LIP6 - UPMC
Master SAR - RAAR

Année 2010–2011
Aide mémoire

Aspects sémantiques des systèmes dynamiques

Table des matières

1 Introduction : problématiques de modélisation et de vérification

2 Systèmes de transitions

2.1 Définitions générales

Définition 2.1.
Un système de transitions sur l'alphabet d'actions Act est un triplet $\mathcal{T} = (S, S_0, T)$ où S est l'ensemble des configurations, S_0 est le sous-ensemble de S des configurations initiales et $T \subseteq S \times (Act \cup \{\varepsilon\}) \times S$ est l'ensemble des transitions.

On note généralement $s \xrightarrow{a} s'$ une transition (s, a, s') de T. Une transition étiquetée par le mot vide ε correspond à l'exécution d'une action non observable, dite *action interne*. Rappelons que pour tout mot w de Act^*, $w \cdot \varepsilon = w$.

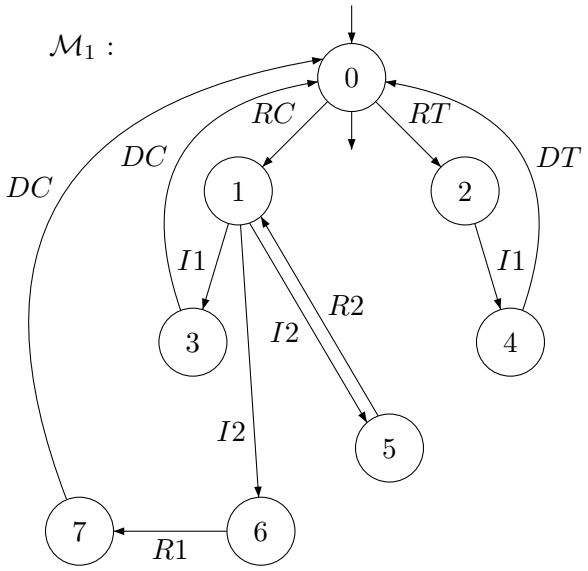
Une exécution de $\mathcal{T}$ est un chemin dans ce graphe, c'est-à-dire une séquence de transitions $(s_0, a_1, s_1)(s_1, a_2, s_2)\ldots$, qui s'écrit aussi $s \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \ldots$, commençant dans une configuration initiale. Le mot $w = a_1 a_2 \ldots$ est l'*étiquette* du chemin (ou de l'exécution).

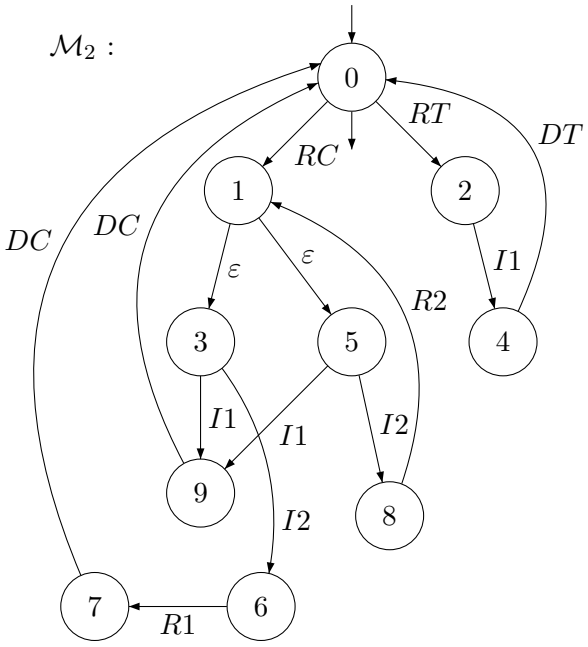
Un automate fini est un système de transitions

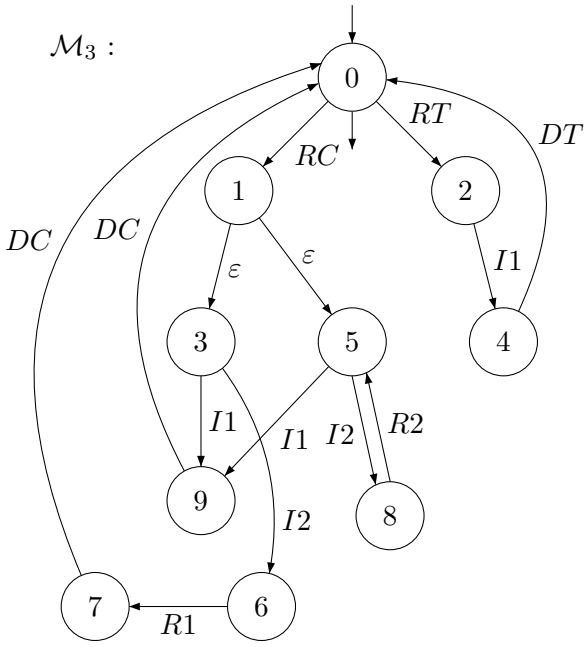
- dont l'ensemble S des configurations est fini (les éléments de S sont appelés des états),
- auquel on adjoint un sous-ensemble F de S correspondant aux états finaux.

Si $\mathcal{M} = (S, S_0, T, F)$ est un automate fini sur Act, on peut lui associer le langage des mots acceptés, noté $L(\mathcal{M})$ et défini comme suit :
un mot w de Act^* appartient à $L(\mathcal{M})$ s'il existe une exécution finie $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \dots \xrightarrow{a_n} s_n$ commençant dans un état initial ($s_o \in S_0$), se terminant dans un état final ($s_n \in F$) et ayant w comme étiquette.

2.2 Des modèles de machines à café







2.3 Equivalence de langages

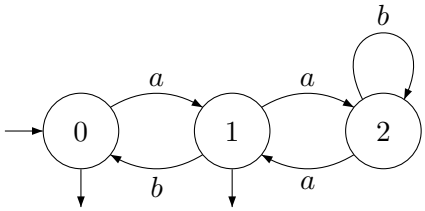
Deux automates finis $\mathcal{M}_1$ et $\mathcal{M}_2$ sur un même alphabet *Act* sont langage-équivalents si $L(\mathcal{M}_1) = L(\mathcal{M}_2)$. Pour tester cette équivalence, on peut penser d'abord à calculer une expression rationnelle qui décrit chacun des deux langages.
Pour cela, on considère la famille d'automates obtenus à partir de $\mathcal{M} = (S, S_0, T, F)$ en prenant un état quelconque $s \in S$ comme état initial : $\mathcal{M}_s = (S, \{s\}, T, F)$ et on note L_s le langage accepté par $\mathcal{M}_s$.
Il est alors facile de voir que si les transitions sortant de s sont $s \xrightarrow{a_1} s_1$, $s \xrightarrow{a_2} s_2$, ..., $s \xrightarrow{a_p} s_p$, on a l'égalité suivante :

$$L_s = \begin{cases} \varepsilon + a_1 L_{s_1} + \cdots + a_p L_{s_p} \text{ si } s \text{ est un état final} \\ a_1 L_{s_1} + \cdots + a_p L_{s_p} \text{ sinon} \end{cases}$$

À partir de là, on résoud le système d'équations en utilisant la proposition 2.2 suivante :

Proposition 2.2. *Soient X et M deux langages et K un langage ne contenant pas le mot vide ε. L'équation $X = KX + M$ a pour unique solution $L = K^*M$.*

Exercice 1.
Calculer l'expression rationnelle décrivant le langage accepté par l'automate fini ci-dessous.



Exercice 2.
Trouver une autre expression rationnelle décrivant le même langage que $E = (a^*b)^* + a^*$.

Exercice 3.
Quelle autre méthode peut-on envisager pour tester l'équivalence de langages ?

2.4 Déterminisation

Définition 2.3. *Un automate fini $\mathcal{M} = (S, S_0, T, F)$ sur Act est déterministe si :*
- *il a un unique état initial $S_0 = \{s_0\}$,*
- *aucune transition n'est étiquetée par ε,*
- *pour tous les états s, s_1, s_2 de S et pour toute action $a \in Act$, si $s \xrightarrow{a} s_1$ et $s \xrightarrow{a} s_2$ alors $s_1 = s_2$.*

On a alors :

Proposition 2.4. *Pour tout automate fini $\mathcal{M}$, on peut construire un automate fini déterministe $\mathcal{D}$ qui soit langage-équivalent à $\mathcal{M}$.*

Partant de $\mathcal{M} = (S, S_0, T, F)$, la construction est la suivante :
- L'ensemble des états de $\mathcal{D}$ est l'ensemble des parties de S, noté 2^S.
- L'unique état initial de $\mathcal{D}$ est la partie

$$I = \{s \in S \mid \text{ il existe un état } s_0 \in S_0 \text{ et un chemin d'étiquette } \varepsilon \text{ de } s_0 \text{ à } s\}.$$

– Une partie P de S est un état final de $\mathcal{D}$ ssi $P \cap F \neq \emptyset$.
– Pour toute partie P de S et toute action a de Act, la transition associée de $\mathcal{D}$ est $P \xrightarrow{a} P'$ avec :

$$P' = \{s' \in S \mid \text{il existe des états } s_1 \in S \text{ et } s \in P \text{ tels que } s \xrightarrow{a} s_1 \text{ est une transition dans } T \text{ et il existe un chemin d'étiquette } \varepsilon \text{ de } s_1 \text{ à } s'\}.$$

2.5 Equivalences comportementales

Définition 2.5. *Deux systèmes de transitions $\mathcal{T}_1$ et $\mathcal{T}_2$ sont fortement bisimilaires s'il existe une relation $\mathcal{S}$ entre leurs états ($\mathcal{S} \subseteq S_1 \times S_2$) telle que :*
– *toute configuration initiale de $\mathcal{T}_1$ est en relation avec une configuration initiale de $\mathcal{T}_2$, et réciproquement,*
– *si $s_1 \mathcal{S} s_2$ et $s_1 \xrightarrow{a} s'_1$ dans $\mathcal{T}_1$ alors il existe $s'_2 \in S_2$ tel que $s_2 \xrightarrow{a} s'_2$ dans $\mathcal{T}_2$ et $s'_1 \mathcal{S} s'_2$,*
– *réciproquement, si $s_1 \mathcal{S} s_2$ et $s_2 \xrightarrow{a} s'_2$ dans $\mathcal{T}_2$ alors il existe $s'_1 \in S_1$ tel que $s_1 \xrightarrow{a} s'_1$ dans $\mathcal{T}_1$ et $s'_1 \mathcal{S} s'_2$.*

Ces équivalences sont présentées dans le cadre des systèmes de transitions. Pour des automates finis $\mathcal{M}_1$ et $\mathcal{M}_2$, il faut ajouter des conditions reliant les états finaux :
tout état final de $\mathcal{M}_1$ est en relation avec un état final de $\mathcal{M}_2$, et réciproquement.
La bisimilarité (faible) accepte des séquences non observables avant et après une action. Pour une action $a \in Act \cup \{\varepsilon\}$, on définit :

$$s \overset{a}{\Rightarrow} s' \text{ s'il existe un chemin d'étiquette } a \text{ de } s \text{ à } s'$$

Définition 2.6. *Deux systèmes de transitions $\mathcal{T}_1$ et $\mathcal{T}_2$ sont (faiblement) bisimilaires s'il existe une relation $\mathcal{S}$ entre leurs états, qui met en relation leurs états initiaux, et telle que :*
si $s_1 \mathcal{S} s_2$, alors :
si $s_1 \xrightarrow{a} s'_1$ dans $\mathcal{T}_1$ alors il existe $s'_2 \in S_2$ tel que $s_2 \overset{a}{\Rightarrow} s'_2$ dans $\mathcal{T}_2$ et $s'_1 \mathcal{S} s'_2$,
et réciproquement.

Là encore, pour des automates finis, il faut ajouter les conditions énoncées plus haut sur les états finaux. Dans ce cadre, on a bien sûr les propriétés suivantes :
si deux automates sont fortement bisimilaires, alors ils sont faiblement bisimilaires,
si deux automates sont faiblement bisimilaires alors ils sont langage-équivalents.

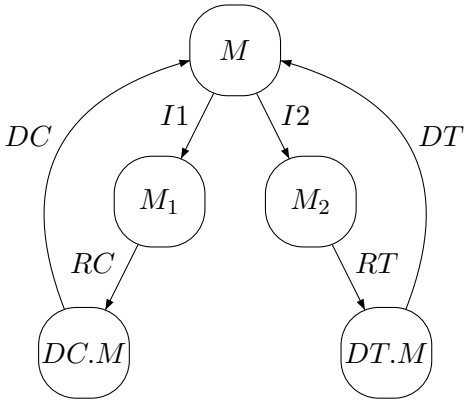
3 Algèbres de processus

3.1 Définition

Les algèbres de processus proposent une description syntaxique de systèmes de transitions. On pourrait définir une quatrième machine à café par les équations suivantes :

$M \overset{def}{=} I_1.M_1 + I_2.M_2$
$M_1 \overset{def}{=} RC.DC.M$
$M_2 \overset{def}{=} RT.DT.M$

ce qui décrit le système de transition :



La définition qui suit est assez restreinte, elle est susceptible d'être étendue par la suite. On suppose que $Act = A \cup \overline{A}$, où les actions de $\overline{A}$ sont les complémentaires des actions de A, avec $\overline{\overline{a}} = a$.

Définition 3.1. *L'ensemble $\mathcal{E}$ des* expressions *(ou termes) d'une algèbre de processus sur un ensemble d'actions Act, contient des variables, des constantes, et est défini récursivement par :*

$a\ E$ *pour* $a \in Act \cup \{\varepsilon\}$ *et* $E \in \mathcal{E}$,
$E_1 + E_2$ *pour* $E_1, E_2 \in \mathcal{E}$,
$E_1 \parallel E_2$ *pour* $E_1, E_2 \in \mathcal{E}$,
$E \setminus K$ *pour* $K \subseteq Act$.

Un *processus* est une expression sans variable libre et une constante est un processus défini par une équation utilisant l'opérateur def (comme dans l'exemple initial). Toute constante est définie par une telle règle.

On pourra par exemple ajouter à ces opérateurs une somme généralisée d'expressions E_i où les indices i sont dans un ensemble I, cette somme valant $\mathbf{0}$ (le processus constant qui ne fait rien) si I est l'ensemble vide.

3.2 Système de transitions associé

Le système de transition associé est obtenu par $\mathcal{T} = (\mathcal{E}, E_0, T)$ sur $Act \cup \{\varepsilon\}$, où les états sont les expressions, un processus initial est choisi et les transitions de T sont données par les règles suivantes.

1) Règles de définition et de préfixage par une action

$$\text{Def} : \frac{E \xrightarrow{a} F}{P \xrightarrow{a} F} \quad \text{si } P \overset{def}{=} E \qquad\qquad \text{Act} : \frac{}{a\ P \xrightarrow{a} P}$$

2) Règles de choix

$$\text{Choix1} : \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} \qquad\qquad \text{Choix2} : \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}$$

ou plus généralement

$$\text{Choix} : \frac{P_j \xrightarrow{a} P'_j}{\sum_{i \in I} P_i \xrightarrow{a} P'_j} \quad \text{si } j \in I$$

3) Règles de composition prallèle

$$\text{Comp1}: \ \frac{P \xrightarrow{a} P'}{P \parallel Q \xrightarrow{a} P' \parallel Q} \qquad \text{Comp2}: \ \frac{Q \xrightarrow{a} Q'}{P \parallel Q \xrightarrow{a} P \parallel Q'}$$

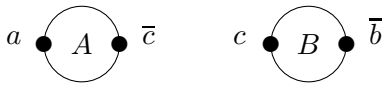
avec la synchronisation

$$\text{Comp3}: \ \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P \parallel Q \xrightarrow{\varepsilon} P' \parallel Q'}$$

4) Règle de restriction

$$\text{Res}: \ \frac{P \xrightarrow{a} P'}{P \setminus K \xrightarrow{a} P' \setminus K} \quad \text{si } a, \overline{a} \notin K$$

Exercice 4. On considère les définitions suivantes, correspondant à deux processus A et B configurés comme ci-dessous :



$$A \stackrel{def}{=} a\ A' \qquad B \stackrel{def}{=} c\ B' \qquad A' \stackrel{def}{=} \overline{c}\ A \qquad B' \stackrel{def}{=} \overline{b}\ B$$

(a) Dessiner le système de transition correspondant aux deux composants indépendants.
(b) Dessiner le système de transition associé au processus $A \parallel B$.
(c) Dessiner le système de transition associé au processus $(A \parallel B) \setminus \{c\}$. Justifier par le système d'inférence les règles suivantes :

$$(A \parallel B) \setminus \{c\} \xrightarrow{a} (A' \parallel B) \setminus \{c\}$$
$$(A' \parallel B) \setminus \{c\} \xrightarrow{\tau} (A \parallel B') \setminus \{c\}$$

Exercice 5. De façon générale, justifier pour deux processus P et Q la règle :

$$((a\ P + b\ \mathbf{0}) \parallel \overline{a}\ Q) \setminus a \xrightarrow{\varepsilon} (P \parallel Q) \setminus a$$

Exercice 6.

(a) Montrer que $(A \parallel B) \setminus c$ est fortement bisimilaire au processus C_1 obtenu par les définitions suivantes :

$$C_0 \stackrel{def}{=} \overline{b}\ C_1 + a\ C_2 \qquad C_1 \stackrel{def}{=} a\ C_3 \qquad C_2 \stackrel{def}{=} \overline{b}\ C_3 \qquad C_3 \stackrel{def}{=} \overline{\tau}\ C_0$$

(b) Quelle relation a-t-on entre les processus $(A \parallel B) \setminus c$ et $a\ D$ où $D \stackrel{def}{=} a\ \overline{b}\ D + \overline{b}\ a\ D$?

Exercice 7. Comparer les processus P et Q définis par : $P \stackrel{def}{=} a\ \mathbf{0} + b\ \mathbf{0}$ et $Q \stackrel{def}{=} a\ \mathbf{0} + \tau\ b\ \mathbf{0}$.

Exercice 8. Comparer les systèmes obtenus par les définitions suivantes :

$$A_0 \stackrel{def}{=} a\ A_0 + b\ A_1 + \tau\ A_1 \qquad A_1 \stackrel{def}{=} a\ A_1 + \tau\ A_2 \qquad A_2 \stackrel{def}{=} b\ A_0$$

et

$$B_1 \stackrel{def}{=} a\ B_1 + \tau\ B_2 \qquad B_2 \stackrel{def}{=} b\ B_1$$

4 Automates temporisés

4.1 Systèmes de transitions temporisés

On considère un domaine de temps $\mathbb{T}$ qui peut être $\mathbb{N}$, $\mathbb{Q}$ ou $\mathbb{R}$ et qui sera toujours plongé dans $\mathbb{R}$.

Définition 4.1.
Un système de transitions temporisé est un système de transitions $\mathcal{T} = (S, S_0, T)$ sur l'alphabet $Act \cup \mathbb{T}$. L'ensemble T des transitions est donc un sous-ensemble de $S \times (Act \cup \{\varepsilon\} \cup \mathbb{T}) \times S$, avec deux types de transitions :
- *les transitions d'actions $s \xrightarrow{a} s'$ avec $a \in Act \cup \{\varepsilon\}$,*
- *les transitions de délais $s \xrightarrow{d} s'$ avec $d \in \mathbb{T}$.*

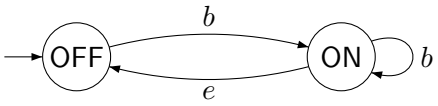
Les transitions $\xrightarrow{d}$, pour $d \in \mathbb{T}$, expriment l'écoulement d'une durée d et doivent de ce fait vérifier des conditions particulières, qui traduisent la compatibilité du système par rapport aux opérations sur le temps :
- *délai nul :* $s \xrightarrow{0} s'$ si et seulement si $s' = s$,
- *additivité :* si $s \xrightarrow{d} s'$ et $s' \xrightarrow{d'} s''$, alors $s \xrightarrow{d+d'} s''$.

De plus, il est parfois exigé du système $\mathcal{T}$ qu'il soit :
- *déterministe par rapport au temps :* $s \xrightarrow{d} s_1$ et $s \xrightarrow{d} s_2$ implique $s_1 = s_2$,
- *continu :* $s \xrightarrow{d} s'$ implique que, pour tous d' et d'' tels que $d = d' + d''$, il existe s'' tel que $s \xrightarrow{d'} s''$ et $s'' \xrightarrow{d''} s'$.

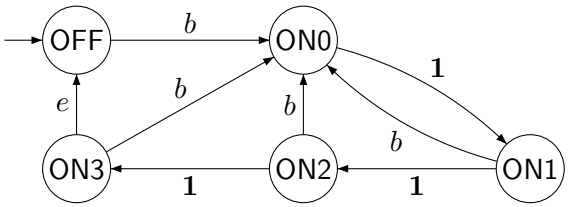
Dans ce cas, une exécution de $\mathcal{T}$ est un chemin $\rho = s_0 \xrightarrow{d_1} s'_0 \xrightarrow{a_1} s_1 \xrightarrow{d_2} s'_1 \xrightarrow{a_2} \cdots$ partant d'une configuration initiale et où les durées alternent strictement avec les actions. A une telle exécution peuvent être associés :
- la suite $(t_i)_{i \geq 0}$ des dates absolues, qui est naturellement donnée par $t_0 = 0$ et $t_i = \sum_{j=0}^{i} d_j$,
- le *mot temporisé* $w = (a_1, t_1)(a_2, t_2)\ldots$, qui correspond à l'observation de la suite des événements, a_i se produisant à la date t_i. C'est donc un mot sur l'alphabet $Act \times \mathbb{T}$ (les paires correspondant à des actions non observables ayant été absorbées).

Le langage associé à un système de transitions temporisé $\mathcal{T}$ est alors l'ensemble des mots temporisés associés à ses exécutions.

Exemple : un interrupteur.



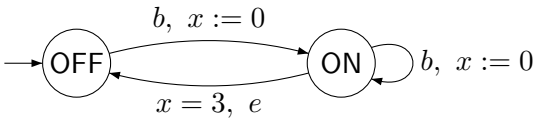
On souhaite ajouter la contrainte : *la lumière reste allumée 3 unités de temps après appui sur le bouton.* Un premier système peut être obtenu par un dépliage en utilisant un temps discret, avec un risque d'explosion combinatoire dû aux constantes impliquées.



4.2 Automates temporisés

Les automates temporisés sont des automates finis qui manipulent des variables appelées horloges, car elles évoluent de manière synchrone avec le temps dans les états. Ces horloges peuvent être testées et remises à zéro lors des transitions.

Ainsi, avec une horloge x, l'interrupteur serait décrit par l'automate temporisé suivant, qui reprend simplement l'automate fini original en le décorant :



On peut aussi adjoindre à l'état ON la contrainte $x \leq 3$ sur l'horloge x pour restreindre la sémantique aux exécutions pour lesquelles la valeur de l'horloge x ne peut pas dépasser 3 dans cet état.

Une exécution dans ce modèle est décrite par l'évolution des états et celle des horloges. Dans l'exemple, on pourrait avoir :

$$(\mathsf{OFF}, 0) \xrightarrow{4.6} (\mathsf{OFF}, 4.6) \xrightarrow{b} (\mathsf{ON}, 0) \xrightarrow{1.7} (\mathsf{ON}, 1.7) \xrightarrow{b} (\mathsf{ON}, 0) \xrightarrow{3} (\mathsf{ON}, 3) \xrightarrow{e} (\mathsf{OFF}, 3)$$

De façon générale, pour un ensemble X d'horloges, on note $\mathcal{C}(X)$ l'ensemble des conjonctions de contraintes atomiques de la forme $x \bowtie c$ où x est une horloge, c une constante (généralement entière) et $\bowtie$ un opérateur de comparaison dans $\{<, \leq, =, \geq, >\}$.

Définition 4.2.
Un automate temporisé *sur un alphabet Act est un quintuplet $\mathcal{A} = (X, Q, q_0, \Delta, Inv)$, où :*

- *X est un ensemble d'horloges,*
- *Q est un ensemble fini d'états (ou modes de contrôle),*
- *$q_0 \in Q$ est l'état initial,*
- *Δ est un sous-ensemble de $Q \times \mathcal{C}(X) \times Act \times 2^X \times Q$,*
- *$Inv : Q \mapsto \mathcal{C}(X)$ associe à chaque état une contrainte de $\mathcal{C}(X)$, appelée* invariant, *formée en utilisant seulement les opérateurs $<$ et $\leq$.*

Une transition de Δ, notée $q \xrightarrow{g,a,r} q'$, exprime un passage possible de q à q' avec l'action a, si la garde g est vraie. Les horloges de $r \subseteq X$ sont alors remises à zéro, ce qui est souvent noté aussi $r := 0$, comme c'est le cas dans l'exemple pour la remise à zéro de l'horloge x.

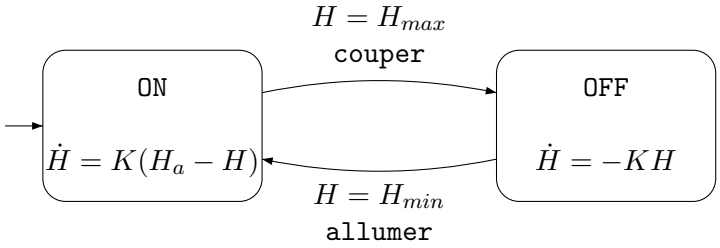
Les contraintes d'horloges (gardes et invariants) sont interprétées sur des valuations d'horloges, c'est-à-dire des applications de X dans $\mathbb{R}_+$, avec $\mathbf{0}$ représentant la valuation nulle pour toutes les horloges de X. Une valuation v satisfait une contrainte g de la forme $x \bowtie c$, noté $v \models g$, si $v(x) \bowtie c$.

On définit :
- le passage du temps depuis une valuation v par :
  $(v+d)(x) = v(x) + d$ pour toute horloge x et tout $d \in \mathbb{T}$, ce qui exprime l'évolution synchrone des valeurs d'horloge lorsqu'une durée d s'écoule,
- la remise à zéro des horloges de $r \subseteq X$ à partir de v par : $v[r \mapsto 0](x) = 0$ si $x \in r$ et 0 sinon.

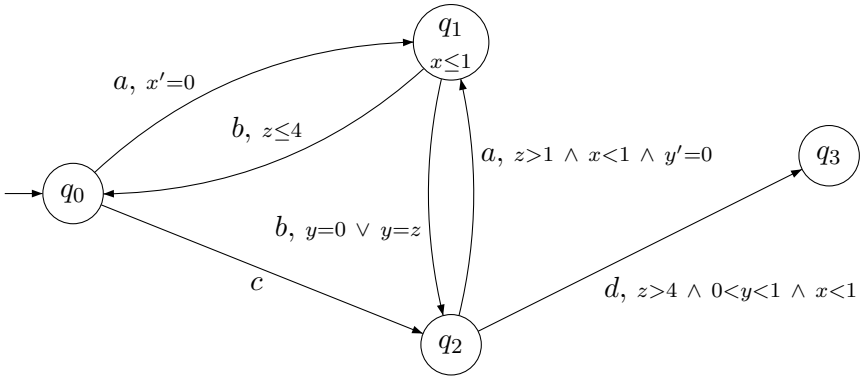
La sémantique d'un automate temporisé $\mathcal{A} = (X, Q, q_0, \Delta, Inv)$ est donnée par un système de transitions temporisé $\mathcal{T}_\mathcal{A} = (S, \{s_0\}, T)$, sur $Act \cup \{\varepsilon\} \cup \mathbb{T}$, avec :
- $S = \{(q, v) \in Q \times \mathbb{R}_+^X \mid v \models Inv(q)\}$. Les configurations sont donc des couples (q, v) où $q \in Q$ et v est une valuation d'horloges satisfaisant l'invariant de l'état.
- La configuration initiale est $s_0 = (q_0, \mathbf{0})$.
- Les transitions de T sont :
  - soit $(q, v) \xrightarrow{d} (q, v + d)$, une transition de durée $d \in \mathbb{T}$, possible si $v + d \models Inv(q)$,
  - soit $(q, v) \xrightarrow{a} (q', v')$, une transition discrète d'étiquette $a \in Act \cup \{\varepsilon\}$, avec $v' = v[r \mapsto 0]$, possible s'il existe $q \xrightarrow{g,a,r} q'$ dans Δ telle que la valuation v satisfasse la garde g.

Remarque. Ces définitions peuvent être généralisées à des modèles plus puissants, comme les automates hybrides :
- en considérant des gardes plus complexes $\sum_{i=1}^{n} a_i x_i + b \bowtie 0$,
- ou en considérant des mises à jour au lieu de simples remises à zéro : $x := \sum_{i=1}^{n} a_i x_i + b$,
- ou encore en considérant des variables dont l'évolution est plus générale que celle des horloges (qui sont affines de pente égale à 1).

Un exemple classique de la littérature est celui du thermostat :



Exercice 9. On considère l'automate hybride linéaire $\mathcal{A}_1$ ci-dessous. Les variables x et y sont des horloges (pente 1 partout) tandis que z a pour pente 1 dans q_1 et 0 dans les autres états de contrôle.



L'état de contrôle q_3 est-il accessible depuis la configuration initiale $(q_0, 0, 0, 0)$? Si oui, donnez une exécution menant à une configuration $(q_3, -, -, -)$.

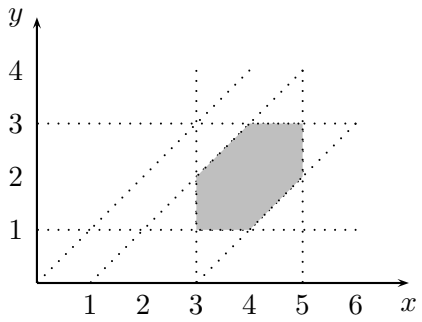
4.3 Zones - Exercice 10

On considère un ensemble X contenant n horloges et on note $\mathcal{C}_1(X)$ l'ensemble des contraintes qui sont des **conjonctions** de contraintes atomiques $x \bowtie k$ ou $x - y \bowtie k$, avec $x, y \in X$, $k \in \mathbb{Z}$ et $\bowtie \in \{\leq, <, =, >, \geq\}$. Une *zone* est un sous-ensemble de $\mathbb{R}_+^n$ défini par une telle contrainte et pour une zone Z, on définit le *futur* de Z, noté $\overrightarrow{Z}$, comme l'ensemble $\{v + d \mid v \in Z, d \in \mathbb{R}_+\}$.

1. Montrer que l'intersection de deux zones est une zone. La réunion de deux zones est-elle une zone ?

2. Soit $\mathcal{A}$ un automate temporisé et q un état de contrôle de $\mathcal{A}$. On considère une transition $q \xrightarrow{g,a,r} q'$ de $\mathcal{A}$, où g est une contrainte de $\mathcal{C}_1(X)$. On suppose que l'ensemble des valuations v atteintes dans q forme une zone Z. Expliquer pourquoi l'ensemble des valuations que l'on peut atteindre dans q' est $Z' = (\overrightarrow{Z} \cap Z_g)[Y \leftarrow 0]$, où Z_g est la zone associée à g. Cet ensemble Z' est-il une zone ?

3. Pour $X = \{x, y\}$, on considère la zone Z_0 dessinée ci-dessous.



(a) Donner une conjonction de contraintes qui la définit. Donner les contraintes définissant $\overrightarrow{Z_0}$.

(b) Construire un automate temporisé $\mathcal{A}$ qui génère cette zone, c'est-à-dire que $\mathcal{A}$ comporte un état q pour lequel l'ensemble des valuations v obtenues est exactement Z_0.

4.4 Algèbres de processus temporisées

On peut définir (encore) une autre variante d'un distributeur de chocolat et de biscuit :

$$M \stackrel{def}{=} in.[(choc.M + bisc.M) \stackrel{60}{\triangleright} (out!.M)]$$

La machine peut recevoir une pièce (in), puis le client choisit entre un chocolat et un biscuit ; cependant, s'il ne choisit pas en moins de 60 secondes, son argent lui est rendu

immédiatement (*out*!) et la machine retourne dans son état initial. On observe dans cette définition le mécanisme de *Time Out*, où un processus $P \overset{d}{\triangleright} Q$ se comporte comme P dans un délai inférieur à d unités de temps, puis au bout de ce délai, se comporte comme Q. Cet opérateur est fréquemment utilisé dans les algèbres de processus temporisées. On observe également l'ajout d'actions dites *immédiates*, notées comme ici *out*!, qui ne peuvent pas être retardées.

On peut associer à ce processus l'automate temporisé suivant :