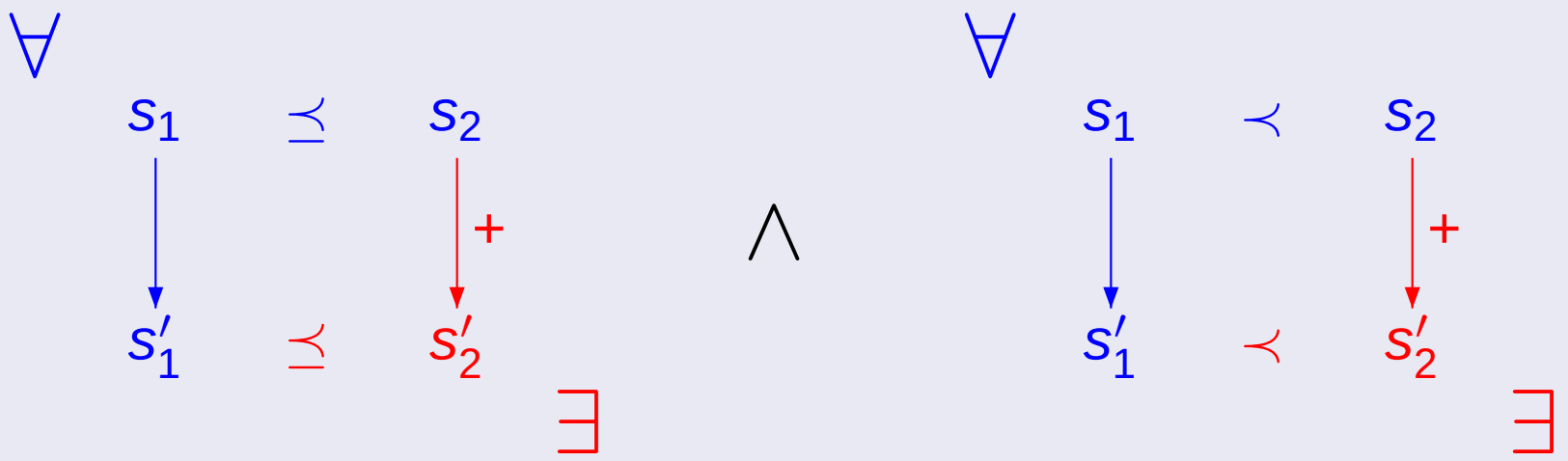
Le problème de la couverture est décidable pour les S.T. bien structurés.

Analyse en avant pour le caractère borné

Problème du caractère borné

Etant donné un S.T. bien structuré et $s_0 \in S$, $post^*(s_0)$ est-il fini ?

Compatibilité transitive stricte



Analyse en avant pour le caractère borné

Problème du caractère borné

Etant donné un S.T. bien structuré et $s_0 \in S$, $post^*(s_0)$ est-il fini ?

Compatibilité transitive stricte

Compatibilité transitive sur $\preceq$ et sur $\prec$

Théorème (Finkel 90)

Le caractère borné est décidable pour les S.T. bien structurés à compatibilité transitive stricte et dont le b.p.o $\preceq$ est un ordre partiel.

Démonstration.

Arbre de Karp-Miller étendu aux S.T. bien structurés. □

Plan

Systèmes à compteurs : syntaxe

Fonction affine

Fonction $f : \mathbb{Z}^n \to \mathbb{Z}^n$ donnée par $f(x) = M.x + v$ où $M \in \mathbb{Z}^{n \times n}$ et $v \in \mathbb{Z}^n$.

Polyèdre (convexe)

Sous-ensemble de $\mathbb{Z}^n$ défini par un système d'inéquations $M.x \leq v$ où $M \in \mathbb{Z}^{m \times n}$ et $v \in \mathbb{Z}^n$.

Syntaxe

Un système à compteur $\mathcal{C}$ est un triplet (Σ, G, F) où :

- Σ : ensemble des actions
- G : fonction de Σ dans l'ens. des polyèdres
- F : fonction de Σ dans l'ens. des fonctions affines

Systèmes à compteurs : syntaxe

Fonction affine

Fonction $f : \mathbb{Z}^n \to \mathbb{Z}^n$ donnée par $f(x) = M.x + v$ où $M \in \mathbb{Z}^{n \times n}$ et $v \in \mathbb{Z}^n$.

Polyèdre (convexe)

Sous-ensemble de $\mathbb{Z}^n$ défini par un système d'inéquations $M.x \leq v$ où $M \in \mathbb{Z}^{m \times n}$ et $v \in \mathbb{Z}^n$.

Syntaxe

Un système à compteur $\mathcal{C}$ est un triplet (Σ, G, F) où :

- Σ : ensemble des actions
- G : fonction de Σ dans l'ens. des polyèdres
- F : fonction de Σ dans l'ens. des fonctions affines

Systèmes à compteurs : sémantique

Syntaxe

Un *système à compteurs* $\mathcal{C}$ est un triplet (Σ, G, F) où :

- Σ : ensemble des actions
- $G(a)$: polyèdre
- $F(a)$: fonction affine

Sémantique

Système de transitions $(S, \Sigma, \rightarrow)$ où $S = \mathbb{Z}^n$ et :

$$s \xrightarrow{a} s' \text{ ssi } s \models G(a) \text{ et } s' = f(s)$$

Exemple

- Réseaux de Petri, réseaux de Petri étendus (test-à-zéro)
- Programmes manipulant des variables entières

Systèmes à compteurs : sémantique

Syntaxe

Un *système à compteurs* $\mathcal{C}$ est un triplet (Σ, G, F) où :

- Σ : ensemble des actions
- $G(a)$: polyèdre
- $F(a)$: fonction affine

Sémantique

Système de transitions $(S, \Sigma, \rightarrow)$ où $S = \mathbb{Z}^n$ et :

$$s \xrightarrow{a} s' \text{ ssi } s \models G(a) \text{ et } s' = f(s)$$

Exemple

- Réseaux de Petri, réseaux de Petri étendus (test-à-zéro)
- Programmes manipulant des variables entières

Systèmes à compteurs : sémantique

Syntaxe

Un *système à compteurs* $\mathcal{C}$ est un triplet (Σ, G, F) où :

- Σ : ensemble des actions
- $G(a)$: polyèdre
- $F(a)$: fonction affine

Sémantique

Système de transitions $(S, \Sigma, \rightarrow)$ où $S = \mathbb{Z}^n$ et :

$$s \xrightarrow{a} s' \text{ ssi } s \models G(a) \text{ et } s' = f(s)$$

Exemple

- Réseaux de Petri, réseaux de Petri étendus (test-à-zéro)
- Programmes manipulant des variables entières

Calcul itératif de $post^*$ (ou pre^*)

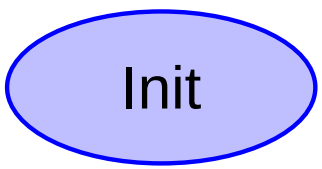
$post^*$ comme plus petit point fixe

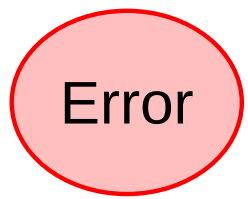
- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 & = & \{s_0\} \\ X_{k+1} & = & X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 = \{s_0\} \\ X_{k+1} = X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

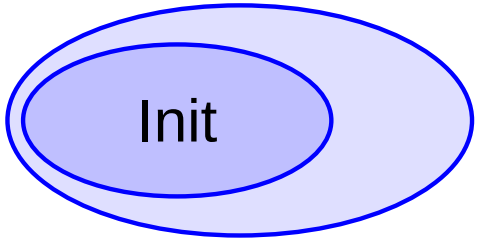


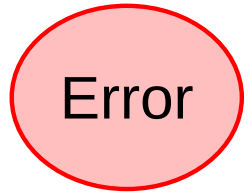


Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 = \{s_0\} \\ X_{k+1} = X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

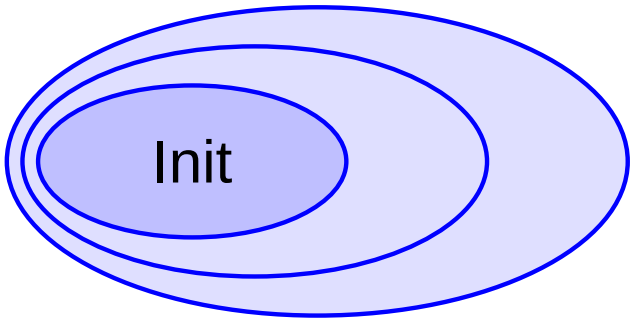


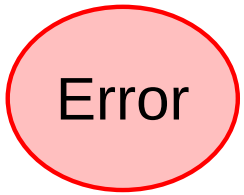


Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 & = & \{s_0\} \\ X_{k+1} & = & X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

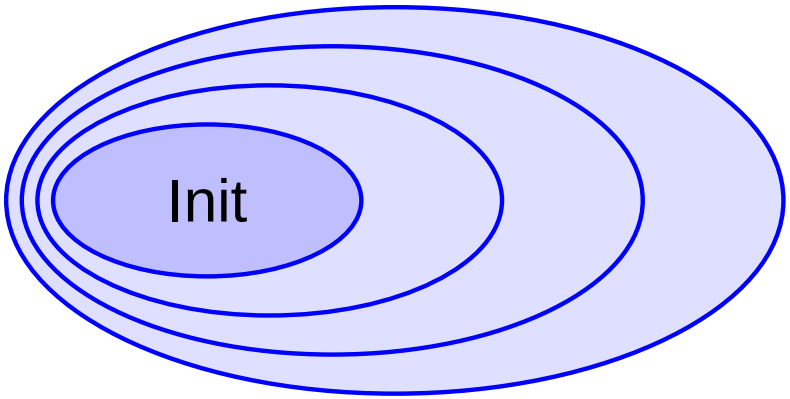


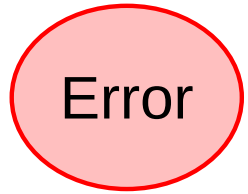


Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 & = & \{s_0\} \\ X_{k+1} & = & X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

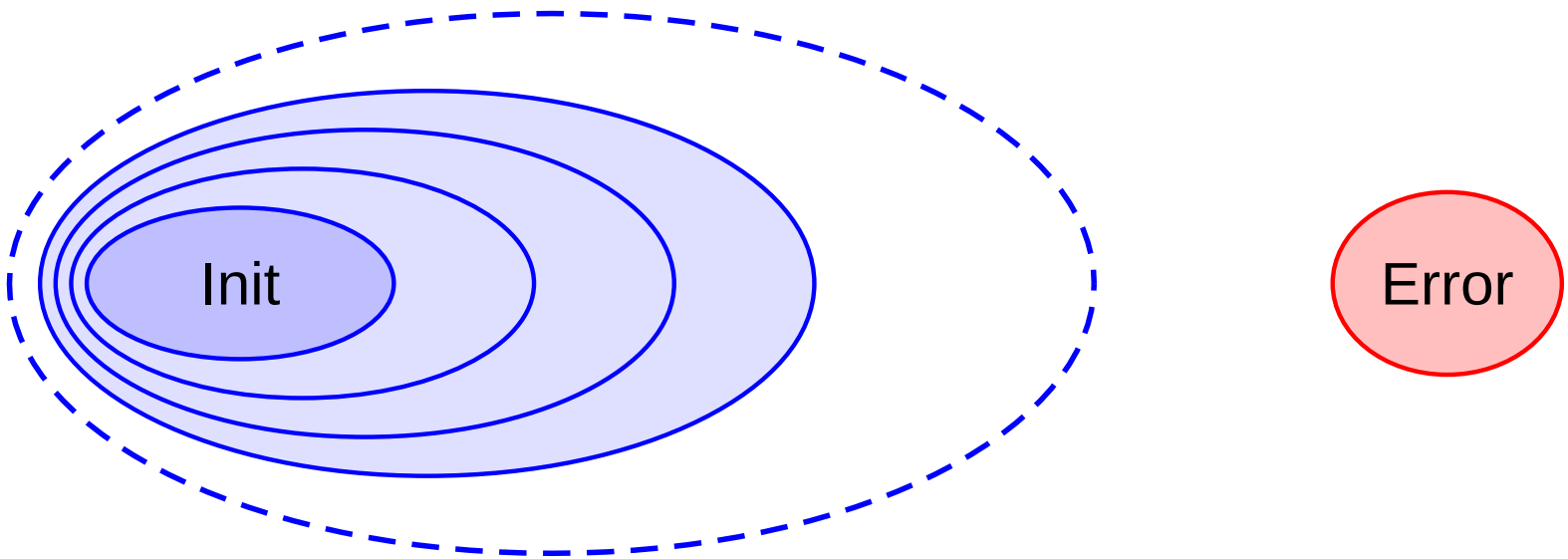




Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 & = & \{s_0\} \\ X_{k+1} & = & X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$



Calcul itératif de $post^*$ (ou pre^*)

$post^*$ comme plus petit point fixe

- $post^*(X_0) = \bigcup_{k \in \mathbb{N}} X_k$
- $\begin{cases} X_0 & = & \{s_0\} \\ X_{k+1} & = & X_k \bigcup \bigcup_{a \in \Sigma} post(X_k, a) \end{cases}$

Besoins

- représentation finie d'ensembles infinis
- clôture effective par $\cup$ et *post*
- = ou $\subseteq$ décidable

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] = \emptyset \\ [\![\top]\!] = S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] = [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] = [\![r]\!] \cap [\![r']\!] \end{cases}$
- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] & = & \emptyset \\ [\![\top]\!] & = & S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] & = & [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] & = & [\![r]\!] \cap [\![r']\!] \end{cases}$

- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] &= \emptyset \\ [\![\top]\!] &= S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] &= [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] &= [\![r]\!] \cap [\![r']\!] \end{cases}$
- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] = \emptyset \\ [\![\top]\!] = S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] = [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] = [\![r]\!] \cap [\![r']\!] \end{cases}$

- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] &= \emptyset \\ [\![\top]\!] &= S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] &= [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] &= [\![r]\!] \cap [\![r']\!] \end{cases}$

- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Structure de Régions

Définition

Une *algèbre de régions* (pour S) est un tuple $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ où :

- R est un ensemble de *régions*, d'*interprétation* $[\![\cdot]\!] : R \to 2^S$
- $\bot, \top \in R$ sont telles que $\begin{cases} [\![\bot]\!] &= \emptyset \\ [\![\top]\!] &= S \end{cases}$
- $\sqcup, \sqcap : R \times R \to R$ sont telles que $\begin{cases} [\![r \sqcup r']\!] &= [\![r]\!] \cup [\![r']\!] \\ [\![r \sqcap r']\!] &= [\![r]\!] \cap [\![r']\!] \end{cases}$

- pré-ordre d'*inclusion induit* $\sqsubseteq$: $r \sqsubseteq r'$ ssi $[\![r]\!] \subseteq [\![r']\!]$

Exemple

- Unions finies de produits d'intervalles
- Unions finies de polyèdres

Représentation Symbolique

Définition

Une *représentation symbolique (en avant)* de $(S, \Sigma, \rightarrow)$ est un tuple $\mathcal{R} = (R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!], post_{\mathcal{R}})$ où :

- $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ est une algèbre de région pour S
- $post_{\mathcal{R}} : R \times \Sigma \rightarrow R$ est telle que $[\![post_{\mathcal{R}}(r, a)]\!] = post([\![r]\!], a)$

Représentation Symbolique

Définition

Une *représentation symbolique (en avant)* de $(S, \Sigma, \rightarrow)$ est un tuple $\mathcal{R} = (R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!], post_{\mathcal{R}})$ où :

- $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ est une algèbre de région pour S
- $post_{\mathcal{R}} : R \times \Sigma \rightarrow R$ est telle que $[\![post_{\mathcal{R}}(r, a)]\!] = post([\![r]\!], a)$

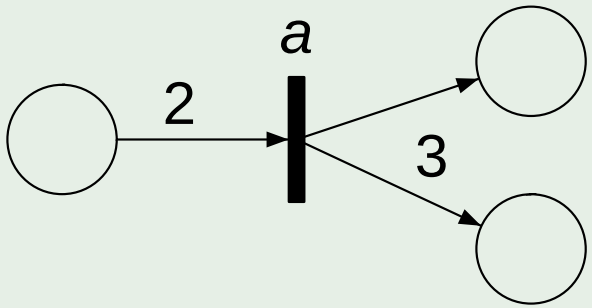
Représentation Symbolique

Définition

Une *représentation symbolique (en avant)* de $(S, \Sigma, \rightarrow)$ est un tuple $\mathcal{R} = (R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!], post_{\mathcal{R}})$ où :

- $(R, \bot, \top, \sqcup, \sqcap, [\![\cdot]\!])$ est une algèbre de région pour S
- $post_{\mathcal{R}} : R \times \Sigma \rightarrow R$ est telle que $[\![post_{\mathcal{R}}(r, a)]\!] = post([\![r]\!], a)$

Exemple



$$post_{\mathcal{R}}((I_1, I_2, I_3), a) = \begin{cases} \bot \text{ si } I_1 \subseteq [0,1] \\ (I_1 - 2, I_2 + 1, I_3 + 3) \text{ sinon} \end{cases}$$

Calcul Symbolique de $post^*$

Semi-algorithme `ExplSymb`

`let` $r \leftarrow r_0$
`do`
  `let` $r' \leftarrow r$
  `foreach` $a \in \Sigma$ `do`
    `let` $r \leftarrow r' \sqcup post_{\mathcal{R}}(r, a)$
`while` $r \not\sqsubseteq r'$
`if` $r \sqcap r_{err} \sqsubseteq \bot$ `return` “safe”
`else return` “unsafe”

Mais...

Avec :

- $[\![r_0]\!]$ fini, et
- $\mathcal{S}$ à branchement fini

`ExplSymb` ne termine pas dès que $post^*([\![r_0]\!])$ est infini

Accélération

Il faut accélérer la convergence !

Calcul Symbolique de $post^*$

Semi-algorithme `ExplSymb`

`let` $r \leftarrow r_0$
`do`
  `let` $r' \leftarrow r$
  `foreach` $a \in \Sigma$ `do`
    `let` $r \leftarrow r' \sqcup post_{\mathcal{R}}(r, a)$
`while` $r \not\sqsubseteq r'$
`if` $r \sqcap r_{err} \sqsubseteq \bot$ `return` “safe”
`else return` “unsafe”

Mais...

Avec :

- $[\![r_0]\!]$ fini, et
- $\mathcal{S}$ à branchement fini

`ExplSymb` ne termine pas dès que $post^*([\![r_0]\!])$ est infini

Accélération

Il faut accélérer la convergence !

Calcul Symbolique de $post^*$

Semi-algorithme `ExplSymb`

`let` $r \leftarrow r_0$
`do`
  `let` $r' \leftarrow r$
  `foreach` $a \in \Sigma$ `do`
    `let` $r \leftarrow r' \sqcup post_{\mathcal{R}}(r, a)$
`while` $r \not\sqsubseteq r'$
`if` $r \sqcap r_{err} \sqsubseteq \bot$ `return` “safe”
`else return` “unsafe”

Mais...

Avec :

- $[\![r_0]\!]$ fini, et
- $\mathcal{S}$ à branchement fini

`ExplSymb` ne termine pas dès que $post^*([\![r_0]\!])$ est infini

Accélération

Il faut accélérer la convergence !

Définition de l'accélération

Idée

- Calcul en une étape de l'effet de l'itération d'une suite d'actions
- Pour $X \subseteq S$ et $\sigma \in \Sigma^*$, calculer :

$$post(X, \sigma^*) = \{s' \in S \mid \exists s \in X, \exists i \in \mathbb{N}, s \xrightarrow{\sigma^i} s'\}$$

Définition

Une *accélération* pour $\mathcal{R}$ est une fonction $\nabla : R \times \Sigma^* \to R$ telle que :

$$[\![\nabla(r, \sigma)]\!] = post([\![r]\!], \sigma^*)$$

Définition de l'accélération

Idée

- Calcul en une étape de l'effet de l'itération d'une suite d'actions
- Pour $X \subseteq S$ et $\sigma \in \Sigma^*$, calculer :

$$post(X, \sigma^*) = \{s' \in S \mid \exists s \in X, \exists i \in \mathbb{N}, s \xrightarrow{\sigma^i} s'\}$$

Définition

Une *accélération* pour $\mathcal{R}$ est une fonction $\nabla : R \times \Sigma^* \to R$ telle que :

$$[\![\nabla(r, \sigma)]\!] = post([\![r]\!], \sigma^*)$$

C.N.S. d'existence d'une accélération

Définition

Une *accélération* est une fonction $\nabla : R \times \Sigma^* \to R$ telle que :

$$[\![\nabla(r, \sigma)]\!] = post([\![r]\!], \sigma^*)$$

Remarque

Une représentation symbolique $\mathcal{R}$ admet une accélération ssi pour tout $\sigma \in \Sigma^*$:

$post(X, \sigma^*)$ est $\mathcal{R}$-représentable pour tout X $\mathcal{R}$-représentable.

C.N.S. d'existence d'une accélération

Définition

Une *accélération* est une fonction $\nabla : R \times \Sigma^* \rightarrow R$ telle que :

$$[\![\nabla(r, \sigma)]\!] = post([\![r]\!], \sigma^*)$$

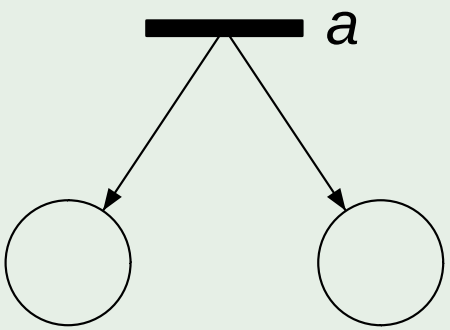
Remarque

Une représentation symbolique $\mathcal{R}$ admet une accélération ssi pour tout $\sigma \in \Sigma^*$:

$post(X, \sigma^*)$ est $\mathcal{R}$-représentable pour tout X $\mathcal{R}$-représentable.

Exemples d'accélération

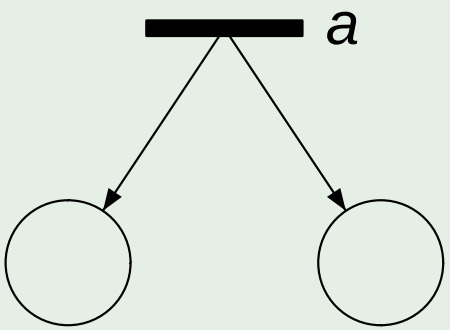
Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

Exemples d'accélération

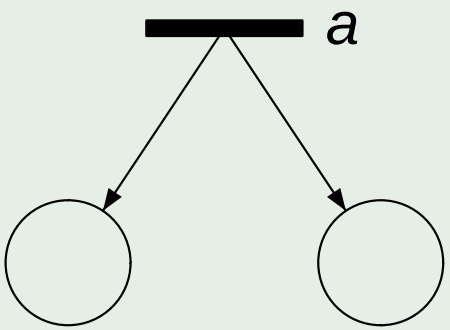
Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

Exemples d'accélération

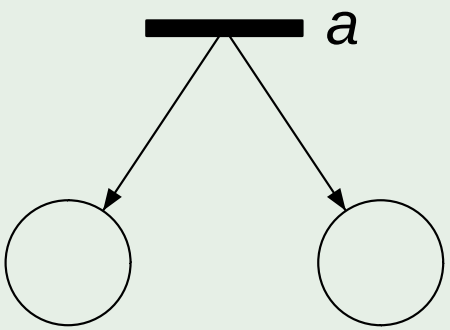
Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

Exemples d'accélération

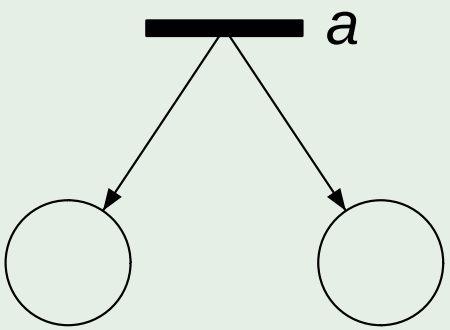
Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

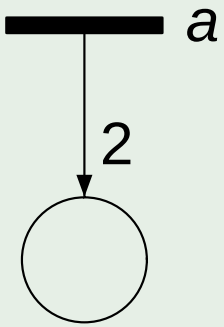
Exemples d'accélération

Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

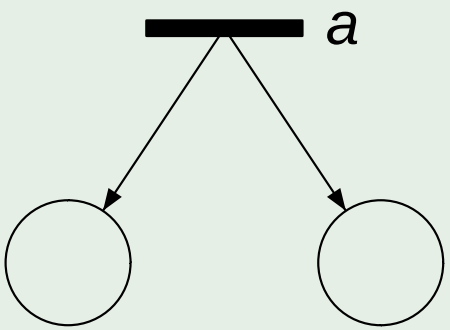
Exemple



- $post(\{(0,0)\}, a^*) = \{2i \mid i \in \mathbb{N}\}$
- unions finies de polyèdres : insuffisant
- polyèdres + modulo ?

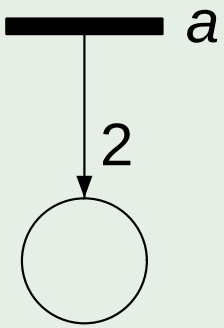
Exemples d'accélération

Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

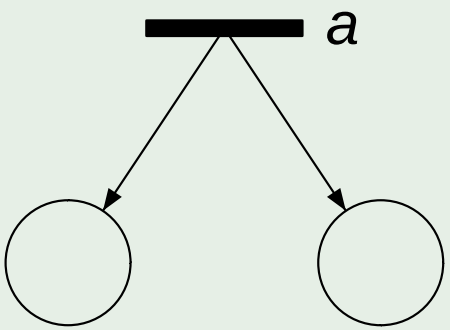
Exemple



- $post(\{(0,0)\}, a^*) = \{2i \mid i \in \mathbb{N}\}$
- unions finies de polyèdres : insuffisant
- polyèdres + modulo ?

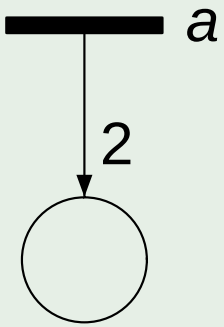
Exemples d'accélération

Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

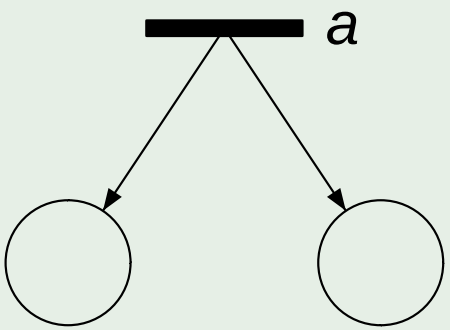
Exemple



- $post(\{(0,0)\}, a^*) = \{2i \mid i \in \mathbb{N}\}$
- unions finies de polyèdres : insuffisant
- polyèdres + modulo ?

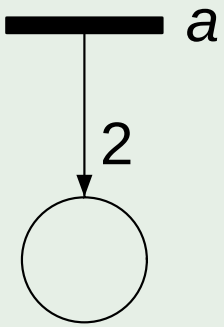
Exemples d'accélération

Exemple



- $post(\{(0,0)\}, a^*) = \{(i,i) \mid i \in \mathbb{N}\}$
- unions finies d'intervalles : insuffisant
- unions finies de polyèdres ?

Exemple



- $post(\{(0,0)\}, a^*) = \{2i \mid i \in \mathbb{N}\}$
- unions finies de polyèdres : insuffisant
- polyèdres + modulo ?

Ensembles semilinéaires : définition

Définition (Semilinéaire)

Union finie de *linéaires*

Définition (Linéaire)

Ensemble de la forme $b + \{p_1, \ldots, p_k\}^*$ où :

- $b \in \mathbb{N}^n$ est la *base*
- $p_1, \ldots, p_k \in \mathbb{N}^n$ sont les *périodes*
- $\{p_1, \ldots, p_k\}^*$ est l'ensemble des $\alpha_1 \cdot p_1 + \cdots + \alpha_k \cdot p_k$ avec $\alpha_i \in \mathbb{N}$

Propriétés

- Clos par $\cup, \cap, \setminus, +$
- Inclusion décidable

Presburger

Les semilinéaires sont les sous-ensembles de $\mathbb{N}^n$ définissables dans l'arithmétique de Presburger $\langle \mathbb{N}, +, \leq \rangle$.

Ensembles semilinéaires : définition

Définition (Semilinéaire)

Union finie de *linéaires*

Définition (Linéaire)

Ensemble de la forme $b + \{p_1, \ldots, p_k\}^*$ où :

- $b \in \mathbb{N}^n$ est la *base*
- $p_1, \ldots, p_k \in \mathbb{N}^n$ sont les *périodes*
- $\{p_1, \ldots, p_k\}^*$ est l'ensemble des $\alpha_1 \cdot p_1 + \cdots + \alpha_k \cdot p_k$ avec $\alpha_i \in \mathbb{N}$

Propriétés

- Clos par $\cup, \cap, \setminus, +$
- Inclusion décidable

Presburger

Les semilinéaires sont les sous-ensembles de $\mathbb{N}^n$ définissables dans l'arithmétique de Presburger $\langle \mathbb{N}, +, \leq \rangle$.

Ensembles semilinéaires : définition

Définition (Semilinéaire)

Union finie de *linéaires*

Définition (Linéaire)

Ensemble de la forme $b + \{p_1, \ldots, p_k\}^*$ où :

- $b \in \mathbb{N}^n$ est la *base*
- $p_1, \ldots, p_k \in \mathbb{N}^n$ sont les *périodes*
- $\{p_1, \ldots, p_k\}^*$ est l'ensemble des $\alpha_1 \cdot p_1 + \cdots + \alpha_k \cdot p_k$ avec $\alpha_i \in \mathbb{N}$

Propriétés

- Clos par $\cup, \cap, \setminus, +$
- Inclusion décidable

Presburger

Les semilinéaires sont les sous-ensembles de $\mathbb{N}^n$ définissables dans l'arithmétique de Presburger $\langle \mathbb{N}, +, \leq \rangle$.

Ensembles semilinéaires : définition

Définition (Semilinéaire)

Union finie de *linéaires*

Définition (Linéaire)

Ensemble de la forme $b + \{p_1, \ldots, p_k\}^*$ où :

- $b \in \mathbb{N}^n$ est la *base*
- $p_1, \ldots, p_k \in \mathbb{N}^n$ sont les *périodes*
- $\{p_1, \ldots, p_k\}^*$ est l'ensemble des $\alpha_1 \cdot p_1 + \cdots + \alpha_k \cdot p_k$ avec $\alpha_i \in \mathbb{N}$

Propriétés

- Clos par $\cup, \cap, \setminus, +$
- Inclusion décidable

Presburger

Les semilinéaires sont les sous-ensembles de $\mathbb{N}^n$ définissables dans l'arithmétique de Presburger $\langle \mathbb{N}, +, \leq \rangle$.

Ensembles semilinéaires : exemple

Définition (Semilinéaire)

Union finie de *linéaires*

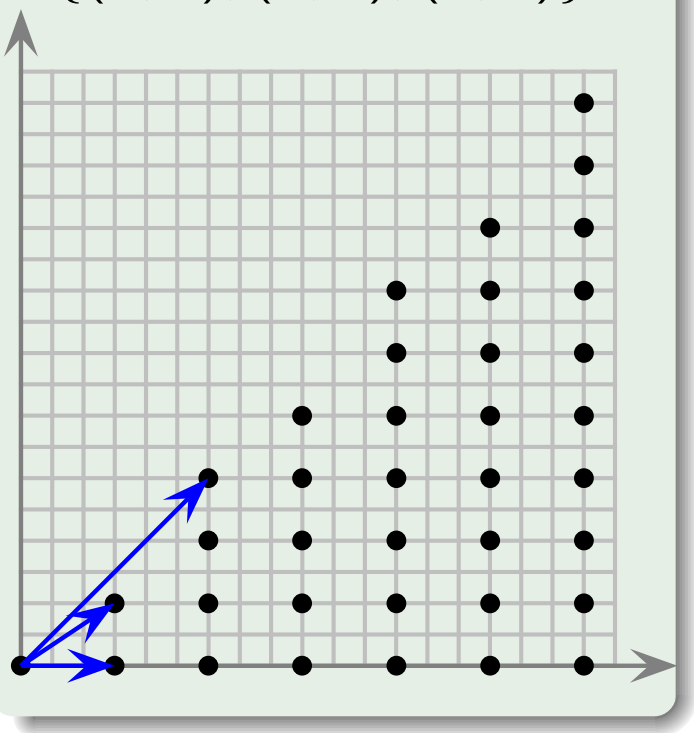
Définition (Linéaire)

Ensemble de la forme $b + \{p_1, \ldots, p_k\}^*$ où :

- $b \in \mathbb{N}^n$ est la *base*
- $p_1, \ldots, p_k \in \mathbb{N}^n$ sont les *périodes*
- $\{p_1, \ldots, p_k\}^*$ est l'ensemble des $\alpha_1 \cdot p_1 + \cdots + \alpha_k \cdot p_k$ avec $\alpha_i \in \mathbb{N}$

Exemple

$\{(3,0),(3,2),(6,6)\}^*$



Représentation symbolique de compteurs

Algèbre de régions

- R est l'ensemble des parties finies de $\mathbb{N}^n \times 2_f^{\mathbb{N}^n}$
- $[\![r]\!] = \bigcup_{(b,P)\in r} b + P^*$
- $\bot = \emptyset$ et $\top = \{(0, \{e_1, \ldots, e_n\})\}$
- $\cup$ est l'union standard ($\cap$ définie algorithmiquement)

Successeur symbolique

Pour un système à compteurs (Σ, G, F), $post_{\mathcal{R}}(r, a)$ est donné par la formule de Presburger (en x') :

$$\exists x \cdot \big(\varphi_r(x) \ \wedge \ G(a) \ \wedge \ x' = f(x)\big)$$

Représentation symbolique de compteurs

Algèbre de régions

- R est l'ensemble des parties finies de $\mathbb{N}^n \times 2_f^{\mathbb{N}^n}$
- $[\![r]\!] = \bigcup_{(b,P)\in r} b + P^*$
- $\bot = \emptyset$ et $\top = \{(0, \{e_1, \ldots, e_n\})\}$
- $\cup$ est l'union standard ($\cap$ définie algorithmiquement)

Successeur symbolique

Pour un système à compteurs (Σ, G, F), $post_{\mathcal{R}}(r, a)$ est donné par la formule de Presburger (en x') :

$$\exists x \cdot \big(\varphi_r(x) \ \wedge \ G(a) \ \wedge \ x' = f(x)\big)$$

Accélération pour les systèmes à compteurs

Système à compteurs à monoïde fini

(Σ, G, F) est à monoïde fini si le monoïde multiplicatif engendré par les matrices des fonctions affines $f(a)$ est fini.

Théorème (Boigelot 98, Finkel-Leroux 02)

Pour tout système à compteurs à monoïde fini et pour tout $\sigma \in \Sigma^*$, la relation $\{(x, x') \mid \exists i \in \mathbb{N}, x \xrightarrow{\sigma^i} x'\}$ est semilinéaire et calculable.

Corollaire

La représentation symbolique de compteurs basée sur les semilinéaires admet une accélération effective.

Accélération pour les systèmes à compteurs

Système à compteurs à monoïde fini

(Σ, G, F) est à monoïde fini si le monoïde multiplicatif engendré par les matrices des fonctions affines $f(a)$ est fini.

Théorème (Boigelot 98, Finkel-Leroux 02)

Pour tout système à compteurs à monoïde fini et pour tout $\sigma \in \Sigma^*$, la relation $\{(x, x') \mid \exists i \in \mathbb{N}, x \xrightarrow{\sigma^i} x'\}$ est semilinéaire et calculable.

Corollaire

La représentation symbolique de compteurs basée sur les semilinéaires admet une accélération effective.

Accélération pour les systèmes à compteurs

Système à compteurs à monoïde fini

(Σ, G, F) est à monoïde fini si le monoïde multiplicatif engendré par les matrices des fonctions affines $f(a)$ est fini.

Théorème (Boigelot 98, Finkel-Leroux 02)

Pour tout système à compteurs à monoïde fini et pour tout $\sigma \in \Sigma^*$, la relation $\{(x, x') \mid \exists i \in \mathbb{N}, x \xrightarrow{\sigma^i} x'\}$ est semilinéaire et calculable.

Corollaire

La représentation symbolique de compteurs basée sur les semilinéaires admet une accélération effective.

Exploration symbolique accélérée

Semi-algorithme ExplSymbAcc

let $r \leftarrow r_0$
do
  let $r' \leftarrow r$
  forsome $\sigma \in \Sigma^*$ do
    let $r \leftarrow r' \sqcup \nabla(r, \sigma)$
  foreach $a \in \Sigma$ do
    let $r \leftarrow r' \sqcup post_{\mathcal{R}}(r, a)$
while $r \not\sqsubseteq r'$
if $r \sqcap r_{err} \sqsubseteq \bot$ return “safe”
else return “unsafe”

Implémentation

Représentation symbolique

- Bases / périodes
- Formules de Presburger + solveur de contraintes
- Number Decision Diagrams (automates) $\Longrightarrow$ représentation canonique

Stratégies d'accélération

- choix des bonnes séquences $\sigma \in \Sigma^*$ à accélérer
- manuel, exhaustif ou guidé par l'exploration symbolique

Implémentation

Représentation symbolique

- Bases / périodes
- Formules de Presburger + solveur de contraintes
- Number Decision Diagrams (automates) $\Longrightarrow$ représentation canonique

Stratégies d'accélération

- choix des bonnes séquences $\sigma \in \Sigma^*$ à accélérer
- manuel, exhaustif ou guidé par l'exploration symbolique

Outils

LASH (Univ. Liège)

- représentation symbolique par automates
- choix manuel des séquences $\sigma \in \Sigma^*$ à accélérer

TReX (LIAFA, Paris 7)

- représentation symbolique par DBM paramétriques
- choix guidé par l'exploration des séquences à accélérer

FAST (LSV, ENS Cachan – LaBRI)

- représentation symbolique par automates
- sélection exhaustive des séquences à accélérer

Eléments de complétude

Et la terminaison du semi-algorithme ?

- La vérification des réseaux de Petri avec tests-à-zéro est indécidable
- Le semi-algorithme peut ne pas terminer
- Mais on obtient de bons résultats en pratique

C.N. de terminaison

Si $post^*$ est calculable par exploration symbolique, alors $post^*$ est semilinéaire

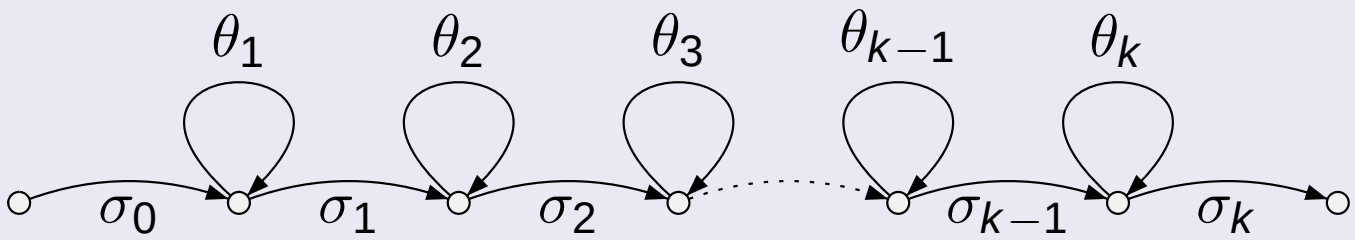
Remarque

Pour les réseaux de Petri étendus avec perte, $post^*$ est semilinéaire mais non calculable

Platitude

Schéma semilinéaire de chemins

Union finie de languages réguliers de la forme $\sigma_0\theta_1^*\sigma_1\cdots\theta_k^*\sigma_k$



Définition

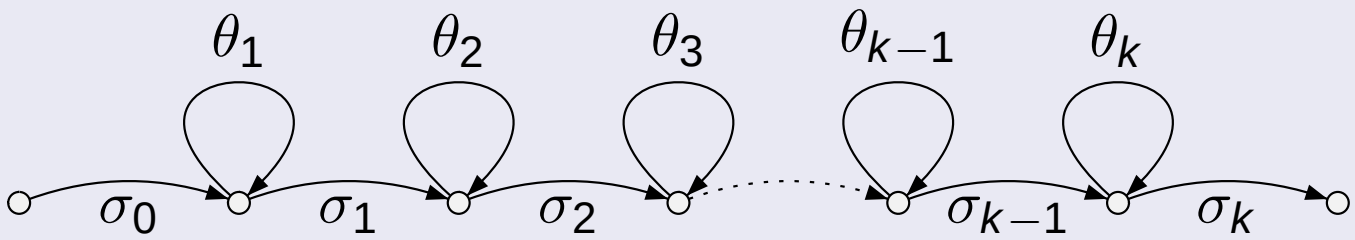
Un système de transition initialisé (S, S_0) est *plat* s'il existe un schéma semilinéaire de chemins ρ tel que :

$$post^*(S_0) = post(S_0, \rho)$$

Platitude

Schéma semilinéaire de chemins

Union finie de languages réguliers de la forme $\sigma_0\theta_1^*\sigma_1\cdots\theta_k^*\sigma_k$



Définition

Un système de transition initialisé $(\mathcal{S}, S_0)$ est *plat* s'il existe un schéma semilinéaire de chemins ρ tel que :

$$post^*(S_0) = post(S_0, \rho)$$

Sur l'omniprésence des systèmes à compteurs plats...

Théorème

Le semi-algorithme `ExplSymbAcc` termine sur $\mathcal{S}$ (muni d'une représentation symbolique) ssi $\mathcal{S}$ est plat.

Classes semilinéaires de systèmes compteurs

- Nombreuses classes de réseaux de Petri : réversibles, cycliques, persistents, ...
- Machines à compteurs "reversal-bounded", à compteurs avec perte / insertion

Théorème (Leroux-Sutre 05)

Toutes ces classes semilinéaires de systèmes à compteurs sont plates.

Plan

Résumé

Systèmes infinis

- Modélisation précise de certains aspects des systèmes
- Algorithmes dédiés, exploration symbolique, abstraction

Réseaux de Petri et systèmes bien structurés

- Calcul de $pre^*(\uparrow S_0)$, Karp-Miller
- Extension aux systèmes de transition bien structurés

Exploration symbolique accélérée générique

- Régions, représentation symbolique, accélération
- Application aux systèmes à compteurs (semilinéaires, résultats d'accélération)
- Mise en oeuvre, complétude

Au-delà des compteurs ?

Algorithmes de vérification dédiés

- Automates à pile (programmes récursifs)
- Algèbres de processus
- Classes de systèmes hybrides, automates temporisés
- ...

Exploration symbolique accélérée

- Systèmes communicants par file FIFO
- Systèmes temporisés, systèmes hybrides

Au-delà des compteurs ?

Algorithmes de vérification dédiés

- Automates à pile (programmes récursifs)
- Algèbres de processus
- Classes de systèmes hybrides, automates temporisés
- ...

Exploration symbolique accélérée

- Systèmes communicants par file FIFO
- Systèmes temporisés, systèmes hybrides

Perspectives

Nouveaux types de données

- Systèmes hétérogènes
- Systèmes à pointeurs

Théorie de l'accélération

- Au-delà de l'itération d'une séquence ?
- Complétude

Abstraction + accélération ?

- Calcul accéléré dans les domaines abstraits
- Abstraction automatique vers des modèles infinis, raffinement