Université Lille 1 - Sciences et Technologies
Licence Informatique 2009–2010

**UE AL - Automates et Langages**

---

# Examen – Session 1

*Durée :* $3\ heures$ – Tous documents autorisés

### Instructions de composition

*Chaque exercice (il y en a 4 au total) sera résolu sur (au moins) une feuille à part entière de sorte à pouvoir diviser l'ensemble de vos copies en 4 ensembles (un par exercice). Merci !*

## 1 Automates finis

On s'intéresse au fonctionnement d'une porte de garage motorisée commandée par l'intermédiaire d'un bouton d'ouverture ($O$) et d'un bouton de fermeture ($F$). On modélise le fonctionnement de cette porte par un automate dont les états représentent la position de la porte et l'alphabet représente les actions reçues de l'utilisateur par l'intermédiaire des boutons (c'est-à-dire $\Sigma = \{O, F\}$). On suppose que, quel que soit l'état courant de la porte, celle-ci peut recevoir un signal d'ouverture ou de fermeture. On appelle *trace d'exécution* une séquence finie de signaux $O/F$ reçus par la porte ; en d'autres termes il s'agit d'un mot sur l'alphabet $\Sigma = \{O, F\}$.

**Exercice 1 : *Porte simple***
On s'intéresse, dans un premier temps, à l'automate déterministe (AFD) $A_1$ qui modélise une porte à deux états, $Q_1 = \{\textit{Ouverte}, \textit{Fermée}\}$. L'automate $A_1$ doit satisfaire la spécification suivante :

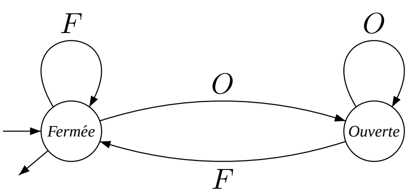
**(S$_1$)** quel que soit l'état courant de l'automate, celui-ci peut recevoir un signal d'ouverture et de fermeture ;

**(S$_2$)** l'automate a pour état initial l'état *Fermée*,

**(S$_3$)** l'automate a pour *unique* état acceptant, l'état *Fermée*.

**Q 1 .** Définir (par un diagramme de transition par exemple) l'AFD $A_1$.
*Indication* : votre automate doit être *complet*.

<u>*Solution:*</u>

$A_1 =$ 


**Q 2 .** Par la méthode de votre choix, déterminer une expression rationnelle représentant toutes les traces d'exécution possibles de $A_1$.

<u>*Solution:*</u>
$\mathcal{L}(A_1) = (O^*F)^*$

**Exercice 2 : *États transitoires***

*Prendre une nouvelle feuille !*

On souhaite construire l'automate non-déterministe avec $\epsilon$-transitions ($\epsilon$-AFN) $A_2$ en ajoutant à $A_1$ les états *transitoires* représentant la porte en cours d'*Ouverture* et en cours de *Fermeture*, c'est-à-dire $Q_2 = \{$*Ouverte*, *Ouverture*, *Fermeture*, *Fermée*$\}$. On appelle *état stable* l'un des états *Ouverte* ou *Fermée*. $A_2$ doit réaliser, en plus de la spécification précédente (**S**$_1$ à **S**$_3$), la spécification suivante :

**(S$_4$)** s'il y a une transition d'un état stable ou transitoire à un état transitoire, celle-ci est réalisée par la réception d'un signal *O* ou *F* ;

**(S$_5$)** s'il y a une transition d'un état transitoire à un état stable, celle-ci est réalisée sans réception de signal, c'est-à-dire par une $\epsilon$-transition ;

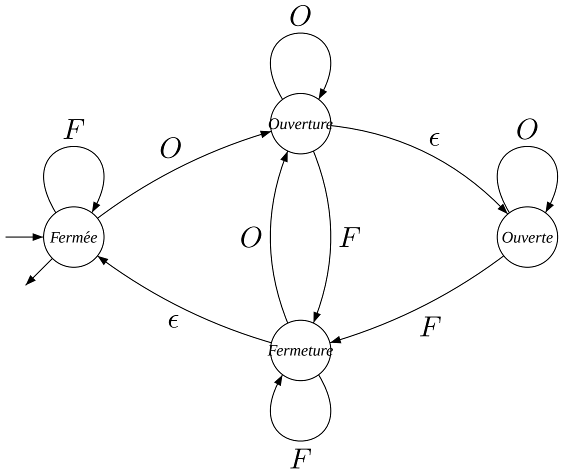
**(S$_6$)** Entre toute paire d'états transitoires il y a une transition.

**Q 1 .** Définir (par un diagramme de transition par exemple) l'automate $A_2$.

*Indication* : votre automate est un $\epsilon$-AFN *complet*. Commencer par définir l'automate, et vérifier ensuite qu'il satisfait la spécification attendue.

*Solution:*

$A_2 =$

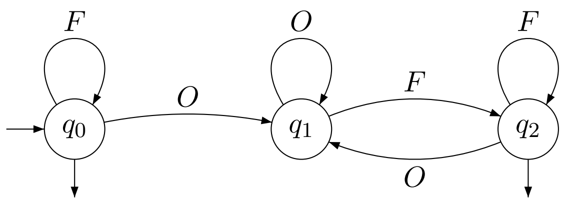



**Q 2 .** Construisez l'AFD $A_3$ résultant de l'élimination les $\epsilon$-transitions de $A_2$.

*Solution:*

| | $O$ | $F$ |
|---|---|---|
| $q_0 = \{$*Fermée*$\}$ | $\{Ouverture, Ouverte\}$ | $\{$*Fermée*$\}$ |
| $q_1 = \{Ouverture, Ouverte\}$ | $\{Ouverture, Ouverte\}$ | $\{$*Fermeture, Fermée*$\}$ |
| $q_2 = \{$*Fermeture, Fermée*$\}$ | $\{Ouverture, Ouverte\}$ | $\{$*Fermeture, Fermée*$\}$ |

$A_3 =$




**Q 3 .** Déterminer une expression rationnelle représentant toutes les traces d'exécution possibles de $A_2$.

*Solution:*
Soient $X_0 = $ *Fermée*, $X_1 = Ouverture$, $X_2 = Ouverte$, et $X_3 = Fermeture$, on a :

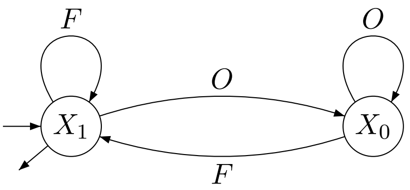
$$\begin{array}{rcl}
X_0 &=& FX_0 + OX_1 + \epsilon \\
X_1 &=& OX_1 + FX_3 + \epsilon X_2 \\
X_2 &=& OX_2 + FX_3 \quad \Rightarrow X_2 = O^*FX_3 \\
X_3 &=& FX_3 + OX_1 + \epsilon X_0 \\
\Rightarrow X_1 &=& OX_1 + (F + O^*F)X_3 = OX_1 + O^*FX_3 \quad \Rightarrow X_1 = O^*O^*FX_3 = O^*FX_3 \\
\Rightarrow X_3 &=& FX_3 + O^+FX_3 + \epsilon X_0 = O^*FX_3 + X_0 \quad \Rightarrow X_3 = (O^*F)^*X_0 \\
\Rightarrow X_0 &=& FX_0 + O(O^*F)(O^*F)^*X_0 + \epsilon \\
\Rightarrow X_0 &=& (F + O(O^*F)^+)^* = \mathcal{L}(A_2)
\end{array}$$

**Q 4 .** Donner l'AFD complet minimal $A_4$ équivalent à $A_2$.

*Solution:*

| | $\equiv_0$ | |
|---|---|---|
| | $O$ | $F$ |
| $q_0$ | $X_0$ | $X_1$ |
| $q_1$ | $X_0$ | $X_1$ |
| $q_2$ | $X_0$ | $X_1$ |

$$\begin{array}{rl}
\equiv_0: & X_0 = \{q_1\}, X_1 = \{q_0, q_2\} \\
\equiv_1: & X_0 = \{q_1\}, X_1 = \{q_0, q_2\} \\
\Rightarrow & \equiv_0 = \equiv_1 = \equiv
\end{array}$$

$A_4 =$ 


**Q 5 .** L'automate $A_2$ possède-t-il le même ensemble de traces d'exécutions que l'automate $A_1$ ? Justifier. *Indication* : En d'autres termes, reconnaissent-ils le même langage ? Attention, il ne suffit pas de comparer "naïvement" (i.e. syntaxiquement) les expressions rationnelles demandées précédemment...

*Solution:*
$A_2$ reconnaît le même langage que $A_4$ qui est lui-même isomorphe à $A_1$. Par conséquent, $A_1$ et $A_2$ reconnaissent le même langage.

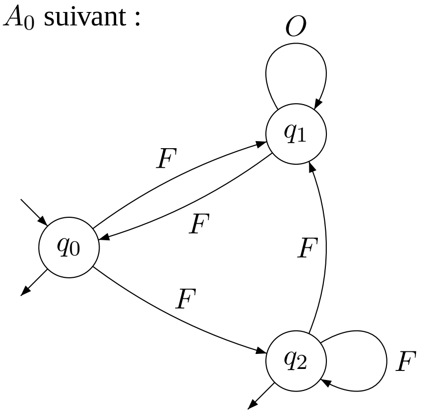
## 2 Logique du 1er ordre

On considère le langage $\mathcal{F}$ des formules de la logique du premier ordre construites à l'aide :
– d'un ensemble de variables $\{x, y, z, \ldots\}$ ;
– des deux symboles de relation unaires *init* et *accept* et des trois symboles de relation binaires *close*, *open* et *epsilon* ;
– d'un ensemble de symboles de constantes $\{q_0, q_1, \ldots\}$.

Ce langage ne contient donc pas de symboles de fonctions. On considère également la famille $\mathcal{A}$ des automates (AFD, AFN ou $\epsilon$-AFN) sur l'alphabet $\{O, F\}$. À tout automate $A = (Q, \{O, F\}, \delta, q_0, Acc) \in \mathcal{A}$, où *Acc* est l'ensemble des états acceptant, on associe la structure $M_A$ qui interprète les formules de la famille $\mathcal{F}$ de la façon suivante :
– le domaine des valeurs interprétant les symboles de constantes est l'ensemble des états $Q$ de $A$ (rappel : c'est aussi l'ensemble des valeurs auxquelles les variables sont assignées, donc "$\forall x$" signifie "pour tout état $x$") ;
– *init*$(x)$ signifie que l'état $x$ est un état initial de $A$,
– *accept*$(x)$ signifie que l'état $x$ est un état acceptant de $A$,
– *open*$(x, y)$ signifie qu'il existe une transition étiquetée par $O$ de l'état $x$ vers l'état $y$ ;
– *close*$(x, y)$ signifie qu'il existe une transition étiquetée par $F$ de l'état $x$ vers l'état $y$ ;
– *epsilon*$(x, y)$ signifie qu'il existe une $\epsilon$-transition de l'état $x$ vers l'état $y$ ;

– $x = y$ a le sens habituel de l'égalité et signifie que l'état $x$ est égal à l'état $y$.
On dira qu'un automate $A \in \mathcal{A}$ est un modèle d'une formule $f \in \mathcal{F}$, noté $A \models f$, si la formule $f$ est vraie pour l'interprétation $I_A$.

**Exemple :** Considérons l'automate $A_0$ suivant :




On a
– $A_0 \models \forall x \forall y(\mathit{open}(x, y) \Rightarrow x = y)$, c'est-à-dire "dans l'automate $A_0$, pour tous états $x$ et $y$, s'il y a une transition de $x$ à $y$ étiquetée $O$ alors $x$ et $y$ représentent le même état", et
– $A_0 \models \neg\exists x(\mathit{accept}(x) \wedge \exists y\, \mathit{open}(y, x))$ car "dans l'automate $A_0$, il n'existe pas d'état acceptant $x$ ayant un transition entrante depuis un certain état $y$ et qui soit étiquetée par $O$".

**Exercice 3 :** ***Formalisation des automates***
*Prendre une nouvelle feuille !*
**Q 1 .** Que peut-on dire d'un langage accepté par un automate $A$ qui est modèle de la formule suivante :

$$f_0 = \exists x \exists y \exists z(\mathit{init}(x) \wedge \mathit{open}(x, y) \wedge \mathit{close}(y, y) \wedge \mathit{open}(y, z) \wedge \mathit{accept}(z))$$

*Solution:*
Ce langage contient les mots $OF^*O$.

**Q 2 .** Déterminer les formules $f_i$ suivantes du langage $\mathcal{F}$ pour lesquelles on donne la traduction en langage naturel :
– $A \models f_1$ ssi "dans A, il existe un état acceptant" ;
– $A \models f_2$ ssi "dans A, il existe un seul et unique état acceptant" ;
– $A \models f_3$ ssi "dans A, l'état initial est un état acceptant" ;
– $A \models f_4$ ssi "A est un automate complet"

*Solution:*
$f_1 = \exists x\, \mathit{accept}(x)$, $f_2 = \exists x(\mathit{accept}(x) \wedge \forall y(\mathit{accept}(y) \Rightarrow y = x))$,
$f_3 = \forall x(\mathit{init}(x) \Rightarrow \mathit{accept}(x))$, $f_4 = \forall x(\exists y\, open(x, y) \wedge \exists z\, close(x, z))$

**Q 3 .** En supposant que l'automate $A$ est un $\epsilon$-AFN qui ne possède que 3 états, donner une formule $f_5$ pour laquelle $A$ est un modèle dès qu'il existe dans $A$ une séquence (éventuellement vide) d'$\epsilon$-transitions menant de l'état initial à un état acceptant.

*Solution:*
$f_5 = \exists x(init(x) \wedge (accept(x) \vee \exists y(epsilon(x, y) \wedge accept(y)) \vee \exists y \exists z(epsilon(x, y) \wedge epsilon(y, z) \wedge accept(z))))$

**Q 4 .** ($\star\star\star$) Peut-on répondre à la question précédente si on ne connaît pas le nombre d'états (ou une borne supérieure) de $A$ ? Justifier.
*Indication* : Question difficile, ne pas y perdre trop de temps...

*Solution:*
Non : il faudrait pouvoir exprimer l'induction dans la logique du 1er ordre ce qui n'est pas possible...

On considère les formules suivantes

$$\begin{array}{rcl} f_6 & = & \exists y(init(x) \wedge open(x,y) \wedge close(y,z) \wedge accept(z)) \\ f_7 & = & \forall x'(epsilon(x',y') \wedge accept(y')) \end{array}$$

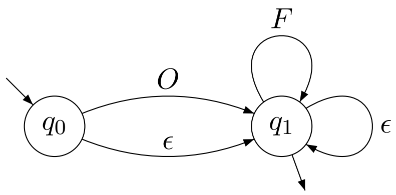
**Q 5 .** Déterminer les variables libres et les variables liées des formules $f_6$ et $f_7$.

<u>*Solution:*</u>
$fv(f_6) = \{x, z\}, bv(f_6) = \{y\}$ , $fv(f_7) = \{y'\}, bv(f_8) = \{x'\}$

**Q 6 .** Donner une assignation $\sigma$ des variables libres des formules $f_6$ et $f_7$ et un automate $A_4$ *à deux états* qui est modèle de $f_6$ et $f_7$ pour $\sigma$ (c'est-à-dire tel que $A_4 \models f_7[\sigma]$ et $A_4 \models f_8[\sigma]$).
*Indications* : ne pas oublier de préciser l'état initial et le (ou les) état(s) acceptant(s).

<u>*Solution:*</u>

$\sigma = \{x \mapsto q_0, z \mapsto q_1, y' \mapsto q_1\}$, $A_4 =$




**Exercice 4 :** ***Formalisation de la porte de garage***
*Prendre une nouvelle feuille !*
**Q 1 .** Traduire en formules de $\mathcal{F}$, les assertions de spécification $\mathbf{S}_1$ à $\mathbf{S}_3$ de l'exercice 1.

<u>*Solution:*</u>

$$\begin{array}{rcl} \mathbf{S}_1 & = & \forall x(\exists y\, open(x,y) \wedge \exists z\, close(x,z)) \\ \mathbf{S}_2 & = & init(\textit{Fermée}) \\ \mathbf{S}_3 & = & \forall x(accept(x) \Rightarrow x = \textit{Fermée}) \end{array}$$

**Q 2 .** Traduire en formules de $\mathcal{F}$, les assertions de spécification $\mathbf{S}_4$ à $\mathbf{S}_6$ de l'exercice 2.

<u>*Solution:*</u>

$$\begin{array}{rcl} \mathbf{S}_4 & = & \forall x \forall y((y = Fermeture \vee y = Ouverture) \Rightarrow \neg epsilon(x,y)) \\ \mathbf{S}_5 & = & \forall x \forall y\big((x = Fermeture \vee x = Ouverture) \wedge (y = \textit{Fermée} \vee y = \textit{Ouverte})\big) \Rightarrow \\ & & (\neg open(x,y) \wedge \neg close(x,y)) \\ \mathbf{S}_5 & = & \forall x \forall y\big((x = Fermeture \vee x = Ouverture) \wedge (y = \textit{Fermeture} \vee y = \textit{Ouverture})\big) \Rightarrow \\ & & (open(x,y) \wedge close(x,y) \wedge epsilon(x,y)) \end{array}$$